



Dear Readers,

This SNE issue SNE 20/3-4 completes the issue in the 20th anniversary of SNE – an occasion to look back – as in SNE 20/1, and an occasion to look into the future. In 2011 we will introduce novelties and changes in SNE content, structure, layout and distribution.

First, the change of SNE's name from Simulation News Europe to Simulation Notes Europe underlines the focus of SNE; second, SNE is now clearly structured as Print SNE (Print ISSN 2305-9974) and Online SNE (Online ISSN 2306-0271), where we follow the Open Access strategy for basic publication and EUROSIM member subscription for extended publication; third, SNE contributions are identified by a DOI (Digital Object Identifier) assigned to the publisher ARGESIM (DOI prefix 10.11128), allowing better citation and cross referencing; and fourth, SNE Volume 21 starts with a new layout.

Another novelty is happening on the fly SNE's publication strategy with respect to submission of contribution. SNE has published, is publishing and will publish peer reviewed 'Notes' on developments and trends in modelling and simulation in various areas and in application and theory, with main topics being simulation aspects and interdisciplinarity – whereby individual submissions of scientific papers are welcome, as well as post-conference publications of contributions from conferences of **EUROSIM** societies. Up to now many contribution were coming from this 'second' contributions source, but from 2011 on this possibility has been fixed with all EUROSIM member societies – and is open also for future members.

The contributions of this 'last SNE' - Simulation News Europe - issue are mainly post-conference publication of MATHMOD 2009 Vienna (co-organised by ASIM, by support from SLOSIM), and from SIMS Conference 2009 (SIMS 50, Denmark, Frederica).


We hope, readers enjoy the novelties and the content, and we thank all contributors, members of the editorial boards, and people of our ARGESIM staff for co-operation in producing this SNE issue; and on to the next SNE issue, the 'first SNE' - Simulation Notes Europe - issue..

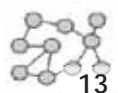


Felix Breitenecker, editor-in-chief, eic@sne-journal.org

SNE 20/3-4 in Three Minutes


Impressum, Table of Contents: page 4


 5 *A Machine-Job Incidence Matrix* models and supports simulation of manufacturing systems.

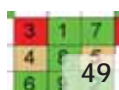
 13 *Declarative Resource Discovery* analyses distributed automation systems

$B = \{b_1, b_2, \dots\}$
 $\forall b \in B$ 21 *Intelligent System Development: Agent reasoning on Trust and reputation*


$\sum_{i=1}^n x_i \theta^i$ 29 *Stochastic Models for Intermittent Demands Forecasting and Stock control*

 37 *Decision Making with a Random Walk in a Discrete Time Markov Chain*


 43 *CPL Clustering Based on Linear Dependencies*

 *Alternately Evolving Genetic Algorithm* - A proposal and its application

 55 *Research and Education : MATLAB-based Robot Control Design Environment*

 67 *RapOpt* – An Automation Tool for Production-orientated Run-to-run Model Evaluation.

$2H_2O \rightleftharpoons H_2 + O_2$
 k_1, k_2, K_2 75 *Computational Methods for Reaction Equilibrium* - A Comparison

 *EUROSIM short info* gives a short overview over EUROSIM and its national societies, and provides contact addresses.
6 pp.



ASIM



ASIM



ASIM - Buchreihen / ASIM Book Series

Fortschritte in der Simulationstechnik (FS) / Frontiers in Simulation (FS) Proceedings Conferences - Monographs, Proceedings:

- I. Troch, F. Breitenacker (eds): *Proceedings MATHMOD 09- Abstract Volume / Full Papers CD Volume*. Proc. 6th Conference Mathematical Modelling Vienna, February 2009, Vienna; ARGESIM Reports no. 34 & no. 35, ASIM/ARGESIM Vienna, 2009; ISBN 978-3-901608-34-6, ISBN 978-3-901608-35-3
- M. Rabe (ed.): *Advances in Simulation for Production and Logistics Applications*. Proc. 13. ASIM-Fachtagung Simulation in Produktion und Logistik, October 2008, Berlin; Fraunhofer IRB-Verlag, Stuttgart, 2008, ISBN 978-3-8167-7798-4.
- B. Zupančič, R. Karba, S. Blažič (eds.): *Proceedings EUROSIM 2007 - Abstract Volume / CD Volume*. Proc. 6th EUROSIM Congress on Modelling and Simulation, Sept. 2007, Ljubljana, Slovenia. ARGESIM Report no. 32, ASIM/ARGESIM Vienna, 2007; ISBN 978-3-901608-32-2.
- S. Collisi-Böhmer, O. Rose, K. Weiß, S. Wenzel (Hrsg.): *Qualitätskriterien für die Simulation in Produktion und Logistik*. AMB 102, Springer, Heidelberg, 2006; ISBN 3-540-35272-4.
- M. Rabe, S. Spiekermann, S. Wenzel (Hrsg.): *Verifikation und Validierung für die Simulation in Produktion und Logistik*. AMB 103, Springer, Heidelberg, 2006; ISBN 3-540-35281-3.
- W. Borutzky: *Bond Graphs Methodology for Modelling Multidisciplinary Dynamic Systems*. FS 14, ISBN 3-936150-33-8, 2005.

Fortschrittsberichte Simulation (FB) - ARGESIM Reports (AR) - Special Monographs, PhD Theses, ASIM Workshop Proceedings

- D. Leitner: *Simulation of Arterial Blood Flow with the Lattice Boltzmann Method* ARGESIM Report 16, ASIM/ARGESIM Vienna, 2009; ISBN 978-3-901608-66-7.
- Th. Löscher: *Optimisation of Scheduling Problems Based on Timed Petri Nets*. ARGESIM Report 15, ASIM/ARGESIM Vienna, 2009; ISBN 978-3-901608-65-0.
- R. Fink: *Untersuchungen zur Parallelverarbeitung mit wissenschaftlich-technischen Berechnungsumgebungen*. ARGESIM Report 12, ASIM/ARGESIM Vienna, 2008; ISBN 978-3-901608-62-9.
- M. Gyimesi: *Simulation Service Providing als Webservice zur Simulation Diskreter Prozesse*. ARGESIM Report 13, ASIM/ARGESIM Vienna, ISBN 3-901-608-63-X, 2006.
- J. Wöckl: *Hybrider Modellbildungszugang für biologische Abwasserreinigungsprozesse*. ARGESIM Report 14, ASIM/ARGESIM Vienna, ISBN 3-901608-64-8, 2006.
- H. Ecker: *Suppression of Self-excited Vibrations in Mechanical Systems by Parametric Stiffness Excitation*. ARGESIM Report 11, ISBN 3-901-608-61-3, 2006.
- C. Deatcu, P. Dünow, T. Pawletta, S. Pawletta (eds.): *Proceedings 4. ASIM-Workshop Wismar 2008 - Modellierung, Regelung und Simulation in Automotive und Prozessautomation*. ARGESIM Report 31, ASIM/ARGESIM Vienna, ISBN 978-3-901608-31-5, 2008.
- J. Wittmann, H.-P. Bader (Hrsg.): *Simulation in Umwelt- und Geowissenschaften - Workshop Dübendorf 2008*. Shaker Verlag, Aachen 2008, ISBN 978-3-8322-7252-4.
- A. Gnauck (Hrsg.): *Modellierung und Simulation von Ökosystemen - Workshop Kölpinsee 2006*. Shaker Verlag, Aachen 2007, AM 107; ISBN 978-3-8322-6058-3.

Available / Verfügbar: ASIM/ARGESIM Publisher Vienna - WWW.ASIM-GI.ORG
SCS Publishing House e.V., Erlangen, WWW.SCS-PUBLISHINGHOUSE.DE
Download ASIM Website WWW.ASIM-GI.ORG (partly; for ASIM members),
Bookstores / Buchhandlung, tw. ermäßigter Bezug für ASIM Mitglieder WWW.ASIM-GI.ORG



REPORTS



REPORTS

**Table of Contents**

TN	Modeling and Simulation of Manufacturing Systems Based on a Machine-Job Incidence Matrix. <i>I. Sindicic, T. Petrovic, S. Bogdan</i>	5
TN	Declarative Resource Discovery in Distributed Automation Systems <i>C. Gerdes, C. Kleegrewe, J. P. Müller</i>	13
TN	Agent Reasoning Based on Trust and Reputation <i>W.J. Samek, F. Zboril</i>	21
TN	Stochastic Models for Intermittent Demands Forecasting and Stock control <i>W. Sandmann, O. Bober</i>	29
TN	Decision Making with a Random Walk in a Discrete Time Markov Chain. <i>R. Chelvier, G. Horton, C. Krull, B. Rauch-Gebbensleben</i> ..	37
TN	CPL Clustering Based on Linear Dependencies. <i>L. Bobrowski</i>	43
TN	The Proposal of the Alternately Evolving Genetic Algorithm and its Application. <i>Z. Chen, K. Zhou, Y. Liu, W. Zhan</i>	49
FN	MATLAB-based Robot Control Design Environment for Research and Education. <i>L. Žlajpah, B. Nemeč, D. Omrcen</i>	55
SV	RapOpt – An Automation Tool for Production-orientated Run-to-run Model Evaluation. <i>N Violet, N. Rossner, T. Heine, R. King</i>	67
TN	Comparison of Computational Methods for Reaction Equilibrium. <i>S. A. Jayarathna, B. Lie, M. C. Melaen</i>	75
⊕	SNE News Section:	
	• EUROSIM short info.....	N1- N6

Vlatko Čerić, *vceric@efzg.hr*
Univ. Zagreb, Fac. of Organization and Informatics, Croatia

Russell Cheng, *rhc@maths.soton.ac.uk*
University of Southampton, Fac. Mathematics/OR Group, UK

Horst Ecker, *Horst.Ecker@tuwien.ac.at*
Vienna University of Technology, Inst. f. Mechanics, Austria

Edmond Hajrizi, *ehajrizi@ubt-uni.net*
University for Business and Technology, Pristina, Kosovo

András Jávör, *javor@eik.bme.hu*,
Budapest Univ. of Technology and Economics, Hungary

Esko Juuso, *esko.juuso@oulu.fi*
Univ. Oulu, Dept. Process/Environmental Engineering, Finland

Rihard Karba, *rihard.karba@fe.uni-lj.si*
University of Ljubljana, Fac. Electrical Engineering, Slovenia

Francesco Longo, *f.longo@unical.it*
Univ. of Calabria, Mechanical Department, Italy

David Murray-Smith, *d.murray-smith@elec.gla.ac.uk*
University of Glasgow, Fac. Electrical Engineering, UK

Thorsten Pawletta, *pawel@mb.hs-wismar.de*
Univ. Wismar, Dept. Computational Engineering, Germany

Niki Popper, *niki.popper@drahtwarenhandlung.at*
dwh Simulation Services, Vienna, Austria

Thomas Schriber, *schriber@umich.edu*
University of Michigan, Business School, USA

Peter Schwarz, *Peter.Schwarz@eas.iis.fraunhofer.de*
Fraunhofer Foundation-Design Automation Dresden, Germany

Yuri Senichenkov, *sneyb@dcn.infos.ru*
St. Petersburg Technical University, Russia

Sigrid Wenzel, *S.Wenzel@uni-kassel.de*
University Kassel, Inst. f. Production Technique, Germany

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
office@sne-journal.org; www.sne-journal.org

SNE 20(3-4) Reprint doi: 10.11128/sne.20.34.0998

SNE Editorial Board

SNE - Simulation News Europe (SNE) is advised and supervised by an international editorial board. This board is taking care on peer reviewing and handling of *Technical Notes, Education Notes, Short Notes, Software Notes*, and of *Benchmark Notes* (definitions and solutions). Work of the board is supported by a new SNE Contribution-, Management-, and Reviewing System via the new SNE website. At present, the board is increasing:

Felix Breitenecker, *Felix.Breitenecker@tuwien.ac.at*
Vienna University of Technology, Austria, Editor-in-chief

Peter Breedveld, *P.C.Breedveld@el.utwente.nl*
University of Twente, Div. Control Engineering, Netherlands

Agostino Bruzzone, *agostino@itim.unige.it*
Universita degli Studi di Genova, Italy

Francois Cellier, *fcellier@inf.ethz.ch*
ETH Zurich, Institute for Computational Science, Switzerland

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 benchmarks 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, markus.wallerberger@gmx.at

Administration, Web: Anna Mathe, anna.mathe@tuwien.ac.at

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

Publisher: ARGESIM/ASIM; c/o Inst. for Scientific Computing, TU Vienna, Wiedner Hauptstrasse 8-10, 1040 Vienna, Austria, and ASIM (German Simulation Society), c/o Wohlfartstr. 21b, 80939 München; © ARGESIM/ASIM 2010

Modeling and Simulation of Manufacturing Systems based on a Machine-Job Incidence Matrix

Ivica Sindicic, Exor d.o.o, Zagreb, Croatia

Tamara Petrovic, Stjepan Bogdan, University of Zagreb, Croatia

SNE Simulation Notes Europe SNE 20(3-4), 2010, 5-12, doi: 10.11128/sne.20.tn.09981

This paper presents a method for modeling and simulation of particular class of manufacturing systems. The method is based on so called Machine-Job Incidence Matrix (MJI) that is formed from Steward sequencing matrix and Kusiak machine-part incidence matrix. A model of the system in a form of MJI matrix is easy to comprehend. It can be put in direct relation with other manufacturing systems analysis tools, such as Petri nets and MS matrix model. Moreover, structural properties of MJI matrix offer direct insight in the system configuration, hence, providing a ground for the system analysis and supervisory controller design. Properties, such as circular waits and conflicts, can be determined straightforwardly by using simple matrix manipulations, thus, allowing design of sequencing control algorithms. An extension of MJI matrix in time domain offers a foundation for determination of recursive equation that can be used for simulation of system's dynamics. Although manufacturing systems have been used for validation of the proposed modeling technique, the method can be applied on other discrete event systems as well.

Introduction

Using today's classification of systems, manufacturing systems (MSs) can be treated as hybrid systems that contain a mixture of various dynamic behaviors—continuous and discrete control loops, Boolean variables related to process states, and discrete events, all embraced by a usually hierarchical decision-making overhead.

This means that an MS structure contains both hard and soft technology, first focused on the product fabrication, assembly and distribution, while later the focus is on the support and coordination of manufacturing operations. The MS's hard technology is split into several levels – from the factory level via the operating center, workcell and robotic station levels to a particular manufacturing process level. The accompanying soft technology is also split into several levels – from the highest strategy level, via lower planning, supervisory, and manipulating levels to the basic manufacturing task level.

Today, simulation models provide a very inexpensive and convenient way for complete factory design. Instead of building real systems, a designer first builds new factory layouts and defines resource configurations in the virtual environment and refines them without actual production of physical prototypes. Allowing clear understanding of all potential problems caused by the factory layout and/or dispatching strategy, modeling and dynamic simulation of manufacturing processes has traced a completely

new route to analysis and design of MSs [1–3]. Simulation of robotized manufacturing systems has become much easier and more effective with specialized programs for virtual-factory modeling and simulation. Many virtual-factory simulators have origin in the academia [4–8]. Each of these tools has a mathematical core in a form of an algorithm used to describe dynamic behavior of MS elements. In this paper we exploit so called machine-job incidence matrix (MJI) for purpose of deriving such an algorithm.

The paper is organized in the following way. In the next section, we describe construction of MJI matrix, which is based on two well-known matrices: resource requirements matrix [9], also known as *machine-part incidence matrix* (MPI), and Steward sequencing matrix [10], also referred as *design structure matrix* (DSM). In Section 1 an extension of MJI to design of MS recursive simulation model is given, followed by illustrative example. In Section 2 the recursive mathematical model of *free choice multiple reentrant flowlines* (FMRF) is presented in detail. We consider FMRF systems with multiple flowlines, where resources can hold an arbitrary number of parts simultaneously (*k-limited* systems). In Sections 2.5 and 2.6 a mathematical framework that allows testing of various control policies during the system simulation is presented. The recursive model forms the basis of the developed system simulator, which is presented in Section 3. We conclude the paper with final remarks and an outline for future work.

1 Construction of the machine-job incidence matrix

We start with introduction of basic terms that will be used throughout the paper. Let Π be the set of distinct types of parts produced (or customers served) by a flexible manufacturing system (FMS). Then each part type $P_k \in \Pi$ is characterized by a predetermined sequence of job operations $J^k = \{J_1^k, J_2^k, \dots, J_{L_k}^k\}$ with each operation employing at least one resource (note that some of these job operations may be similar, e.g. J_i^k and J_j^k with $i \neq j$ may both be drilling operations). We uniquely associate with each job sequence J^k , the operations of raw part-in, J_{in}^k , and finished product-out, J_{out}^k . Denote the system resources with $R = \{r_i\}_{i=1}^n$, where $r_i \in R$ can represent a pool of multiple resources each capable of performing the same type of job operation. In this notation, $R_k \subset R$ represents the set of resources utilized by job sequence J^k . Note that $R = \bigcup_{k \in \Pi} R^k$ and $J = \bigcup_{k \in \Pi} J^k$ represent all resources and jobs in a particular FMS. Since the system could be re-entrant, a given resource $r \in R^k$ may be utilized for more than one operation $J_i^k \in J^k$ (*sequential sharing*). Also, certain resources may be used in the processing of more than one part-type so that for some $\{l, k\} \in \Pi$, $l \neq k$, $R^l \cap R^k \neq \emptyset$ (*parallel sharing*). Resources that are utilized by more than one operation in either of these two ways are called *shared resources*, while the remaining are called *non-shared resources*. Thus, one can partition the set of system resources as $R = R_s \cup R_{ns}$, with R_s and R_{ns} indicating the sets of shared and non-shared resources, respectively, where $|R_s| = n_s$ and $|R_{ns}| = n_{ns}$, $n_s + n_{ns} = n$. For any $r \in R$ we define the resource job set $J(r)$. Obviously, $|J(r)| = 1$ (> 1) if $r \in R_{ns}$ ($r \in R_s$). The previously defined job set J and resource set R are associated with MJI matrix in the form of vectors. A *job vector* $V: J \rightarrow \mathfrak{K}$ and a *resource vector* $R: R \rightarrow \mathfrak{K}$ represent the set of jobs and the set of resources corresponding to their nonzero elements. The set of jobs (resources) represented by $V(R)$ is called the support of $V(R)$, denoted $\text{sup } V(\text{sup } R)$; i.e. given $V = [v_1 v_2 \dots v_q]^T$, vector element $v_i > 0$ if and only if $v_i \in \text{sup } V$. In the same manner, given $R = [r_1 r_2 \dots r_q]^T$, vector element $r_i > 0$ if and only if resource $r_i \in \text{sup } R$.

MJI matrix, used herein, describes *free-choice multiple re-entrant flowline* (FMRF) class of manufacturing systems [11]. This class has the following properties: each operation in the system requires one and only one resource with no two consecutive jobs using the same resource, i.e. $\forall J_i^k \in J, R(J_i^k) = 1$ and $R(J_i^k)$

$\neq R(J_{i+1}^k)$, there are no assembly jobs, and there are shared resources in the system. In FMRF some jobs have the option of being machined in a resource from a set of resources (routing of jobs), and each resource might be used to machine different jobs. For each job that can be performed by more than one resource, there exists a material handling buffer (routing resources) that routes parts. A subclass of FMRF systems that does not allow routing of jobs, i.e. each job is executed by a single pre-assigned resource, is called *multiple re-entrant flowlines* (MRF). Having defined basic terms we proceed with MJI construction. DSM is a square matrix containing a list of tasks in the rows and columns with matrix elements indicating the execution sequence. In case of FMRF systems DSM is subdiagonal identity matrix of the following form

$$\Gamma = \begin{matrix} & \begin{matrix} J_1^1 & J_2^1 & \dots & J_{L_1}^1 & J_1^2 & \dots & J_{L_2}^2 & \dots & J_1^m & \dots & J_{L_m}^m \end{matrix} \\ \begin{matrix} J_1^1 \\ J_2^1 \\ \vdots \\ J_{L_1}^1 \\ J_1^2 \\ J_2^2 \\ \vdots \\ J_{L_2}^2 \\ \vdots \\ J_1^m \\ \vdots \\ J_{L_m}^m \end{matrix} & \begin{vmatrix} 0 & 0 & \dots & 0 & & & & & & & \\ 1 & 0 & \dots & 0 & & & & & & & \\ \vdots & \vdots & \ddots & \vdots & & & & & & & \\ 0 & 0 & \dots & 0 & & & & & & & 0 \\ & & & & 0 & & & & & & \\ & & & & & \ddots & & & & & \\ & & & & & & 0 & & & & \\ & & & & & & & \ddots & & & \\ & & & & & & & & 0 & & \\ & & & & & & & & & \ddots & \\ & & & & & & & & & & 0 \end{vmatrix} \end{matrix}$$

The order of tasks in the rows or columns indicates the execution sequence. The relationships among the tasks are usually represented by '1' in the corresponding cells, i.e. if task j is immediate predecessor of task i then DSM element (i, j) is equal to 1, otherwise it is zero. The second matrix, used for system description, is MPI. It captures relations between resources and parts processed by a system. Generally, MPI has the following form,

$$\Theta = \begin{matrix} & \begin{matrix} R_1 & R_2 & \dots & R_q \end{matrix} \\ \begin{matrix} P_1 \\ P_2 \\ \dots \\ P_p \end{matrix} & \begin{vmatrix} 0/1 & 0/1 & \dots & 0/1 \\ 0/1 & 0/1 & \dots & 0/1 \\ \dots & \dots & \dots & \dots \\ 0/1 & 0/1 & \dots & 0/1 \end{vmatrix} \end{matrix}$$

where 0/1 represents entry of 0 or 1 depending on relation between corresponding part and resource: if resource j is processing a part i then MPI element (i, j) is equal to 1, otherwise is zero.



Since the sequence $J^k = \{J_1^k, J_2^k, \dots, J_{L_k}^k\}$ represents P_k processing order, by combining DSM and MPI matrices, we get general form of machine-job incidence matrix for an FMRF system as

$$\Lambda = \begin{matrix} & R_1 & R_2 & \dots & R_q \\ \begin{matrix} J_1^1 \\ J_2^1 \\ \dots \\ J_{L_1}^1 \\ J_1^2 \\ J_2^2 \\ \dots \\ J_{L_2}^2 \\ \dots \\ J_1^m \\ J_2^m \\ \dots \\ J_{L_m}^m \end{matrix} & \begin{vmatrix} 0/1 & 0/1 & \dots & 0/1 \\ 0/1 & 0/1 & \dots & 0/1 \\ \dots & \dots & \dots & \dots \\ 0/1 & 0/1 & \dots & 0/1 \\ 0/1 & 0/1 & \dots & 0/1 \\ \dots & \dots & \dots & \dots \\ 0/1 & 0/1 & \dots & 0/1 \\ \dots & \dots & \dots & \dots \\ 0/1 & 0/1 & \dots & 0/1 \\ \dots & \dots & \dots & \dots \\ 0/1 & 0/1 & \dots & 0/1 \end{vmatrix} \end{matrix}$$

In case job i is performed by resource j , matrix element (i, j) is equal to 1, otherwise is zero. It should be noted that, according to definition of FMRF class of systems, some operation in the system could be executed by several resources, hence, multiple ones would appear in corresponding row of MJI matrix. On the other hand, column comprising multiple entries of '1', represents shared resource. If one observes an MRF system, its machine-job matrix will have exactly one '1' in each row, and possibly multiple '1' in columns. Machine-job incidence matrix can be defined separately for each part type in an FMS.

2 Recursive system equations from MJI matrix

2.1 Introduction to modeling of FMRF systems

The system model should provide the insight into states of system jobs and resources. In other words, given the recursive system model and its initial state, the user should, in each discrete event iteration number k , be able to know which jobs are inactive, completed, which resource are available and which are not.

The basic property of FMRF systems is that these systems include jobs that can be performed by more than one resource from the resource pool. The main idea when developing its recursive model is the following: if there is a well-defined job routing strategy, in each step k , it is uniquely decided which resource will be assigned to each job, hence, in each step k system can be seen as an MRF system. Since the resource assigned to a certain job changes during the work of the system, for each step k the FMRF systems will be represented with structurally distinct MRF models.

This method will be explained by the example that follows. Let us consider a system with the following MJI matrix:

$$\Lambda = \begin{matrix} & R_1 & R_2 & R_3 & R_4 \\ \begin{matrix} J_1^1 \\ J_2^1 \\ J_3^1 \end{matrix} & \begin{vmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{vmatrix} \end{matrix}$$

The system embraces three jobs J_1, J_2 and J_3 and four resources R_1, R_2, R_3 and R_4 . Job J_1 can be executed by resources R_1 and R_2 . Job J_2 is performed by resource R_4 and job J_3 by resources R_2 or R_3 . Let us, for instance, define a control strategy such that jobs J_1 and J_3 are performed alternately by assigned resources, i.e. resource for job J_1 is assigned in the following order: $(R_1, R_2, R_1, R_2\dots)$ and for job J_3 : $(R_3, R_2, R_3, R_2\dots)$. Since the first resources to perform considered jobs are R_1 and R_3 , the MJI matrix of MRF substitute model Λ_s in the initial state is:

$$\Lambda_s = \begin{matrix} & R_1 & R_2 & R_3 & R_4 \\ \begin{matrix} J_1^1 \\ J_2^1 \\ J_3^1 \end{matrix} & \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{vmatrix} \end{matrix}$$

When one of the jobs J_1 and J_3 is started, the resource assigned to this job alternates, thus, the Λ_s matrix changes to:

a) job J_1 starts – next resource assigned to it is R_2

$$\Lambda_s = \begin{matrix} & R_1 & R_2 & R_3 & R_4 \\ \begin{matrix} J_1^1 \\ J_2^1 \\ J_3^1 \end{matrix} & \begin{vmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{vmatrix} \end{matrix}$$

b) job J_3 starts – next resource assigned to it is R_2

$$\Lambda_s = \begin{matrix} & R_1 & R_2 & R_3 & R_4 \\ \begin{matrix} J_1^1 \\ J_2^1 \\ J_3^1 \end{matrix} & \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{vmatrix} \end{matrix}$$

c) both jobs J_1 and J_3 start – next resource assigned to them is R_2

$$\Lambda_s = \begin{matrix} & R_1 & R_2 & R_3 & R_4 \\ \begin{matrix} J_1^1 \\ J_2^1 \\ J_3^1 \end{matrix} & \begin{vmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{vmatrix} \end{matrix}$$

As we can see from the example, matrices that describe the system are MRF system matrices since they contain no multiple '1' in rows.

Given the basic idea behind the development of the FMRF system model, the general recursive procedure is given as:

Algorithm. Input: FMRF system's MJI matrix, Job routing strategy.

In each discrete event iteration step k do:

1. Determine Λ_s – MJI matrix of MRF system substitution, from the system state in step $(k - 1)$ and the given routing strategy
2. Calculate the system state in step k based on MRF model (Λ_s)

2.2 Basic recursive model of MRF system

The basic recursive model is developed for MRF system with the following properties: each resource can hold maximally one part at a time and there is one sample of each resource in the resource pool. Hence, each job or resource, seen as a place in Petri-net formalism, can contain maximally one token. Further, a system is autonomous, thus, a part can enter the system each time the resource assigned to the input operation is available. In other words, input buffers always contain parts waiting to be processed.

First, we define the following notation for the rest of the chapter: $\mathbf{x}(k)$ denotes the value of vector \mathbf{x} in discrete step k , while $x_i(k)$ denotes the value of the i -th element of the vector \mathbf{x} in discrete step k . The system MJI matrix is denoted as Λ_s .

Definition 1 (completed jobs vector)

The *completed jobs vector* \mathbf{v} , is a column vector with dimension equal to the number of jobs in the system. $v_i = 1$ if i -th job is completed, otherwise $v_i = 0$.

Definition 2 (idle resource vector)

The *idle resource vector* \mathbf{r} , is a column vector with dimension equal to the number of resources in the system. $r_i = 1$ if the i -th resource is available, otherwise $r_i = 0$.

The state of the system is completely described by the values of completed jobs and idle resources vector. The relation between vector \mathbf{v} and vector \mathbf{r} is:

$$\mathbf{r} = \overline{(\Lambda_s)^T \Delta \mathbf{v}} \quad (1)$$

Besides these two vectors, we define an auxiliary vector \mathbf{r}_v with dimension equal to the number of jobs, whose i -th element, r_{vi} , is equal to '1' if resource assigned to job i is available. Vector \mathbf{r}_v is determined as follows:

$$\mathbf{r}_v = \Lambda_s \Delta \mathbf{r} = \Lambda_s \Delta \overline{(\Lambda_s)^T \Delta \mathbf{v}} \quad (2)$$

The operations on matrices are defined in an *and/or* algebra, denoted Δ and ∇ , where standard multiplication is replaced by logical *and* and standard addition by logical *or*. Given a natural number a , its negation \bar{a} is such that $\bar{a} = 0$ if $a > 0$, otherwise $\bar{a} = 1$.

In general, the completed jobs vector \mathbf{v} is determined as follows:

$$\mathbf{v}(k) = \mathbf{v}(k - 1) + \mathbf{d}(k) \quad (3)$$

Given a certain job i , $d_i(k)$ equals the difference between the number of parts the job i starts processing $d_i^+(k)$ and the number of parts released by the job i in discrete event iteration step k , denoted by $d_i^-(k)$:

$$d_i(k) = d_i^+(k) - d_i^-(k) \quad (4)$$

The job i can start processing a part in step k if the previous job in line, $(i - 1)$ -th job, is completed and if the assigned resource is available:

$$d_i^+(k) = v_{i-1}(k - 1) \cdot r_i(k - 1) \quad (5)$$

The job i can release a part it holds in step k , if it is completed and if the resource assigned to the next job in line, $(i + 1)$ -th job, is available:

$$d_i^-(k) = v_i(k - 1) \cdot r_{i+1}(k - 1) \quad (6)$$

Definition 3 (vector shift)

Let \mathbf{a} be a vector in \mathfrak{R}^m , $\mathbf{a} \in \mathfrak{R}^m$. Vector $\mathbf{b} \in \mathfrak{R}^m$, which is a result of upwards ($m = 1$) or downwards ($m = -1$) vector shift operation, denoted $\mathbf{b} = \uparrow_m \mathbf{a}$, is calculated as:

$$b_j = \begin{cases} 1 & (m = 1 \wedge j = n) \vee (m = -1 \wedge j = 1) \\ a_{j+m} & \text{otherwise} \end{cases}$$

Using vector shift operation, the overall change in value of completed jobs vector \mathbf{v} can then be written as:

$$\begin{aligned} \mathbf{d}(k) &= \mathbf{d}^+(k) - \mathbf{d}^-(k) = \\ &= (\uparrow_{-1} \mathbf{v}(k - 1)) \cdot \mathbf{r}_v(k - 1) - \\ &\quad - \mathbf{v}(k - 1) \cdot (\uparrow_1 \mathbf{r}_v(k - 1)) \end{aligned} \quad (7)$$

A recursive mathematical model is obtained by combining equations (1 – 7) and it can be written in the following form

$$\begin{aligned}
 \mathbf{v}(k) &= \mathbf{v}(k-1) + (\uparrow_{-1} \mathbf{v}(k-1)) \mathbf{r}_v(k-1) - \\
 &\quad - \mathbf{v}(k-1) \cdot (\uparrow_1 \mathbf{r}_v(k-1)) \\
 \mathbf{r}(k) &= (\Lambda_s)^T \Delta \mathbf{v}(k) \\
 \mathbf{r}_v(k) &= \Lambda_s \Delta \mathbf{r}(k)
 \end{aligned} \tag{8}$$

where “ \cdot ” denotes element-wise product of two vectors.

2.3 Recursive model of k -limited MRF systems

In general, MRF systems include resources that can hold more than one part simultaneously. These systems are called k -limited systems, where k denotes the maximum number of parts a single resource can hold at a time. The previously considered systems were 1-limited.

For k -limited systems, elements of vectors \mathbf{v} and \mathbf{r} can obtain values from zero to the maximum number of parts that the corresponding resource can hold – k . That is, if $k > 1$, completed jobs and idle resources vectors comprise integer values unlike binary values for $k = 1$.

If the resource i does not perform any job, the value of r_i corresponds to the maximum number of parts that resource i can hold. Each time a resource i starts processing a part, the value of r_i is decremented until it reaches zero, i.e. for $r_i = 0$ the resource is not available. We can see that the same reasoning was made for 1-limited systems. We introduce a new vector, \mathbf{r}_{cap} , with dimension equal to the number of resources, with $(r_{cap})_i$ equal to the maximal number of parts resource i can hold.

Further, although vectors \mathbf{v} and \mathbf{r} can obtain integer values for k -limited systems, vectors $\mathbf{d}^+(k)$ and $\mathbf{d}^-(k)$ should nevertheless be binary vectors. One can obtain a binary (0,1) value that corresponds to an integer value (0,1,2,...) by doing a double negation on integer number. Thus, the model that describes k -limited systems is structurally the same as model (8), with vectors $\mathbf{d}^+(k)$ and $\mathbf{d}^-(k)$ double negated. The model can be written as:

$$\begin{aligned}
 \mathbf{v}(k) &= \mathbf{v}(k-1) + \overline{\overline{(\uparrow_{-1} \mathbf{v}(k-1)) \cdot \mathbf{r}_v(k-1)}} - \\
 &\quad - \overline{\overline{\mathbf{v}(k-1) \cdot (\uparrow_1 \mathbf{r}_v(k-1))}} \\
 \mathbf{r}(k) &= \mathbf{r}_{cap} - (\Lambda_s)^T \Delta \mathbf{v}(k) \\
 \mathbf{r}_v(k) &= \Lambda_s \Delta \mathbf{r}(k)
 \end{aligned} \tag{9}$$

2.4 Recursive model of MRF system with more than one flowline

The models introduced so far are valid for systems with a single flowline. If the system embraces more

than one flowline, the previous models should be modified. First, to construct an MJI matrix of such system, we separate different flowlines with zero rows. This is convenient since in (F)MRF systems, at least one resource is assigned to each job, hence, zero rows cannot appear in any other place in matrix MJI. MJI matrix and vectors \mathbf{v} and \mathbf{r} are then constructed as follows:

$$\Lambda_s = \begin{bmatrix} \Lambda_{s1} \\ \mathbf{0} \\ \Lambda_{s2} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \Lambda_{sn} \end{bmatrix}, \quad \mathbf{v} = \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{0} \\ \mathbf{v}_2 \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \mathbf{v}_n \end{bmatrix}, \quad \mathbf{r} = \begin{bmatrix} \mathbf{r}_1 \\ \mathbf{0} \\ \mathbf{r}_2 \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \mathbf{r}_n \end{bmatrix}$$

If vectors \mathbf{v} and \mathbf{r} would be included in the previously given models, the results would be inaccurate due to inserted zeros. To neutralize the influence of these zeros in vectors \mathbf{v} and \mathbf{r} we introduce an auxiliary vector, \mathbf{u}_{aux} :

$$\mathbf{u}_{aux} = \Lambda_s \Delta \mathbf{I}_{m,1} \tag{10}$$

where $\mathbf{I}_{m,1}$ is a column vector with dimension m that is filled with '1's. According to equation (10), auxiliary vector \mathbf{u}_{aux} element is zero if it corresponds to zero row in MJI, otherwise it is one. If we denote with “ \vee ” element-wise logical or operation, the recursive model for systems with more than one flowline can be written as:

$$\begin{aligned}
 \mathbf{u}_{aux} &= \Lambda_s \Delta \mathbf{I}_{m,1} \\
 \mathbf{v}(k) &= \mathbf{v}(k-1) + \\
 &\quad + (\uparrow_{-1} \mathbf{v}(k-1) \vee \overline{\mathbf{u}_{aux}}) \cdot \mathbf{r}_v(k-1) - \\
 &\quad - \mathbf{v}(k-1) \cdot (\uparrow_1 \mathbf{r}_v(k-1) \vee \overline{\mathbf{u}_{aux}}) \\
 \mathbf{r}(k) &= (\Lambda_s)^T \Delta \mathbf{v}(k) \\
 \mathbf{r}_v(k) &= \Lambda_s \Delta \mathbf{r}(k)
 \end{aligned} \tag{11}$$

From equation (10) an attentive reader can conclude that auxiliary vector enables correct implementation of parts input and output jobs.

It should be noted that vector shift operation (Definition 3) need to be performed separately on each sub-vector, i.e.

$$\uparrow_m \mathbf{v} = \begin{bmatrix} \uparrow_m \mathbf{v}_1 \\ \mathbf{0} \\ \uparrow_m \mathbf{v}_2 \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \uparrow_m \mathbf{v}_n \end{bmatrix}$$

2.5 Conflict resolution in recursive model of MRF system

In case two or more jobs, assigned to a single resource, can be started in the same step k , the system designer should define which of them should be performed. This kind of situation is called a conflict. From the mathematical point of view, conflict develops when more than one element of vector \mathbf{v}^+ that corresponds to a single resource is equal to '1'. Conflict is solved, depending on the resource dispatching strategy, in such a way that jobs of lower priority are forbidden i.e. the corresponding elements of $\mathbf{d}^+(k)$ are set to zero. The mathematical approach to conflict resolution is analogous to the one described in [11].

Let us define the following:

Definition 4 (conflict jobs vector)

The *conflict jobs vector* \mathbf{v}_d comprises information on the jobs performed by shared resources. $v_{di} = 1$ if job i is done by a shared resource, otherwise $v_{di} = 0$. The vector dimension equals the total number of jobs in the system. If a shared resource vector is denoted as \mathbf{r}_s ($r_{si} = 1$ if i -th resource is shared, otherwise $r_{si} = 0$), vector \mathbf{v}_d can be determined as follows:

$$\mathbf{v}_d = \Lambda \Delta \mathbf{r}_s \quad (12)$$

Definition 5 (dispatching matrix)

The *dispatching matrix* \mathbf{F}_d is determined from the conflict jobs vector \mathbf{v}_d as follows:

$$f_{d(i,j)} = \begin{cases} 1 & v_{di} = 1 \text{ and } j = \sum_{k=1}^i u_{dk} \\ 0 & \text{otherwise} \end{cases} \quad (13)$$

Definition 6 (dispatching vector)

The dispatching vector \mathbf{u}_d is a column vector with dimension equal to the number of conflict jobs in the system. If job i is of the highest priority among conflict jobs, $u_{di} = 1$, otherwise $u_{di} = 0$. The priorities of jobs depend on the applied conflict resolution strategy.

As we said earlier, in case of the conflict, the jobs of lower priority should be forbidden, hence, corresponding element of vector $\mathbf{d}^+(k)$ should be set to zero. Recursive matrix model for the system with conflict resolution is:

$$\begin{aligned} \mathbf{v}(k) = & \mathbf{v}(k-1) + \\ & + (\uparrow_{-1} \mathbf{v}(k-1)) \cdot \mathbf{r}_v(k-1) \cdot \overline{\mathbf{F}_d \Delta \mathbf{u}_d} - \\ & - \mathbf{v}(k-1) \cdot (\uparrow_1 \mathbf{r}_v(k-1)) \end{aligned}$$

$$\begin{aligned} \mathbf{r}(k) = & \overline{(\Lambda_s)^T \Delta \mathbf{v}(k)} \\ \mathbf{r}_v(k) = & \Lambda_s \Delta \mathbf{r}(k) \end{aligned} \quad (14)$$

The model in form of (14) is suited only for 1-limited systems.

To define a conflict resolution for k -limited systems, the element $\overline{\mathbf{F}_d \Delta \mathbf{u}_d}$ needs to be included in the expression for $\mathbf{d}^+(k)$. Conflict resolution model for k -limited systems is then given as:

$$\begin{aligned} \mathbf{v}(k) = & \mathbf{v}(k-1) + \\ & + (\uparrow_{-1} \mathbf{v}(k-1)) \cdot \mathbf{r}_v(k-1) \cdot \overline{\mathbf{F}_d \Delta \mathbf{u}_d} - \\ & - \mathbf{v}(k-1) \cdot (\uparrow_1 \mathbf{r}_v(k-1)) \\ \mathbf{r}(k) = & \mathbf{r}_{cap} - (\Lambda_s)^T \Delta \mathbf{v}(k) \\ \mathbf{r}_v(k) = & \Lambda_s \Delta \mathbf{r}(k) \end{aligned} \quad (15)$$

Since vector \mathbf{r}_{cap} is filled with ones for 1-limited systems, model (15) is suitable both for 1-limited and k -limited systems with $k > 1$. If one wants to apply the same procedure for k -limited systems with more than one flowline, the auxiliary vector \mathbf{u}_{aux} should be included in model (15) as well:

$$\begin{aligned} \mathbf{u}_{aux} = & \Lambda_s \Delta \mathbf{I}_{m,1} \\ \mathbf{v}(k) = & \mathbf{v}(k-1) + \\ & + (\uparrow_{-1} \mathbf{v}(k-1) \vee \mathbf{u}_{aux}) \cdot \mathbf{r}_v(k-1) \cdot \overline{\mathbf{F}_d \Delta \mathbf{u}_d} - \\ & - \mathbf{v}(k-1) \cdot (\uparrow_1 \mathbf{r}_v(k-1) \vee \mathbf{u}_{aux}) \\ \mathbf{r}(k) = & \mathbf{r}_{cap} - (\Lambda_s)^T \Delta \mathbf{v}(k) \\ \mathbf{r}_v(k) = & \Lambda_s \Delta \mathbf{r}(k) \end{aligned} \quad (16)$$

The model (16), with properly determined associated vectors \mathbf{u}_d , \mathbf{r}_{cap} , \mathbf{u}_{aux} , comprises all previously considered system models: (8), (9), (11) and (15) and is therefore the most suitable for implementation.

2.6 Recursive model of FMRF system

As we stated earlier, the idea behind determination of the model of FMRF system is that, having a dispatching strategy, FMRF system can be represented as corresponding MRF system in each discrete event iteration step k .

The MJI matrix Λ_s , which is an MRF substitute of the FMRF system in step k , depends on the applied strategy.

Definition 7 (MRF substitute MJI matrix)

The MRF substitute MJI matrix $\Lambda_s \in \mathfrak{R}^{m \times n}$ is a matrix with dimensions equal to dimensions of FRMF system MJI matrix Λ . Λ_s is obtained as follows:

$$\Lambda_s(i, j) = \begin{cases} 0 & \Lambda(i, j) = 0 \\ 1 & \Lambda(i, j) = 1 \text{ and } \sum_{k=1}^n \Lambda(i, k) = 1 \\ u_l \in \{0,1\} & \Lambda(i, j) = 1 \text{ and } \sum_{k=1}^n \Lambda(i, k) \geq 2 \end{cases} \quad (17)$$

As we can see, the matrix Λ_s is a structural copy of matrix Λ , where each '1' in rows that contain more than one '1' is substituted with corresponding control variable u_l . If we return to the example from the beginning, matrix Λ_s is determined as a function of control variables as:

$$\Lambda = \begin{array}{c} J_1^1 \\ J_2^1 \\ J_3^1 \end{array} \left| \begin{array}{cccc} R_1 & R_2 & R_3 & R_4 \\ u_1 & u_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & u_3 & u_4 & 0 \end{array} \right|$$

Control variables have binary values depending on whether or not the corresponding resource performs the job. The values of control variables should be such that in every step k exactly one variable in the row is equal to '1'.

3 System simulator design

Based on the formulas presented in this paper the application called *MJIWorkshop*, which simulates the FRMF systems is developed. The simulator's main window is shown in Figure 1. The system is initially given by its MJI matrix and initial state and for each step k a new state value is determined. The state of the system is visually represented in form of a token matrix shown in Figure 2. The token matrix is structurally the same as the MJI matrix. The matrix element (i, j) contains a token in step k if job i is executed by resource j in step k . The token matrix is, as well as Petri nets, convenient for visualizing the discrete event systems.

Developed system simulator is standard Multiple-document Microsoft Foundation Class (MFC) application created with Microsoft Visual Studio 6.0 [12] and is written in C++. A reason to use MFC instead of .NET is an intention to get a program that is able to run without a problem even on older PCs and further, to make application portable, i.e. able to run from USB stick without need to install it. Internal structure of program is more or less similar to other multiple document programs created with Visual Studio wizard.

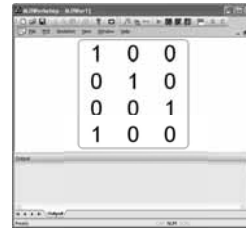


Figure 1. MJIWorkshop input window

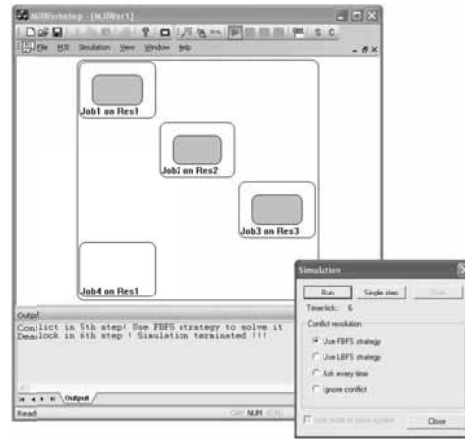


Figure 2. Visualizing system state in MJIWorkshop including conflict resolution window

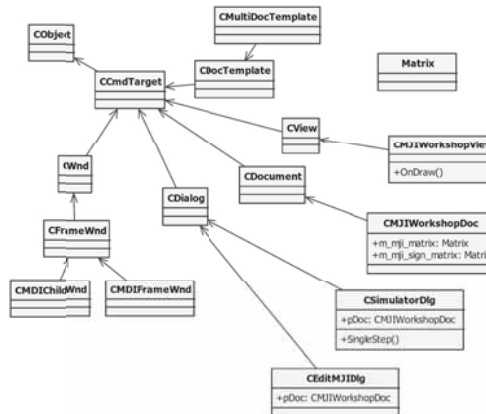


Figure 3. Simplified UML class diagram

It is standard document-view architecture software, which is mainly based on MFC classes CDocument and CView. Main class, which holds all necessary data, is CMJIWorkshopDoc.

This class, using its methods, performs editing and simulation by creating and running two other objects: CEditMJIDlg and CSimulatedlg. Class CMJIWorkshopView takes care of visualization.

Its object, using associated document, takes all necessary data to display the current system state on the screen, as shown in Figure 4. Unified Modeling Language (UML) class diagram, which shows main relationships between classes is given in Figure 3.

4 Conclusion and future work

Developed simulator is based on the model that consists of a single matrix: the machine-job incidence matrix. The MJI matrix comprises all information needed to fully model an FMRF system. Recursive models are given separately for basic types of systems, while the developed simulator encloses all of these models and implements the overall model of k -limited FMRF systems with more than one flowline, together with the job and conflict resolution. The current state of the system is obtained by performing various simple matrix operations, thus, the recursive model procedure is easy to implement and non-time-consuming. Matrix dimensions depend on the size and complexity of the analyzed system. The simulator is convenient for system analysis in design stage since it allows the user to apply various job and resource dispatching strategies on-line while simulating the system. In the future, operating times are to be included in the application together with the support for system performance analysis. Also, the connection with other simulating tools, such as Matlab and Petri.NET [16] is to be developed.



Figure 4. Signal visualization with GnuPlot

References

- [1] Viswanadham N, Narahari Y. *Performance Modeling of Automated Manufacturing Systems*. New Jersey: Prentice Hall, 1992.
- [2] Vince J. *Virtual Reality Systems*. Reading, MA: Addison-Wesley, 1995.
- [3] Mayr H. *Virtual Automation Environments – Design, Modeling, Visualisation, Simulation*. New York Basel: Marcel Dekker, 2002.
- [4] Gertz M W, Khosla P K. Onika: *A Multilevel Human-Machine Interface for Real-Time Sensor-Based Robotics Systems*, Proc. of SPACE 94: 4th Int. Conf. and Exposition on Engineering and Construction, 1994.
- [5] Nethery J, Spong M W. Robotica: A Mathematica Package for Robot Analysis, IEEE Rob. Aut. Mag. 1994; 1: 1: 13–20.
- [6] Ge S S, Lee T H, Gu D L, Woon L C. A One Stop Solution in Robotic Control System Design, IEEE Rob. Aut. Mag. 2000;7:3:42–54.
- [7] Corke P. *Robotic Toolbox for Matlab*, CSIRO Manufacturing Science and Technology, <http://www.cat.csiro.au/cmst/>, visited 2005.
- [8] Choi B, Park B, Ryu H Y. *Virtual Factory Simulator Framework For Line Prototyping*, J. of Advanced Man. Sys., 2004;3:1:5–20.
- [9] Kusiak A., Ahn J. *Intelligent Scheduling of Automated Machining Systems*, Computer-Integrated Manufacturing Systems, 1992; 5; 1: 3-14.
- [10] Steward, D.V. *The Design Structure System: A Method for Managing the Design of Complex Systems*, IEEE Transactions on Engineering Management, 1981; 28: 71-74.
- [11] Bogdan S, Lewis FL, Mireles J, Kovacic Z. *Manufacturing Systems Control Design: a matrix based approach*, Springer, 2006.
- [12] Visual Studio 6.0, [webpage] (2008, November 26) online: <http://msdn.microsoft.com/enus/library/ms950417.aspx>
- [13] GnuPlot homepage, [webpage] (2008, November 3) Available at: <http://www.gnuplot.info/>
- [14] PNML View - Cross platform viewer for PNML files, [webpage], (2008, October 15) Available online: <http://www.vanwal.nl/pnmlview/>
- [15] Matrix C++ - Template class library with full source code, [webpage], (2008, October 2) Available online: <http://www.techsoftpl.com/matrix/>
- [16] Genter G., Bogdan S., Kovacic Z., Grubisic I.: *Software tool for modeling, simulation and real-time implementation of Petri net-based supervisors*, Control Applications, 2007. CCA 2007. IEEE International Conference on , vol., no., pp.664-669, 1-3 Oct. 2007

Corresponding author: Stjepan Bogdan,
University of Zagreb, Faculty of EE&C,
Dept. of Control and Computer Engineering,
Laboratory of Robotics and Intelligent Control Systems
Unska 3, 10000, Zagreb, Croatia
stjepan.bogdan@fer.hr

Received & Accepted: MATHMOD 2009: -

Revised: September 10, 2009 -

Accepted: July 10, 2010 -

Declarative Resource Discovery in Distributed Automation Systems

Christoph Gerdes¹, Christian Kleegrewe¹, Jörg P. Müller²

¹ Siemens AG, Corporate Technology, Information & Communications, Germany

² Clausthal University of Technology, Germany

SNE Simulation Notes Europe SNE 20(3-4), 2010, 13-20, doi: 10.11128/sne.20.tn.09983

Engineering of complex automation systems is a major cost factor. One reason for the complexity of engineering processes is the tight coupling of automated functions and the actual automation device. The paper introduces a method and architecture to de-couple functions from devices. As foundation Peer-to-peer and Grid computing technologies are applied to provide a flexible framework for automated functions. A key issue is the applicability in actual industrial systems which is accounted for by designing an algorithm which can be implemented cost efficiently on resource constraint devices. The approach is evaluated through extensive simulations which demonstrate benefits and applicability of the proposed architecture.

SNE 20/3-4, December 2010

Introduction

Engineering is the major cost factor in the construction of state of the art industrial systems. Automation equipment features a multitude of parameters that need to be configured and parameterized during commissioning of a plant. Additionally, the environment is highly heterogeneous with multiple equipment vendors and product versions. Finally the recent demand from customers to build more flexible, easier to customise and fully integrated manufacturing systems adds to the complexity.

Several approaches have been proposed to cut engineering costs. One way to achieve more flexible automation system is to shift to decentralised architectures [14, 11]. However, these approaches mark a major paradigm shift in automation which is currently not supported by neither equipment vendors nor system integrators. The key problem, however, the so called point to point dilemma, i.e., the tight coupling of automated functions and resources remains un-addressed. Today systems are designed statically by directly specifying end-to-end communication. For example, sensor S delivers data to motor M. If M is replaced, or other units are interested in data generated by S, each new connection needs to be configured end-to-end. Thus an enormous conglomeration of static communication links is the result. Besides being expensive to configure, the static interweaving of automation equipment prevents flexible execution of control software.

In this paper, we present an architecture capable of de-coupling individual automation equipment.

We propose a robust and self-organising system to discover resources at runtime in a networked automation system using declarative resource discovery. We use Peer-to-Peer (P2P) and Grid computing technology originally designed for the Internet and advance them such that they can be implemented on even resource constraint equipment. P2P systems such as Chord [18] and CAN [15] provide the foundation for loosely coupled systems in the Internet. However, both their code complexity as well as stabilisation effort, i.e. the intensity of required communication, suggest cost intensive realisations and hence those approaches are rather unlikely to emerge in industrial products. We therefore investigate how a lightweight algorithm can provide similar features albeit being less capable in an Internet scale, i.e., millions of users, scenario.

The structure of this article is as follows. First we describe the related art in P2P computing and its application in automation systems. After that a system overview introduces architecture and core concepts like queries, stabilisation and grid functionality. Providing a simple application example the discovery process is illustrated. Sections with simulations and the discussion of our results conclude the paper.

1 Related work

In recent years structured P2P protocols have been extensively studied. Ratnasamy et al. suggest in [15] a multidimensional hash table for distributed data management whereas in most of the related work like [18, 2] different routing algorithms in one dimensional distributed hash tables (DHT) are suggested.

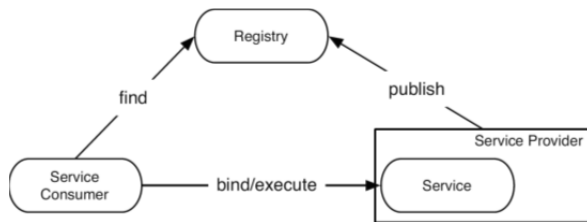


Figure 1. Actors in service oriented architectures: provider, consumer and registry

Since DHTs are not suitable for complex queries such as range queries, there is considerable research effort in enhancing such distributed data structures in a way that range queries can be managed at low costs. One major approach in this field of research is the application of prefix hash trees (PHTs) as proposed in [17]. This approach suggests that data is stored in a tree structure that grows and shrinks dynamically with the amount of data being stored. Data is sorted in the tree leaves according to a prefix matching algorithm. The prefixes of the tree nodes are then stored in a DHT. For range queries the first node that contains data is evaluated by consistent hashing and after finding this node only the subtree beyond this tree node must be searched for data. A similar approach is suggested in [12]. A different method for range queries in DHT structures is based on space filling curves, first described by Giuseppe Peano in 1890. Such search algorithms are described in [1, 8] and [21]. In [1] a range query method is introduced that is optimised for information search in grid information services. Spanning Trees to enhance routing capabilities in P2P networks are evaluated in [6] and [10]. In [10] a self-stabilising spanning tree algorithm is suggested that is capable of handling churn events like common DHT protocols for Internet scale applications.

The algorithm additionally provides optimum results in networks where no global network view can be provided. In [6] an algorithm is described that allows placing of processes in a self-healing and ordered spanning tree in which distributed object queries are routed. In the context of industrial automation systems, concrete P2P mechanisms are proposed in Drinjakovic et al. [5] who suggest lookup methods in a process control system using P2P networks. Thereby information resources are grouped according to a structured naming scheme and search methods provide deterministic access at runtime. The described query method, however, is one dimensional and thus limits queries to keyword searches.

2 Overview of our approach

We address the point to point dilemma by designing automation systems as loosely coupled orchestrations of automation assets. Similar approaches are taken in service oriented architectures (SOA) such as sensor Grids or similar Grid computing [7] applications. In SOA, assets are described by the service, i.e. the function, they provide. Services encapsulate resources which execute the functions provided by the service via a service interface which in turn provides a consistent view on the resource. Besides the service interface, a service description provides information about the interface as well as qualities of operation and management. In order to use a service, a service consumer states his interest in a service query and issues the query at a service registry which maintains all available service descriptions. Once found, the service consumer binds to the service first and then executes the desired functions (Figure 1). A key advantage of the service oriented approach are the different temporal options for service binding. Early binding is referred to at design time when requirements are mapped to service descriptions. Late binding, in contrast, renders a system more flexible as the service consumer binds the service at runtime. Ultra-late binding takes the concept one step further as applications are created by composing services dynamically for each invocation and removing services from the application after they are no longer needed.

In order to support late and ultra-late binding paradigm, the service registry must provide efficient means to process the service query and match it to service descriptions accordingly. Solutions for this service discovery process range from simple keyword searches to evaluation of fuzzy logic and semantic reasoning. While the expressiveness of keyword searches is often not sufficient for complex service descriptions, the resource demand for semantic matching prohibits cost efficient solutions. A flexible compromise is provided by complex declarative queries, common in most database systems. Query languages like the standard query language (SQL [4]) provide rich semantics to express complex interests yet they can be implemented efficiently.

While centralised registries can be implemented efficiently, they are additional infrastructure components that need to be integrated and maintained. Distributed registries, however, use already existing resources of the networked assets.

They are further more robust and scale dynamically with the number of networked assets. In the following paragraphs we describe a generic service oriented architecture for loosely coupled automation devices. Being a key component, we emphasis in the description of a distributed service registry with support for declarative queries. Figure 2 illustrates the multi-layer architecture of a service oriented automation system. As foundation, the distributed registry collects service descriptions and provides query interfaces. Networked assets are interlinked via a self-organising overlay network. The overlay acts as network virtualisation by abstracting physical network topologies and providing content based addressing schemes. A query engine parses incoming queries and generates a query execution plan. A data management component organises service descriptions and handles their storage on respective assets. Building on the abstract service descriptions, a resource virtualisation layer coordinates resources allocation. It provides a level of abstraction to establish location transparent resource access. Further it allows to aggregate several atomic resources to higher level concepts. For example a boiler is a high level concept being composed of several assets, e.g., sensors, pumps and heaters. Finally the service level combines atomic and composite resources to services and service workflows like automated functions or monitoring and analysis functions.

Before we can describe the resource discovery process we need to introduce a simplified resource model. The model is based on Chen's Entity-Relationship model [3]. It is rather generic and can be mapped to more advanced approaches like the WS-* standards. Based on this model, a domain specific query language is introduced that enables declarative resource discovery.

2.1 Resource model

The concept of a *resource* denotes any asset in a networked automation system such as sensors, actuators, as well as information entities like device states and condition. Each resource R is modelled through a finite and ordered list of features f_1, \dots, f_n . Features have a class L and a value V as well as an informal description. The set of all features of all resources span a vector space F . A resource relationship is an association among resources. For instance, a resource temperature sensor is associated to a resource controller as it delivers required measurements. A correlation of resources describes a temporary semantic similarity of resources.

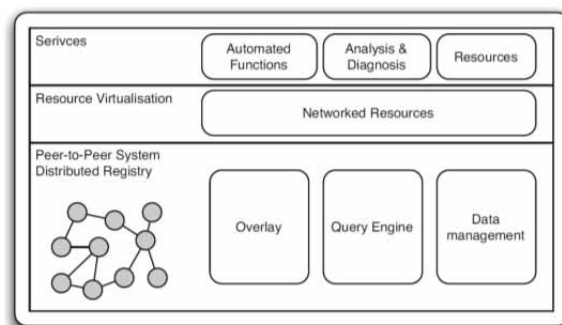


Figure 2. Automation Grid Architecture

Correlations can be quantified via a distance metric M . Hence correlated resources can be organised in a cluster structure. Put formally, resource R_i belongs to group with centroid G_i iff $\|R_i - G_i\|_M \leq r$ where r is an application specific radius. Whereas the centroid is a numerical representation of the correlation.

2.2 Peer-to-peer subsystem

As peer we understand a software component that is hosted by networked assets. Resources are managed by peers and all peers are interconnected in a P2P network. Resource correlations emerge dynamically during the lifetime of a system. There are computed continuously at individual peers by exchanging information on hosted an associated resources and applying a correlation function.

The topology of the overlay network is determined by resource relationship and correlations. Peers hosting correlated resources are more likely to be neighbours while peers with non-correlated resources are unlikely to be directly linked.

2.3 Resource discovery

Supporting existing programming models, automated services can be defined using common standards, e.g., IEC 61131, augmented with a declarative semantics for resource discovery. Hence resources are addressed not directly using physical memory or network addresses but rather declaratively, namely using queries that describe what features of the resource are required. To support query composition and execution, we developed a query language to express resource interest in an abstract form, hiding unnecessary details. A complete description of the language is beyond the scope of this paper. In the following we introduce the key features of the language by discussing a simple query statement.

```

1 SELECT * FROM RESOURCES
2 WHERE resource
3 HAS FEATURE(r=0,575,35345,767854,*)
4 TOP 1
5 WINDOW(0, FOREVER, 1s)

```

Listing 1. Example of a continuous query

The example provided in Listing 1 illustrates a simple resource query stated in our language. The language is based on the well-known and widely adopted standard query language (SQL [4]) but augmented with additional statements for resource selection. While the `SELECT ... WHERE` clause is standard, the `HAS FEATURE(tuple)`, specifying the requested resource, is a unique extension in our system.

A tuple consists of a specification of the maximum distance r between query and resource whereby $r = 0$ limits the result to only exact matches. The rest of the elements specify the features of the resource being searched for. The wildcard character '*' causes the particular feature to be excluded from the evaluation. The `TOP n` operator reduces the result set to the n resources closest to the feature specification. Queries can run either once as snapshot query or continuously over a period of time. If run once, the query returns a single result set while if running continuously, the query initiates a data stream of result sets.

The `WINDOW` operator specifies the activation interval of the data stream. The first two parameters set start and end time while the third parameter specifies the interval of execution, e.g., every 10 seconds. Once injected into the network, queries are compiled to binary form and loaded in the local query engine. An execution plan (Figure 3) is generated and scheduled for processing. The plan lists all actions necessary to deliver the requested result set. The plan depicted in Figure 3 first checks locally if the query can be evaluated with local information alone. If so query execution is complete. Otherwise the query is optimised and rewritten for distributed execution.

Query optimisation is a complex procedure where the query is restructured to reflect the overlay topology as well as states of other concurrent queries scheduled in the same query engine. Afterwards a resolving action is triggered which determines which peer might provide the requested information. This procedure is discussed in detail in Section 3. Subsequently the query as a whole or in part is assigned to the peer accordingly. Intermediate results are stored in the local storage.

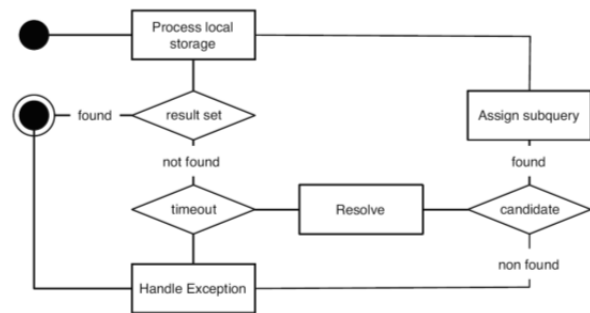


Figure 3. Query execution plan

The process continues until all requested information is contained in the local storage. In case the maximum number of retries is reached or a timeout occurred an exception is raised and the execution ends.

Continuous queries are constantly evaluated, and hence can adapt to network reconfigurations and resource dynamics. Therefore, each peer collects information on resource states and conditions from its neighbours and re-optimises the query to reflect the new network condition.

3 Routing and stabilisation

Besides query optimisation, resolving of candidates for query processing is a key element for every execution plan. Each peer stores service descriptions of the resources it hosts in its local storage. Descriptions of composite resources are hosted by all peers having associated resources. Hence the problem to address is to find a mapping from the content requested via the query to a respective peer that can provide the content. In distributed hash tables a hash function is applied which maps the query, i.e., the keyword to an identifier space which is partitioned over all peers participating the DHT.

For the distributed registry as described in section 3 a multidimensional lookup mechanism is required which locates a resource based on a set of features specified in the query. Using a DHT, this would require to maintain multiple overlay structures, e.g., a Chord ring for each feature causing considerable compute and network overhead. Multidimensional approaches such as CAN seem more suitable but are restricted by their enormous code complexity which yet let to only small experimental deployments in artificial environments. Additionally, using a uniform hash function, peers in standard DHTs are treated as equal independent of their resource capacities.

Data is stored as determined by the hash function, thereby not considering the specifics of individual peers. Hence, resource constraint devices might get easily overloaded with popular keywords, e.g., 'temperature' whereas other devices might have unused capacities. On the other hand, unstructured approaches like Gnutella v2, are able to assign different roles to peers depending on their network capabilities. However, due to their non-deterministic behaviour, i.e., stored items are not guaranteed to be found, they have not been considered in technical systems. Playing on the peculiarities of automation systems features of both DHTs and unstructured approaches can be combined to support complex query execution.

Automation systems are built for reliable operation for long periods of time. Thus, in comparison to the Internet [19], from which most P2P algorithms originate, we can assume the environment to be rather static i.e. device failure is rare compared to the time required to stabilise routing tables. Having to deal with limited dynamics an iterative approach based on *K-means* clustering, e.g., [20] is chosen to stabilise the overlay and route information between peers.

Routing and stabilisation evolves through two phases. In the first phase the system initialises, i.e., peers exchange information about their resources, features and physical network addresses. Based on this information exchange, peers cluster around certain centroids. In each cluster peers maintain a set of links to their neighbours. Based on their neighbour links local routing tables are built which are used to map features and combination of features to peers. The first phase ends when clusters and therefore routing tables stabilised leaving the system in a ready to operate state. It is important to note that any further communication between entities in the system is based on the routing information initialised during this phase. Only in the event of unit failure or reconfiguration, routing information needs to be updated. This ensures that communication can be realised very fast by conducting only lookup in local routing tables.

In the 2nd phase peers monitor their resources as well as the resources of their neighbours in the cluster. If a peer departs from a cluster it drops its routing tables and enters phase 1 to locate an alternative cluster. Communication in phase 2 is gradually reduced to a minimum. If a peer failure is detected, a notice is propagated within the cluster causing all connected peers to reevaluate their connections within the cluster.

Having provided a high level overview of the routing and stabilisation procedure we now examine the cluster formation and inter-cluster communication in further detail. At iteration $k = 0$, the network is initialised with a random set of centroids $\{C_{j,i,0}: 1 \leq j \leq K\}$ distributed over all peers. Each peer counts the number of resources associated to a centroid $\|C_{j,k}\|$. In each iteration, the i -th peer P_i picks a selection C of n peers at random from the set of all peers and exchanges its local set of centroids and the resource counts with the selected peers. Having received the centroids each peer begins to update, i.e. compute the mean, its local centroids thereby using resource counts as weights. Peers in C that have resources associated to the same centroids as P_i are stored locally in a neighbour table. This process is continued until the change of centroids between iterations is smaller than a predefined threshold. Once the change of centroids of all peers drop below the threshold, the initialisation phase terminates and each peer has its features associated to at least one centroid.

While clusters are highly connected, further links for inter-cluster communication need to be established. Therefore, for each cluster, bridge peers are determined that establish links to other clusters. Initially in the process each peer assumes it is the bridge to other clusters and forwards this information along with its distance to the target cluster to all of its neighbours. Peers process this information and select the neighbour closest to the target cluster as bridge peer. Acting as bridge, respective peers collect additionally to cluster neighbours also peers from clusters they bridge to. Consequently inter-cluster links are established as the algorithm iterates.

4 Application example

To further illustrate the workings and benefits of the declarative resource discovery mechanism, consider the temperature control system depicted in Figure 4 based on [13]. It consists of an oven, e.g., to cure products made of epoxy resin. Attached to the oven is a temperature sensor and a heater unit. We assume that both heater and sensor are intelligent units that provide some form of communication capabilities as well as a compute component to handle query processing.

A controller unit monitors the temperature readings of the sensor and sends control commands to the heater in order to maintain the required temperature in the oven.

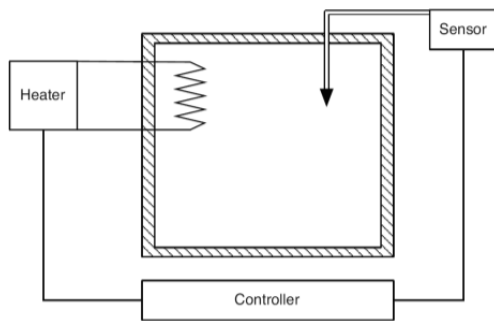


Figure 4. Simple control loop

It applies a simple PID controller to adjust the heater intensity. This simple control loop, where measurements are acquired, processed by control logic and finally control actions are executed is exemplary for a whole range of industrial control problems. Listing 2 shows the control program.

```

1 DO
2   heaters -> SELECT * FROM RESOURCES
3     WHERE resource
4       HAS FEATURE(r=0, 'heater', 'oven')
5
6   sensors -> SELECT * FROM RESOURCES
7     WHERE resource
8       HAS FEATURE(r=0, 'temperature', 'oven')
9
10  heater.temperature =
    PID(sensor.temperature, 180)
11 LOOP

```

Listing 2. Simple control program

The system is modelled with heater and sensor as atomic resources while the oven is a composite resource comprised of heater, sensor and controller (Figure 5). The oven contains two associations namely heaters and sensors which are defined as resource queries. For this simple example we assume that respective devices can be uniquely identified by the specification of only two features in the query. The relations heaters and sensors contain all sensors and heaters respectively that are currently installed. The feature valued 'oven', causes all entities to group in a single cluster. Upon query execution, the resolver iterates through the list of neighbours matching the query against the features of the neighbour peers. Subsequently full service descriptions are retrieved from the matching peers and cached locally.

The statement at line 10 sets the temperature feature of the heaters according to the control logic. In case of multiple elements in the heaters relationship all features are set accordingly.

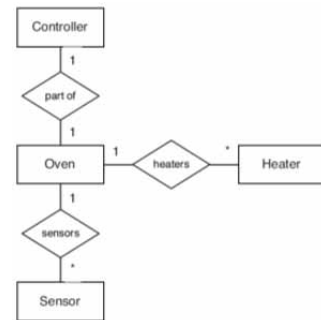


Figure 5. Simple control loop

Thereby the data management component ensures consistent propagation of the feature updates. Since the underlying system handles the resource lookup the control program will remain the same independently of where it is executed. Also if additional heaters or sensors are built into the oven, no changes are necessary for neither control program nor sensor devices.

5 Simulations

We conducted extensive simulations to demonstrate the stability of our algorithms. To assist real world applicability our simulations are based on realistic network and failure models [9]. We do not assume synchronised clocks at the peers and the simulation is capable to handle failure situations like failing peers and message loss. In order to visualise the simulation results, we simulated peers with each having one resource with two features. Both static features, i.e., having a constant value and dynamic features, i.e., a value changing over time were simulated whereas dynamic features are based on the test signals shown in Figure 6a.

Limited to static features, the algorithm showed fast convergence and stable routing tables after only a few iterations. More interestingly were simulations with dynamic features. Figure 6b shows a snapshot of a simulation with dynamic features after 30 iterations and a cluster radius of 15 units. For this simulation peers were assigned to five feature combinations: signal1:signal2, signal2:signal1, signal3:signal4, signal4:signal5, signal5:signal3.

As becomes evident in the figure, peers cluster around the five centroids and connections between clusters have been formed. There are slight deviations in the proximity to the centroids due to the unsynchronised clocks and randomised initial feature readings.

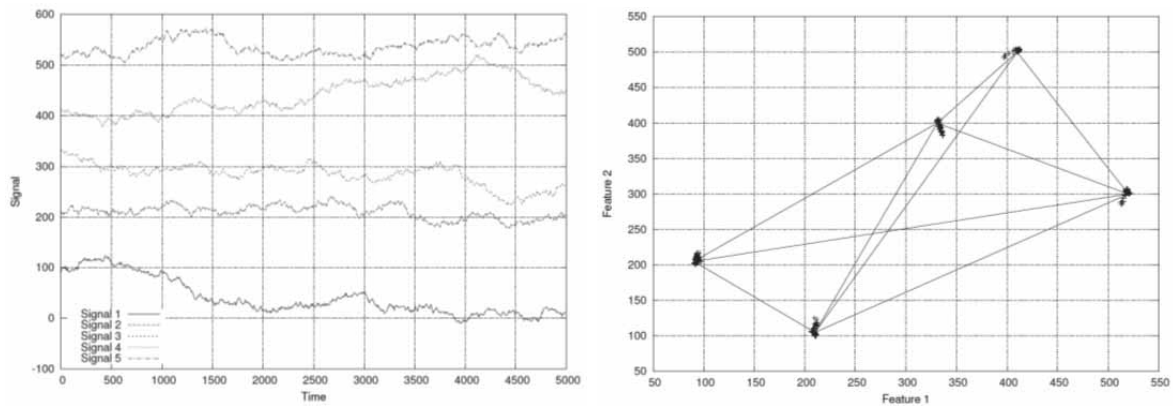


Figure 6. Simulation results for $n = 400$ peers: (a) Test signals (b) network configuration

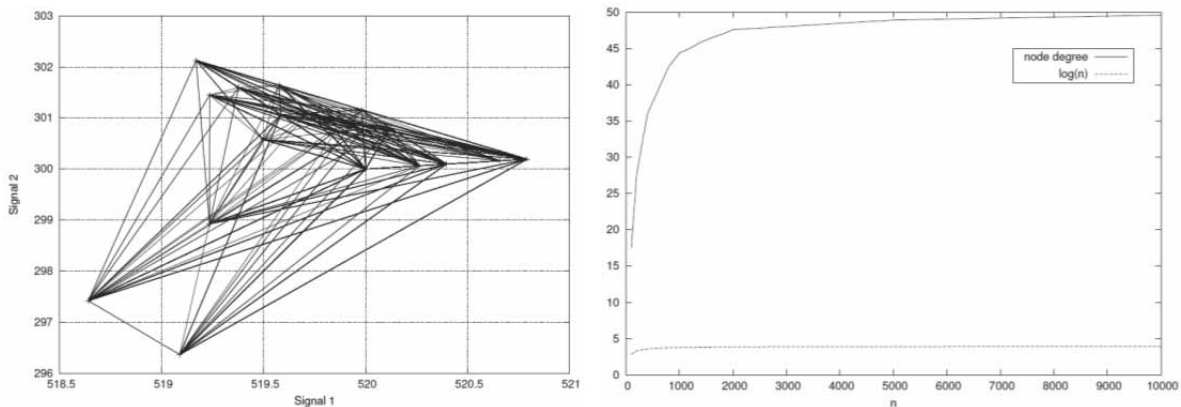


Figure 7. Simulation results from $n=100$ to 10000 peers
(a) Highly connected cluster, (b) Node degree in correspondence to the number of peers

As can be seen in the figure, the clusters have not fully stabilised with some peers located outside the cluster radius.

Figure 7a provides a closer look on a cluster after 50 iterations. The cluster is fully stabilised and peers within a cluster are highly connected. The average number of connections per peer within a cluster is 36.725 with almost equal cluster sizes of around 80 in the case for $n = 400$ peers.

Figure 7b plots the average node degree in correspondence to the total number of simulated peers. The average number of connections per peer grows larger than $\log n$ which yields, according to Erdős-Reyni, connectedness of the graph with high probability.

In general it is possible that clusters cannot stabilise since service descriptions are not correlated. In this situation, the network will be highly connected because each peer acts as bridge to all other peers.

6 Conclusion

We presented a light-weight method and architecture for declarative resource discovery in distributed automation systems. Our approach differs from other content addressable network approaches e.g. [16] in that it is less complex and hence easier to stabilise. The simplicity of our method alleviates implementation on resource constrained embedded devices.

The approach is particularly suited to the automation domain. It benefits from the static environment and the, in comparison to the Internet, small number of nodes. Clearly, using the mean as correlation function will work only for simple correlations. In more complex and dynamic scenarios other methods might be more appropriate. However, connectedness of the overlay is guaranteed due to inter-cluster connection. The k-means clustering is highly efficient for static scenarios where features describe the static capabilities of the device. Already in the static case the late-binding capabilities of the described method become effective and hence have influence on engineering complexity.

Based on the simulations conducted, we are convinced that our method is well suited for multidimensional resource lookup in modern automation systems. The declarative approach will simplify the engineering process while at the same time increase robustness and enable automation systems to react adaptively to reconfigurations and failures. The approach is not disruptive but allows smooth migration from existing systems towards the more flexible solution as devices can be gradually upgraded. Benefits become effective from the second device on.

New is the initial phase where the system collects autonomously all relevant information required for operation. Formally this process was conducted manually by an engineer or an engineering tool at design time. Once initialised the system can function as before with the addition of continuous but low bandwidth monitoring and optimisation of the overlay network.

References

- [1] A. Andrzejak and Z. Xu. *Scalable, efficient range queries for grid information services*. In: Proc. IEEE Conf. on Peer-to-Peer Computing, pages 33–40, 2002.
- [2] M. Castro, P. Druschel, Y.C. Hu, and A. Rowstron. *Proximity neighbor selection in tree-based structured peer-to-peer overlays*. Technical Report MSR-TR-2003-52, Microsoft Research, 2003.
- [3] P.P.-S.S. Chen. *The entity-relationship model: Toward a unified view of data*. ACM Transactions on Database Systems, 1(1):9–36, 1976.
- [4] C. J. Date. A critique of the SQL database language. SIGMOD Rec., 14(3):8–54, 1984.
- [5] D. Drinjakovic and U. Epple. *Search methods in p2p-networks of process control systems*. Industrial Informatics, 2004. INDIN '04. 2004 2nd IEEE Int. Conf. on, pages 101–107, June 2004.
- [6] H. Evans and P. Dickman. *Peer-to-peer programming with teaq*. Revised Papers from the NETWORKING 2002 Workshops on Web Engineering and Peer-to-Peer Computing, 2002.
- [7] I. Foster and C. Kesselman. *The Grid 2: Blueprint for a New Computing Infrastructure (The Morgan Kaufmann Series in Computer Architecture and Design)*. Morgan Kaufmann, November 2003.
- [8] P. Ganesan and H. Garcia-Molina. *Efficient queries in peer-to-peer systems*. IEEE Data Eng. Bull., 28(1):48–54, 2005.
- [9] K. P. Gummadi, S. Saroiu, and S. D. Gribble. King: estimating latency between arbitrary internet end hosts. In IMW '02: Proc. 2nd ACM SIGCOMM Workshop on Internet measurement, pages 5–18, New York, NY, USA, 2002. ACM.
- [10] Herault, Lemarinier, Peres, Pilard, and Baeuquier. *Self-stabilizing spanning tree algorithm for large scale systems*. Technical Report 1457, Laboratoire de Recherche en Informatique, August 2006.
- [11] Holonic Manufacturing Consortium. *Holonic manufacturing systems overview*. Online, 2002.
- [12] R. Huebsch, B. Chun, J. Hellerstein, B. Loo, P. Maniatis, T. Roscoe, S. Shenker, I. Stoica, and A. Yumerefendi. *The architecture of pier: an internet-scale query processor*, 2005.
- [13] R. W. Lewis. *Modelling Distributed Control Systems Using IEC 61499*. Institution of Electrical Engineers, Stevenage, UK, UK, 2001.
- [14] M. G. Mehrabi, A. G. Ulsoy, and Y. Koren. *Reconfigurable manufacturing systems: Key to future manufacturing*. Journal of Intelligent Manufacturing, 11 (4):403–419, 2000.
- [15] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In SIGCOMM '01: Proc. of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, pages 161–172, New York, NY, USA, 2001. ACM.
- [16] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. *A scalable content addressable network*. Technical Report TR-00-010, Intel Research Berkeley, New York, NY, USA, 2001.
- [17] S. Ratnasamy, J. M. Hellerstein, and S. Shenker. *Range queries over dhds sylvia ratnasamy*. Technical Report IRB-TR-03-009, Intel Research Berkeley, June 2003.
- [18] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. *Chord: A scalable peer-to-peer lookup service for internet applications*. In SIGCOMM '01: Proc. of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications, pages 149–160, New York, NY, USA, 2001. ACM.
- [19] D. Stutzbach and R. Rejaie. *Towards a better understanding of churn in peer-to-peer networks*. Technical report, Department of computer science, University of Oregon, November 2004.
- [20] R. Wolff and A. Schuster. *Association rule mining in peer-to-peer systems*. Systems, Man, and Cybernetics, Part B, IEEE Transactions on, 34(6):2426–2438, 2004.
- [21] C. Zhang, A. Krishnamurthy, and O. Y. Wang. *Brushwood: Distributed trees in peer-to-peer systems*. In In Proc. 4th Int. Workshop on Peer-to-Peer Systems (IPTPS 05), pages 47–57, 2005.

Corresponding author: Christoph Gerdes
Siemens AG, Corporate Technology
Information & Communications
Otto-Hahn-Ring 6, 81739 Munich, Germany
c.gerdes@siemens.com

Received & Accepted: MATHMOD 2009: -
Revised: September 10, 2009 -
Accepted: July 10, 2010 -

Agent Reasoning Based on Trust and Reputation

J. Samek, F. Zbořil, Brno University of Technology, Czech Republic

SNE Simulation Notes Europe SNE 20(3-4), 2010, 21-28, doi: 10.11128/sne.20.tn.09985

In the area of intelligent systems development some deterministic or nondeterministic decision algorithms and mechanisms should be used to enable agents to behave intelligently. We are trying to enhance agent reasoning and especially agent decision making with a usage of trust and reputation of particular intelligent elements (agents) as well as some social groups. There can be large agent societies, where collaboration between agents is the best way and sometime the only possibility to achieve non-trivial goals. Often it is very difficult to find *best counterparts* for collaboration. Our approach works with trust and reputation principles which are inspired from real-world societies and we try to shift them into artificial societies to make their interaction and cooperation more effective.

SNE 20/3-4, December 2010

21

Introduction

Trust is very important aspect in our everyday interaction with people, groups and institutions in our society. We should have a trust in the surrounding environment, people and institutions as well. We are often rated and judged on the basis of our trustworthiness and this defines a different manner of the interactions in our social life. We behave more openly towards subjects on account of the strong confidence and trustworthy subjects can access different types of information which can be confidential. In the case of abuse of the information, the trust of the subject rapidly decrease and it is usually very hard to restore it again.

Recent researches shows [5, 6] that system based on trust and reputation have great potentiality, for example in the e-commerce and autonomous distributed computer systems. This can be seen for example on the leading auction server *eBay*, where the selection of seller (from the buyer point of view) is based also on his or her reputation. All participants in the system are treated on the bases of his or her reputation. Trustworthiness of a seller so as of a buyer is represented by some value, which is update by the *eBay* system and depends on cumulating positive and non-positive ratings from other sellers or buyers. This reputation system, from our point of view, can be considered as relatively simple and closely aimed system.

In more sophisticated systems [3], we must deal with trust as strictly *subjective* and *context specific* metric, because it is assessed from the unique perspective of the element which has to trust somebody or somewhat and our interest is limited only to those actions (context) of a trustee that have relevance to the trust value.

In our proposal, we need to take into account many specific problems which come with *trust based reasoning*.

This paper describes preliminary proposal core for an *agent reasoning framework* based on trust and reputation principles. We proposed how a trustworthy value will *create/receive*, *store* and *represent* and *use* to agent decision. Our framework does not create next multi-agent architecture.

We are trying to build new layout based on known and well formalized bases (such as BDI [11]). This layout allow to agents to use trustworthy value to be more effectively in decision making and interacting with other agents.

The remainder of this paper is organized as follows: in Chapter 1, we describe theoretical background of trust and reputation in different disciplines of the real world; his typical characteristics and issues which are need to be take into account when are used in such context. Description of the core of our framework proposal – agent reasoning – is in Chapter 2. We go from some bases terms and notations and describe defined formulas. Finally, Chapter 3 concludes our paper, discusses open issues and our future work.

1 Trust and reputation meaning

1.1 Trust

Trust as an explicit concept is not the one that has a mutually accepted definition. We have identified the existence of trust and reputation in many disciplines of human behavior, for example: economists, sociologists and computer science [1, 2, 8].

In different areas we have different definitions as well as several different definitions in one discipline.

For our purposes, we adopt some following definition, which is used in computer science for the computation model of trust and reputation rating systems: trust is a subjective expectation an agent has about another's future behavior based on history of their encounters [1]. For our model, trust is internal rating (value) of each agent towards other agents in the system. It is based on bias or on reputation. Trust is evaluated in time, when is needed to make an agent decision, it's not persistent value in agent belief base and may vary in time.

1.2 Reputation

Reputation is an agent's mental attitude toward other agents gained during previous experiences (even indirect) with such agents. Based on trust meaning description, reputation in our model is realized as set of values which are given from past agent interaction or received *recommendation*. Reputation is stored in agent *belief* base (knowledge database or something else) when agent finished some interaction and made necessary evaluation or when agent receives some recommendation from other agent(s) in environment.

Typically, it is difficult to gain reputation from interaction in the large scale multi-agent systems. The interaction generally runs in small agent groups, where agents are close by distances or by their purposes. In the case, that these agent groups (or just each single agent from group) want to communicate with each other's is good to use recommendations. To get the best possible recommendation, we need to ask most trustworthy entity (agent) as we can. Recommendation trustworthy value and also self-trustworthy of target agent mainly depend on recommendation entity. If we trust to this entity, recommendation will be more valuable for our purposes.

There are many approaches and mechanism to ensure trustworthy entities in system. We can use PKI [10] – certification authorities and root authorities as we know from security of information systems. Toward to our approaches, it is more applicable to use *web of trust* [9] between agents and groups.

It allows us to use system more distributive without central entities – possibly points of failure. This web of trust is also more closely to the real word principles and is suitable to the agent and multi-agent systems principles.

1.3 Recommendation

The reputation value usually depends on recommendations. In recommendation process always participate three agents: the querying agent a_q , answering agent a_r (recommender) and the target of recommendation agent a_t . In the recommendation case, agent get indirectly trust value from recommendation agent to target agent [3]. This given recommendation value can be accepted as the agent's trust value to the target agent at or serves just for updating of the trust value previously counted. This recounting trust value depends on many aspects, also mainly on how trustworthy a recommender agent is.

1.4 Context and individualization of trust

There are many aspects, which comes with reasoning based on trust and reputation. These aspects are need to be take into account and will be described in this subsection. The primary aspect which is closely connected with terms such as trust and reputation is *subjective reception* and *individualization*. In a real word, each of us trust in such degree to our friends. This trust degree is based on his outer behavior but also is mainly depend on our internal “metrics”, which we using to measure his trustworthy. This metrics are strictly individual for each of us.

Typical example is human quality “prejudice” – without knowing about something man X , based on his visage (for example) we make opinion to his trustworthy. Someone, who also does not know X , makes another opinion, which can be absolutely different from our opinion. The same visage, the same man, the same knowledge about him may mean different trustworthy into him.

This is just simple example to demonstrate that trust is strictly subjective and mainly depends on our internal evaluating our perceptions for each entity (human, agent). This perception and internal evaluating may vary in time – it depends on ability of evaluating entity: *learning in time* based on previous *experiences*. In different cases, the perceptions may by for all entities the same (each agent have same sensors) but internal evaluating are different.

Perception is represented into internal agent mental state and based on agent knowledge is differently interpreted – in this case, we call it as *agent personality*.

Another very important aspect is that trust and reputation are both *context dependent* [1, 7].

It means that trust and reputation are not one-dimensional values – they are at least two-dimensional.

We must say in which context the entity is trustworthy, if we talk about entity trustworthiness. We can't simply say: "he is trustworthy" or not. He or she must be trustworthy in some context – in some quality.

Context may be for example: "can cook" or "economic advice". If we need advice in some economic problem, we ask someone who is trustworthy in context "economic advice", because advice from someone who is trustworthy in "can cook" in our economic problem may not be fine. In the next case, one entity may be in some context trustworthy and in another not.

For example: if our friend Bob is a doctor, then he is trustworthy in the context "can save our life", but if we need to cook apple pie, we will go for someone who is trustworthy in the context "can cook". So, Bob is trustworthy as a doctor, but he is untrustworthy as a chef.

With this context aspect many other problems and open issues come. At first, if we would like to evaluate some experiences after an interaction, we need to decide in which context or contexts the interaction was done. Based on this decision, we may update our belief base and finally we can do interaction evaluations.

From one interaction different reputation value in different contexts may be obtained. Another important but implementation difficult aspect is *reputation transference* – transference of one's reputation from one context to another [2]. For example: when we know that Bob is trustworthy as doctor, does it mean that Bob is trustworthy as chef or not trustworthy as chef – is this decidable?

This problem may be decided on the bases of context similarity – we need to find algorithm which is able to compare two different context (context is composed from attributes – will be described in the section 3.3) and decide *similarity degree* between them.

Similarity degree allows us to decide if the transfer from one context to another is possible. This transference problem is quite complex problem and is outside the scope of this paper.

Two cases of trusts and reputation contexts in the system are possible [1]:

1. *Uniform context*. In the uniform context environment, we rate all the agents in the same context (every agent is related to the same subject matter). For example, we have a set of agents providing email service which have related attributes, so we can rate every agent in this service context. We omit all others context in this simple mail service system and we do not define context for reputation because it is known and only one.
2. *Multiple contexts*. In the second case, we have multiple context environments. In the multiple contexts environment, any agent's reputation is clearly context dependent. We need to take into account similarities and differences among the contexts. Transference of one's reputation from one context to another may be used.

In our framework proposal, we use multiple contexts environment, which is most suitable for distributive multi agent systems and reflect the real world principles.

2 Framework for agent reasoning

Before we start to formalize our framework core components, we need to show from which phases *the reputation is built and trust evaluating process* is composed. It allows us to understand following formal notation and the used principles.

2.1 Reputation building and trust evaluating

If we want to make decision based on trust value, we need to do some steps. First of all, we need to do some monitoring of trustee performance – *monitoring phase*.

Based on this, we make some experiences with trustee or we gather some information about him or her from the reputation. Asking for a reputation of trustee is used, when direct monitoring – experience of an agent is not possible.

From the phase of monitoring of an agent's performance we need to interpret some facts, store them into some belief base (knowledge base) and then we make decision if this experience was good, bad or neutral.

This phase is called *interpretation phase*. Recommendation process, when another agent (recommender) gives us some information about trustee is also kind of experience and they also need to be stored in agent's belief base.

The experiences in the belief base needs to be stored with *time stamps*. This means that every interaction or recommendation stored in the belief base will be dated with unique (actual) time stamp. This is useful to ensure that negative or positive experience gained long time ago will have not the same impact as *fresh experience*.

After the interpretation phase, the trust value *evaluation phase* can start. Given set of experiences in a time allow us to use trust update algorithm which update agent trust value in a context. This algorithm has many different inputs – such as agent mental states, agent individual preferences, environment specific preferences and so on. There is out of scope of this paper to describe trust evaluating process, this will be our task for future work.

From all the previous phases, final ensured trust value can be used as one of many input parameter for agent decision making. If the agent's decision will be evaluated as satisfying or not, agent can increase or decrease weight function based on trust value parameter in the future decision making process.

2.2 Trust and reputation value representing

In some models [2, 4] the trust/reputation value is represented as a binary value r , typically $r \in \{0,1\}$, it means $r \in \{\text{untrustworthy}, \text{trustworthy}\}$. In our framework, we would like to express such kind of partial trustworthy or partial untrustworthy for modeling trust and recommendations effects closely.

Toward this, we define trust value as natural number in an interval $r \in (x, y)$, where x represent the worst possible rating and y represent the best possible rating of agent's trustworthy.

It is not important if $x = 0$ and $y = 100$ or $x = -100$, $y = 100$. Decisions about this interval will be implementation specific. However it is important to ensure that the trust value must change from x to y with difference Δr , which respect to model requirements and trust evaluating manners of the agent system.

2.3 Framework notation

Basis entity of each agent system is an agent. We define set of agent A as set of all possible agents in the system:

$$A = \{a_1, a_2, a_3, \dots, a_n\}.$$

To store reputation or incoming recommendation into the belief base and to make trust evaluation process it is need to determine context in which the reputation or recommendation was done.

Toward this, we need to define context. In our proposal the context definition is based on the terms *attribute* and *attribute domain*. Attribute domain means possibly range of attribute. So, we define set of all possible attribute domains Ω , when each element from this set is a domain:

$$\Omega = \{\omega_1, \omega_2, \omega_3, \dots, \omega_m\}.$$

One domain ω_1 may be for example set of *natural numbers* $\omega_1 = \mathbb{N}$, next domain ω_2 for example set of real numbers $\omega_2 = \mathbb{R}$, boolean type $\omega_3 = \{0, 1\}$ or set of named constants (enumerated type) $\omega_m = \{\text{large}, \text{big}, \text{huge}\}$, etc. Finally, we define set of all attributes B , where each attribute from this set is always projected to such domain:

$$B = \{b_1, b_2, b_3, \dots, b_k\},$$

$$\forall b \in B \exists \omega \in \Omega: \text{dom}(b) = \omega$$

Attribute b_1 may be for example Intelligence Quotient value (IQ). We can define domain ω_1 for this attribute as set of natural numbers in range from 0 to 200: $\omega_1 = (0, 200)$ a $\omega_1 \subset \mathbb{N}$. The tuple – attribute and his domain – can be written as $\langle \text{IQ}, \omega_1 \rangle$.

Example of attribute sex (as an example of another attribute b_2) based on named constants domain: $\text{sex} \in \omega_2, \omega_2 \in \{\text{male}, \text{female}\}$.

At this moment, we can use previous definition to define the term *context*. We can theoretically define context as a set of tuples: *attribute* \times *value*, where *value* is element from the *attribute domain*.

For example, context “intelligent male” or “intelligent female” may be defined as follows:

- “intelligent male” = $\{(IQ, 100), (\text{sex}, \text{male})\}$
- “intelligent female” = $\{(IQ, 100), (\text{sex}, \text{female})\}$

But in this context definition, there is problem to express some kind of *inequality*. In the previous example we can see that “intelligent male” is only the male who has exactly the same IQ as number 100. Actually, every male who have IQ equal or greater than 100 may be “intelligent male”. Toward this, we need to add new element into context definition, this element will define range of values which attribute can take. This element represents an *operator* and we define set \ddagger as set of all basic operators:

$$\Theta = \{=, >, <, \leq, \geq, <>\}$$

These operators have meaning of usual relation operators. Theirs application to the domains ω of some attribute creates range of values, which is a subset of ω . For each attribute domain $\omega \in \Omega$ it is necessary to define a function, which makes a mapping for each operator and some parameters to a subset of the original domain.

When the usual mathematical sets and the usual operators are used the evaluation is simple: in the domain ω_1 for attribute IQ from the previous example the result of application (IQ, \geq , 100) is the range: (100, 200).

For other cases, where attribute domain is for example an enumerated type or other special domains, they should be evaluated by a function defined explicitly. Result of application (sex, $>$, female) is undefined without special function, which defines the result of these comparison. In the other cases application on the same domain is transparent: (sex, $=$, male) results {male}; there is no need for a comparison function definition.

Finally, we can define context as set of triples: attribute \times operator \times value (from attribute domain); and set of all context C as follows:

$$C = \{c_1, c_2, c_3, \dots, c_l\}$$

$$\forall c \in C: c \subset B \times \Theta \times \Omega_B$$

From all the previous definitions, we provide basic terms definitions toward our notation: *trust*, *reputation* and *recommendation*. Trust in our proposal is defined as a function T . Result of this function is actual trustworthy value from some unified domain $\omega \in \Omega$ (described in Section 2.2) in such context $c \in C$ into another agent $a \in A$.

$$T: A \times C \rightarrow \Omega$$

As we say in Section 2.1, we need to ensure that recommendation and reputation will be marked with some timestamp, which allow us to use more relevant information in the belief base. Timestamp help us to determine freshness of this information. At this point, we define time set TS as set of all time units in which interaction updates belief base was done.

$$TS = \{ts_1, ts_2, ts_3, \dots, ts_x\}$$

In a *recommendation* function, we need to implement source of recommendation (recommender agent) and target of recommendation (target agent). As we say above, it is very useful to know when the recommendation was done. So we can define recommendation function E which maps agents, context and time moment to a value from an attribute domain.

$$E = A \times A \times C \times TS \rightarrow \Omega$$

And finally, *reputation* function R is defined as a mapping to a gained value on some unified domain from a target agent in such context in time – it is defined as follows:

$$R = A \times C \times TS \rightarrow \Omega$$

2.4 Agent belief base

We provided formal bases of our framework in the previous chapter. This chapter extends these bases and define how information is interpreted and stored in agent’s belief base. This description is provided from the point of view of an evaluated agent. In our framework, we recognize three sources of information to evaluate trustworthy. These sources are:

1. *Recommendation* – information about an agent trustworthy in a context, this is obtained indirectly from another agent.
2. *Reputation* – information about an agent trustworthy in a context, this is obtained directly from own experience with her or him; or this is obtained from observing or premises.
3. *Facts* – information about an agent attributes – qualities.

Last mentioned sources are the facts. Facts about agents are created and updated in time and they are based on some received recommendations or they are based on reputation.

We define fact with a function k , where inputs are an agent $a \in A$ and attribute $b \in B$. Result of this function is a value from attribute b domain and an operator $\theta \in \Theta$.

$$k: A \times B \rightarrow \theta \times \Omega_B$$

For example the fact about agent a_1 (in respect to example from the previous chapter where attribute is IQ and his domain is in the range $\langle 0, 200 \rangle$) write the following: $k(a_1, IQ) = (=, 100)$ – which means: we know, that agent a_1 has attribute (quality) IQ and this attribute is equal to the value 100 (from attribute domain $\langle 0, 200 \rangle$).

Retrieving and maintaining the facts about other agents are needed for inferencing another attributes and for building another reputation in such context based on the inferred attributes. If we know that context c is composed from some set of attributes and we have no direct experience in the context c , we can build default trustworthy from the known attributes obtained from other contexts. This inferencing deals with *reputation transference* – described in Section 1.4.

At this moment we can provide simple example of attribute inference from some reputation:

- Let the context c “intelligent male” be defined as: $c = \{IQ, >, 100\}, \{sex, =, male\}$,
- reputation of agent a in a context c “intelligent male” is 100, which means (in a unified reputation domain) maximal trustworthy,
- we can infer from this reputation two facts:
 - $k(a, IQ) = (>, 100)$,
 - $k(a, sex) = (=, male)$,
- let context c_2 “male” be defined as: $c_2 = \{sex, =, male\}$,
- let context c_3 “intelligent” is defined: $c_3 = \{IQ, >, 100\}$,
- we can infer reputation from the facts for a in context c_2 and c_3 without direct experience or without given recommendation in these context:
 - $R(a, c_2, time_x) = 100$,
 - $R(a, c_3, time_x) = 100$.

This very simple example of inferencing and reputation transference shows, that it is possible to infer reputation from the facts, respectively infer facts from the reputation. In some complicated cases, similarity degree must be used to decide which attributes can be inferred and which cannot be inferred.

2.5 Trust evaluation

Based on definitions mentioned in the previous subsection, we propose in less formally way the trust evaluation algorithm. In this evaluation process we must combine reputation history with recommendations. Results of this evaluation are used for agent decision making about with whom it is good to cooperate and with whom it is not good.

After each interaction or received recommendation, the agents can make an evaluation and update their belief bases. On the bases of such evaluations the trust value of their counterparts is updated. Evaluation mainly depends on the reputations and facts. In a case when no interaction has been made in the past and no reputation value has been set, the agent uses some default politics to bind initial trust value into some “default value”. There are many default politics to bind default trust value, for example:

- “paranoid” – the agent never trusts anyone until he or she prove his or her own trustworthy fairly,
- “neutral” – the agent takes a neutral position, it is capable to cooperate on the bases of positive recommendation,
- “friendly” – the agent is open to cooperate with anyone without previous experience.

This default politics may vary in time for each agent. In typically cases when an agent is new in an agent system, he is “friendly” and he is trying to make more friends. After time, when he was well profiled in the system and is trustworthy in his perimeter, it may change our politics to “neutral” or “paranoid” for example.

Building agent interaction history (reputations set) can be called to be *learning* process. Generally, agent increase trust to another agent, if he or she evaluates interaction as “satisfying” [7]. In “not satisfying” case, agent decrease the trust value.

During the agent learning process, if the decision of interaction (cooperate/defect) is based on other agent recommendations, the agent will also update its trust after any agent gives a recommendations.

For example, if *Alice* recommend to *Carol* that *Bob* is very good auto mechanic and *Carol* decide to go to *Bob* for her car repair, then *Carol* update trust into *Alice* also in such context as “recommendation” if will be satisfied (or not) with *Bob* service.

2.6 Agent decision based on trust

There are a lot of input parameters which can enter agent decision procedure, and the trust value can be one of them. In our agent system, we suppose that trust value is one of the main input parameter. We propose the *decision function*, which uses agent belief base – facts, reputation and recommendations – and maps it in a simple case to a binary value: cooperate/refuse (true/false, +/-). This value enters the decision procedure as a recommendation parameter to interact or not.

There are many variants of *decision functions* value types (ranges); they can be defined also as domain of attributes. For example, in a sophisticated case, the return value can be defined on interval $(-2,2)$, which may mean:

- -2: strong recommendation – do not interact,
- -1: light recommendation – you should not interact
- 0: no recommendation (unable to evaluate recommendation or neutral position),
- 1: light recommendation – you should interact,
- 2: strong recommendation – do interaction!

Internal evaluation mechanism of *decision functions* can be generally describes as follow. At first, agent must estimate some *threshold* value which is compared to trust domain range and defines delimiter for assignation return value.

If the internal trust value into agent in such context was higher or equal, agent decide to return “+”, otherwise “-” (depends on return value domain). For example, we estimate *threshold* to 80 and our *trust* to an agent is 90: the resulting value was then “+” (for a simple case) or “2” (for a sophisticated case).

Estimate function for threshold value differs due to agent mental state and many other aspects. To define threshold as a constant (for example 0.5) is a simple way to implement the estimate function. More sophisticated algorithm may use history of interactions: for example pair “*cooperate*” decision with “*non-satisfied*” results of interaction and update threshold value toward this. It is out of scope of this paper to define all implementing variants for estimate function.

2.7 After decision belief base update

If an agent decide to *interact* and it is based on *trust decision functions*, the feedback from interaction (agent was satisfied or not) update agent belief base. Agent updates our interaction history and may update trust to recommenders when interaction was made based on recommendation. We combine interaction history with feedback value to provide probability of next successful interaction in such context.

Updating reputations into each recommender after every interaction which was made on the recommendation based is also complex problem. We need to deal with feedback value, given reputation value and interaction history in the context “recommendation” for each of the recommenders.

This recommender rating is also very important for building set of agents, which are good in the recommendation context and which are not. This learning process allows us to be more effectively in time.

3 Conclusion and future work

In this paper we present preliminary framework proposal for multiple context model of trust and reputation which may allow agent reasoning based on trust. We describe critical common trust and reputation problems which are needed to be taken into account in solving reasoning problem based on trust principles. This proposal is based on known interaction protocols for the most used agent architectures such as BDI. Agents build their belief base: stores interactions history retrieves recommendations and infer facts and infers decisions.

Our model makes explicit difference between trust and reputation. We define reputation as a quantity inferred from interactions which can be highly relative toward to evaluating agent mental state and the interaction history.

We define trust as agents (trustor) internal quantity toward to trustee in a context. It can be inferred from facts or from reputation and recommendations about the trustee. It always represents strictly individual metrics. We show that trust and reputation ratings should be context and individual dependent quantities.

The framework notation, which was presented, allows us to simulate our proposal in future work. We will concentrate on formalization of the trust evaluating process before we simulate the system model. Also there are still a lot of works on formalization context transference process and context inference from agent attributes facts.

These tasks are very complex problems and must be well mapped to provide more effectively trust *decision function*, which is a core of our framework.

Acknowledgement

The research has been supported by the project MSM0021630528 and grants GP 102/07/P431 and GACR 102/09/H042.r

References

- [1] L. Mui. *Computational Models of Trust and Reputation: Agents, Evolutionary Games, and Social Networks*. PhD thesis, Massachusetts Institute of Technology, 2003.
- [2] L. Mui, M. Mohtashemi, and A. Halberstadt. *A computation model of trust and reputation*. In Proc. 35th Annual Hawaii International Conference on System Sciences (HICSS'02), volume 7, page 188, 2002. Casti, J. L.: *Reality Rules – Picturing the World in Mathematics: I, II*. Wiley, New York, 1992.
- [3] J. Samek and F. Zboril. *Multiple Context Model for Trust and Reputation Evaluating in Multi-Agent Systems*, In: Proc. of CSE 2008, Košice, SK, 2008, pages 336-343, ISBN 978-80-8086-092-9.
- [4] S. Sen and N. Sajja. *Robustness of reputation-based trust: Boolean case*. In AAMAS '02: Proc. 1st Int. joint conference on Autonomous agents and multiagent systems, pages 288–293, 2002.
- [5] P. Resnick and R. Zeckhauser. *Trust among strangers in internet transactions: Empirical analysis of Ebay's reputation system*. In NBER Workshop on Empirical Studies of Electronic Commerce, 2000.
- [6] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman. *Reputation systems*. Commun. ACM 43, 12 (Dec. 2000), 45-48
- [7] Y. Wang and J. Vassileva. *Trust and reputation model in peer-to-peer networks*. In P2P '03: Proc. 3rd Int. Conf. on Peer-to-Peer Computing, page 150, 2003.
- [8] A. Abdul-Rahman, S. Hailes. *Using recommendations for managing trust in distributed systems*. In: Proc. IEEE Malaysia Int'l Conf. on Communication 1997.
- [9] Abdul-Rahman. *The PGP Trust Model*. EDI-Forum, April 1997.
- [10] Wikipedia. *Public key infrastructure — Wikipedia, the free encyclopedia*, 2008. http://en.wikipedia.org/wiki/Public_Key_Infrastructure, accessed 24-11-2008.
- [11] M. Wooldridge. *Reasoning About Rational Agents*. The MIT Press. 2000. ISBN 0-262-23213-8.

Corresponding Author: J. Samek,
Brno University of Technology,
Department of Intelligent Systems,
Božetěchova 2, 612 66 Brno, Czech Republic;
samejan@fit.vutbr.cz

Received & Accepted: MATHMOD 2009 -

Revised: September 10, 2009 -

Accepted: June 10, 2010 -

Stochastic Models for Intermittent Demands Forecasting and Stock control

W. Sandmann, O. Bober, University of Bamberg, Germany

SNE Simulation Notes Europe SNE 20(3-4), 2010, 29-36, doi: 10.11128/sne.20.tn.09987

Demand forecasting with regard to stock control is a central issue of inventory management. Serious difficulties arise for intermittent demands, that is, if there are slow-moving items demanded only sporadically. Prevalent methods then usually perform poorly as they do not properly take the stochastic nature of intermittent demand patterns into account. They often rely on theoretically unfounded heuristic assumptions and apply inappropriate deterministic smoothing techniques. We overcome these weaknesses by means of systematically built and validated stochastic models that properly fit to real (industrial) data. Initially, no assumptions are made but statistical methods are invoked for model fitting. Reasonable model classes are found by summary statistics and correlation analysis. Specific models are obtained by parameter estimation and validated by goodness-of-fit tests. Finally, based on the stochastic models, stock control strategies are proposed to facilitate service levels guarantees in terms of probability bounds for being out of stock.

Introduction

Any organization or company that offers, sells and delivers items to others has to take care about proper inventory management. Success and efficiency substantially depend on the ability to provide and deliver demanded items within reasonable time. Stock control is crucial and the inventory policy manages how many units of an item must be in stock subject to certain constraints. Inventory capacities are limited and inventory costs should be as low as possible but at the same time a desired level of item availability should be assured.

Typically, in order to be well prepared, stock control relies on forecasting future demands by means of time series analysis based on past demand patterns. Comprehensive treatments of time series analysis and forecasting can be found in, e.g., [1, 2, 4, 5, 8]. For the broader scope of inventory management we refer the reader to [14, 18, 19].

In practice, the most common forecasting technique is simple exponential smoothing (SES), that is forecasts are made by means of a weighted sum of past observations in that based on given time series data x_1, \dots, x_t a forecast \hat{x}_{t+1} for the next data point x_{t+1} is computed recursively by $\hat{x}_1 = x_1$ and $\hat{x}_{t+1} = \alpha x_t + (1 - \alpha)\hat{x}_t$ for $t > 0$ where $\alpha \in (0,1)$ is a smoothing constant that needs to be chosen appropriately. Unfortunately, SES does not provide satisfactory forecasts for intermittent demands, i.e. in the case of so-called slow-moving items or low-demand items that are only demanded sporadically.

Instead, Croston's method [6] is most widely applied to intermittent demands.

Croston separates the time intervals between successive demands (interdemand times) and the number of units that are demanded (demand sizes). He argues that the time periods (measured in days, weeks, or months) between successive demands as well as the demand sizes are independent and identically distributed (iid) random variables, which means that intermittent demands essentially appear at random without identifiable trends, seasonality, or the like. He heuristically assumes the geometric distribution for the interdemand times and the normal distribution for the demand sizes.

If a demand occurs, separate forecasts for both the interdemand time and the demand size are updated according to SES using the same smoothing constant for both forecasts and the current demand per period forecast is obtained by the ratio of these two forecasts. Obviously, the critical issue of choosing an appropriate smoothing constant remains open.

A couple of drawbacks such as biased forecasts or potential violations of the independence assumptions have been reported in the literature and many corrections and modifications, respectively, have been proposed, e.g. [3, 11, 12, 15, 16, 17]. However, though specifically targeted to intermittent demands and often more accurate than SES, in some cases Croston's method and its various modifications do not provide proper forecasts and not even outperform pure SES.

After all, there is still no satisfactory approach to deal with stock control for slow-moving items based on forecasting intermittent demands. We argue that the problems essentially stem from the application of deterministic smoothing techniques to random patterns. Stochastic models appear to be more appropriate and promising for tackling the peculiarities of intermittent demands forecasting and stock control.

1 Stochastic modeling approach

In order to improve forecasting and stock control for slow-moving items we first have to figure out the weaknesses of existing methods and the requirements for overcoming these weaknesses. The mixture of assuming stochastic behavior and applying SES as a forecasting technique is inappropriate. The modifications of Croston's original version, adopting the independence assumptions as well as the assumption of geometrically distributed interdemand times, are mainly concerned with nonnormal (but still continuous) distributed demand sizes and modified forecasts. In particular, they still produce deterministic point forecasts though the demand pattern is essentially random. Mathematically, they work with deterministic realizations rather than with stochastic processes which are supposed to be the data generating mechanism. It has been recently shown by Shenstone & Hyndman [13] that the application of these deterministic forecasting techniques cannot be consistent with stochastic models, because any underlying stochastic model must be non-stationary and defined on a continuous sample space with negative values, which both does not match to the real properties of intermittent demand patterns.

We believe that starting with a forecasting technique and building an according underlying model is exactly the reversed order of what is required. In particular, the major problem lies in the inappropriate forecasting technique rather than in approaching intermittent demands via stochastic models. Additionally, we point out that in the previously cited literature specific probability distributions are heuristically assumed and – if at all – checked against artificial simulated data. It is often just a matter of luck whether or not the assumptions well fit to real intermittent demand data. Consequently, we argue that one should first build an adequate stochastic model based on real data, then validate its goodness of fit and finally derive forecasts and stock control strategies to meet certain requirements such as service level guarantees.

1.1 Stochastic Time Series Models

A time series is an ordered sequence x_1, \dots, x_t , interpreted as a realization of a stochastic process $(X_t)_{t \in T}$. As we are concerned with discrete time points (months), we shall assume that the index set T is a subset of the nonnegative integers. Note that, though often neglected in the literature, there is an important difference between a time series and its "generating" stochastic process. Other than a stochastic process, which is an ensemble of time series, a single time series is just one sequence of deterministic data.

Time series properties are characterized by corresponding properties of stochastic processes. We briefly present those that are most important with regard to stochastic time series models.

Definition 1 (moment functions of stochastic processes)

For a stochastic process $(X_t)_{t \in T}$ its

- *mean function* is defined by $\mu(t) := E[X_t], t \in T$
- *variance function* is defined by $\sigma^2(t) := \text{Var}[X_t], t \in T$
- *autocovariance function* is defined by $\gamma(s, t) := \text{Cov}(X_s, X_t) = E[(X_s - \mu(s))(X_t - \mu(t))], s, t \in T$

Note that $\gamma(t, t) = \sigma^2$ for all $t \in T$.

Definition 2 (strict stationarity)

A stochastic process $(X_t)_{t \in T}$ is called *strictly stationary* if its finite dimensional distributions are time invariant. That is, for all n, t_1, \dots, t_n, s the random vectors $(X_{t_1}, X_{t_2}, \dots, X_{t_n})$ and $(X_{t_1+s}, X_{t_2+s}, \dots, X_{t_n+s})$ have the same distribution.

Definition 3 (weak stationarity)

A stochastic process $(X_t)_{t \in T}$ is called *weakly stationary* if it has constant mean and variance function and its autocovariance function does not depend on specific time points but only on the time difference, the so-called lag τ .

That is, for all $s, t \in T, \tau := t - s$:

$$\begin{aligned} \mu(t) &= \mu < \infty, \sigma^2(t) = \sigma^2 < \infty, \\ \gamma(s, t) &= \gamma(s + \tau, t + \tau) \end{aligned}$$

Then the autocovariance function for lag τ is defined as $\gamma(\tau) = \gamma(t - s) := \gamma(s, t)$.

It follows for the autocovariance function of a weakly stationary stochastic process, that for all τ :

$$\gamma(0) = \sigma^2, \gamma(\tau) = \gamma(-\tau), |\gamma(\tau)| \leq \gamma(0)$$

Definition 4 (autocorrelation function)

The *autocorrelation function* of a weakly stationary stochastic process $(X_t)_{t \in T}$ is defined by

$$\rho(\tau) := \gamma(\tau)/\gamma(0) = \gamma(\tau)/\sigma^2$$

It follows for the autocorrelation function of a weakly stationary stochastic process, that for all τ :

$$\rho(0) = 1, \rho(\tau) = \rho(-\tau), |\rho(\tau)| \leq 1$$

Now, being equipped with the most important properties of stochastic processes, we introduce the most important stochastic processes with regard to time series analysis and in particular with regard to our modeling approach.

Definition 5 (White noise)

A *white noise* is a sequence $(Z_t)_{t \in T}$ of independent and identically distributed (iid) random variables. If all these random variables are normally distributed with expectation $\mu = 0$ and variance $\sigma_2 < \infty$, then $(Z_t)_{t \in T}$ is called a *Gaussian white noise*.

Obviously, a white noise is a strictly stationary stochastic process. In time series analysis, white noise is used for constructing more complex stochastic processes. In the following, let $(Z_t)_{t \in T}$ denote a white noise with expectation $\mu = 0$ and variance $\sigma_2 < \infty$.

Definition 6 (Autoregressive process)

An *autoregressive process* of order p , denoted by $AR[p]$, is a stochastic process $(X_t)_{t \in T}$ defined by

$$X_t = \alpha_1 X_{t-1} + \dots + \alpha_p X_{t-p} + Z_t, \quad t \in T$$

where $\alpha_1, \dots, \alpha_p$ are constant coefficients.

Definition 7 (Moving average process)

A *moving average process* of order q , denoted by $MA[q]$, is a stochastic process $(X_t)_{t \in T}$ defined by

$$X_t = \beta_0 Z_t + \beta_1 Z_{t-1} + \dots + \beta_q Z_{t-q}, \quad t \in T$$

where β_0, \dots, β_q are constant coefficients. The random variables $Z_t, t \in T$ constituting the underlying white noise are usually normalized such that $\beta_0 = 1$.

Definition 8 (ARMA process)

An *autoregressive moving average process* of order (p, q) , denoted by $ARMA[p, q]$, is a stochastic process $(X_t)_{t \in T}$ defined by

$$X_t = \alpha_1 X_{t-1} + \dots + \alpha_p X_{t-p} + Z_t + \beta_1 Z_{t-1} + \dots + \beta_q Z_{t-q}, \quad t \in T$$

where $\alpha_1, \dots, \alpha_p$ and β_0, \dots, β_q are constant coefficients.

Hence, ARMA processes are composed of AR processes and MA processes and both AR processes and MA processes are specific ARMA processes. An $AR[p]$ process is an $ARMA[p, 0]$ process and an $MA[q]$ process is an $ARMA[0, q]$ process. Note that a white noise also fits this framework in that it is an $ARMA[0, 0]$ process.

2 Building and validating stochastic models

We have developed a systematic procedure for model fitting to real data by courtesy of *Siemens AG – Healthcare Sector Customer Services Material Logistics* (a.k.a. *Siemens Medical Solutions*), Erlangen, Germany. Initially, no independence assumption are made but after computing comprehensive statistics, the independence of data is checked. Dependent on the outcome of suitable tests, either the time series corresponding to interdemand times and demand sizes are fitted to autoregressive moving average (ARMA) processes or fitted to adequate probability distributions.

The essential steps of our modeling procedure and their order in an automated algorithmic application starting with the raw data are outlined below. We emphasize that this procedure is flexible and well accessible to practitioners.

Many steps are supported by statistical software packages, which is important for being viable as a part of real inventory management within industrial companies.

1. Study summary statistics and the correlation structure of interdemand times and demand sizes
2. Test the independence hypothesis: Ljung-Box test

3. If independent, fit data to appropriate probability distribution
 - a. Select potentially appropriate distribution families based on summary statistics
 - b. For each candidate distribution, obtain parameters by maximum likelihood estimation
 - c. Validate goodness-of-fit by visualization: graphical plots
 - d. Validate goodness-of-fit by statistical tests: χ^2 , Anderson-Darling, Kolmogorov-Smirnov
4. If not independent, fit data to ARMA model
 - a. Select potentially appropriate class of ARMA based on (partial) autocorrelation functions
 - b. For each candidate class, obtain parameters by
 - i. least squares estimation for the AR part
 - ii. numerical iteration for the MA part
 - c. Validate goodness-of-fit by residual analysis

This procedure has been applied to the demand patterns of 54 different slow-moving items, each recorded from September 1994 to May 2008. In addition to model fitting for each of these items, one goal was to identify similarities in order to build an aggregated model that integrates as much slow-moving items as possible (inventory of Siemens Medical Solutions takes care about altogether about 8500 slow-moving items). In the following we describe some more details of the steps and outline the main findings that we obtained in this manner.

Note that we could have formulated our model fitting procedure without explicitly distinguishing between independent and dependent data by just fitting to an ARMA process and keeping in mind that an ARMA[0,0] process is a white noise which means that the data is independent. However, this would be overly generalized. We make the distinction with regard to the specific fitting methodologies, which are much easier for independent data.

2.1 Summary Statistics and Correlation Structure

Summary statistics are usually not common in time series analysis when trends, seasonality, dependencies or correlations are present. Nevertheless, they should be computed as a first step in any statistical data analysis because they almost always give useful insights, in particular in the case of intermittent demands where completely random patterns in the sense of iid data or white noise are very likely.

In addition, the correlation structure is of major important in time series analysis. Therefore, we consider a variety of statistical measures that give us a first quantitative impression of the data and its correlation structure. More precisely, we compute the following empirical measures from the time series data x_1, \dots, x_n .

- Empirical mean

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i,$$

- Empirical variance

$$s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2,$$

- Empirical standard deviation

$$s = \sqrt{s^2},$$

- Empirical coefficient of variation

$$c = s/\bar{x},$$

- Empirical standard error

$$\sigma_n = s/\sqrt{n},$$

- Empirical skewness

$$\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^3 / \sqrt{\left(\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2\right)^3},$$

- Empirical kurtosis

$$\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^4 / \left(\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2\right)^2 - 3,$$

- Empirical covariance

- $s_{xy} = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y}),$

- Empirical coefficient of correlation

$$r_{xy} = s_{xy}/(s_x s_y)$$

Note that the latter two measures can be computed with regard to different time series as well as within a single time series. One can shift the time series by a lag τ and interpret the resulting pairs $(x_i, y_i) = (x_i, x_{i+\tau})$ in the same way as data points from different time series. Concerned with a single time series one also speaks of autocorrelation as for stochastic processes. In particular, when considering such autocorrelations for different lags, one can get useful insights on the strength of potentially present dependencies, which then yields guidelines for the choice of appropriate ARMA processes.

2.2 Independence test

After computing comprehensive summary statistics and studying the correlation structure of the given data, our next step is to test the hypothesis of independence. More specifically, we want to examine whether it is appropriate to assume that successive interdemand times and demand sizes are independent. In the terminology of stochastic time series models, this independence hypothesis corresponds to the hypothesis that the data generating stochastic processes are white noise.

Hence, we are concerned with the independence of successive data from one time series (not with the independence of two or more data sets). In order to test it, the Ljung-Box test [9] can be applied.

With the Ljung-Box test all autocorrelation coefficients are considered simultaneously. The independence hypothesis H_0 and its alternative H_1 are defined as

$$H_0: \rho(s) = 0, \text{ for all } s \in \mathbb{N}, s > 0,$$

$$H_1: \{s \mid \rho(s) \neq 0, s \in \mathbb{N}, s > 0\} \neq \emptyset$$

and the test statistic is computed by

$$T(\hat{\rho}) = n(n+1) \sum_{j=1}^h \hat{\rho}_j^2 / (n-j).$$

where n is the sample size, $\hat{\rho}_j$ is the empirical coefficient of autocorrelation on lag j and h is the number of lags. Then for a given significance level α the critical section is defined by

$$T(\hat{\rho}) > \chi_{1-\alpha, h}^2$$

where $\chi_{1-\alpha, h}^2$ is the $1 - \alpha$ -quantile of the chi square distribution with h degrees of freedom.

The Ljung-Box test has been specified as an S-Plus program and applied to the intermittent demand patterns, that is to the interdemand times and the demand sizes of all slow-moving items. One of our main findings is that for almost all intermittent demand patterns the independence assumption is valid.

More specifically, according to the Ljung-Box test on a statistical significance level of 0.05, the independence hypothesis was only rejected for two out of the 54 items considered, and the run tests that we additionally performed did not indicate any dependence.

2.3 Fitting dependent data

If the data is assumed to be dependent, our modeling procedure proceeds by fitting the data to an ARMA process.

In order to find appropriate ARMA processes that well represent the measured data, we first selected a potentially appropriate class of ARMA models based on partial autocorrelation functions, estimated the corresponding parameters and validated the fit by residual analysis. Roughly speaking, the fit is made iteratively such that processes of different orders are successively fitted and the sum of squares of differences between the fitted process and the time series data is computed. Finally, the order providing the least square sum is chosen.

Informally, the partial correlation between two variables is the correlation that remains if the possible impact of all other random variables has been eliminated and it is much easier to determine the order of an ARMA process via the partial autocorrelation function than via the original autocorrelation function.

For our intermittent demand data, inspection of the partial autocorrelation functions suggested that pure AR processes are most appropriate. We have specified an according procedure in S-Plus and specifically used the Akaike information criterion (AIC) where

$$AIC(k) = n \log(\hat{\sigma}_{\varepsilon, k}^2) + 2k$$

is considered and the k for which $AIC(k)$ is minimal yields the estimated order of the AR process. For the details we refer the reader to, e.g., [1, 2, 4, 5, 8].

Even for the two potentially critical items that did not pass the Ljung-Box test for independence, the best fits to ARMA processes resulted in neglecting the MA part and low orders of the AR part, that is, these data were best fitted to purely autoregressive processes of order 2 and 4, respectively, which indicates a very weak dependence. Taking the data as independent and fitting to probability distributions resulted in very accurate fits.

Hence, it seems reasonable to assume independence even for these two items. Note that statistical tests give only statements with certain statistical significance, neither proofs nor disproofs of hypotheses.

3 Forecasting and stock control

Once we have built stochastic models and validated their appropriateness, it is clear that deterministic point forecasts are not very meaningful but stock control strategies are possible which do not rely on simple point forecasts.

We obtain service level guarantees in terms of probability bounds on stock out or item availability, respectively. More specifically, we can compute the probability of a demand size being greater than some given threshold, which is closely related to quantiles and tail probabilities of the fitted demand size distribution. Furthermore, to be useful for inventory management in practice where typically not every month each item stock is checked we also need to consider a larger time horizon as the planning period.

Therefore, we compute the probability of more than a given number of demands within a certain time period, essentially via tail probabilities of sums of random variables. After all, we end up with stock control strategies guaranteeing that, given a desired service level in terms of probabilities and the constraint of minimized inventory costs, for every inventory period sufficiently many units of all items are in stock. Finally, thanks to stochastic similarities, items can be aggregated yielding an integrated inventory control system.

Hence, altogether we have a mathematically well-founded model fitting procedure for practicable stock control of slow-moving items that seems to be promising and overcomes some of the weaknesses of currently practiced methods.

3.1 Stock control exemplification

We demonstrate a possible application in practice by a simple example where we assume that the interdemand times and demand sizes are independent and have been properly fitted to probability distributions. We further assume that for an example item a prescribed service levels should be achieved within a time horizon of n months.

The service level is considered to be the item availability, that is the probability η that demands for this item can be immediately served by the units in stock. The question arises how many units of this item must be in stock for a given desired availability.

Though the demand size has no theoretical upper bound, the demand size distribution allows statements on the probability of certain demand sizes. We argue that for practicable stock control only demand sizes with some minimum probability can be reasonably taken into account. In other words, we consider a lower probability bound for the demand sizes such that all demand sizes with a smaller probability are neglected.

Then, whenever a demand occurs, the units to be hold in stock should equal the η -quantile Q_η of the demand size distribution times the number of months in which the item is actually demanded.

Hence, by considering a time horizon of more than one month for the planning period, we are even able to serve extreme demands with unlikely large demand sizes, provided that they do not occur multiple times within the time horizon of the planning period.

The remaining question is how often (in how many months) within the next n months we should be prepared for demands. We do not simply take the expectation of the number of demands within n months, because this does not account for the specific probability distribution.

Similarly as for the demand sizes, we consider an upper bound such that only with a small probability demands occur in more months than suggested by the bound. This probability is determined by and can be obtained from the interdemand size distribution.

For numerical illustration, we consider an example item that has according to our fitting procedure geometrically distributed interdemand times with parameter $p = 0.21168$.

Hence, the probability that the item is demanded in the next month equals p , which is relatively large. This means we should be prepared for a demand in the next month. Moreover, the number of demands is binomially distributed, that is the probability of exactly j demands within the next n months is given by

$$P_{j,n} := \binom{n}{j} p^j (1-p)^{n-j}$$

Now, assume that our time horizon of the planning period is six months. Then the probability that the item is demanded in each of these next six months is

$$P_{6,6} := \binom{6}{6} p^6 (1-p)^0 = 0.00009,$$

which is extremely small. Such a demand pattern will happen on average every thousand years such that it is reasonable to neglect it. Clearly, other patterns are more likely. For instance, the probability of exactly one demand within the next six months is $P_{1,6} = 0.38667$.

Hence, we have to choose a bound ζ such that any probability below this bound will be considered too small to realistically assume the corresponding demand pattern.

Let $\zeta = 0.01$. From the remaining demand patterns with larger probability, choose those with the largest number of months with demand. Multiplying this number with the η -quantile Q_η of the demand size distribution then gives the number m of units required in stock.

For our example item the demand sizes are logarithmically distributed with parameter $\vartheta = 0.54295$. With the choice of $\eta = 0.95$, the 95% quantile of the demand sizes computes as

$$Q_{0.95} = \min \left\{ x : \frac{-1}{\ln(1-\vartheta)} \sum_{i=1}^x \frac{\vartheta^i}{i} \geq 0.95 \right\} = 4.$$

Now, for demonstration purposes, successively compare the probabilities $P_{j,n}$ with $\zeta = 0.01$:

$$\begin{aligned} P_{1,6} &= 0.38667 > \zeta, \\ P_{2,6} &= 0.25957 > \zeta, \\ P_{3,6} &= 0.09293 > \zeta, \\ P_{4,6} &= 0.01872 > \zeta, \\ P_{5,6} &= 0.00201 < \zeta. \end{aligned}$$

Hence, we take four demands within the next six months as sufficiently likely to be prepared for it and the number of units required in stock is therefore

$$m = Q_{0.95} \cdot \max\{j : P_{j,n} > \zeta\} = 4 \cdot 4 = 16$$

The outlined approach is astonishingly simple and provides a useful way of assuring certain service levels. Nevertheless, a couple of issues require further investigation such as a reasonable planning period (time horizon) over which stock control should be considered.

Besides, an additional option to react within the planning period when the number of units in stock falls below a critical level would be surely useful. However, with regard to these points, there are usually practical constraints and not everything what is theoretically desirable is practically possible.

4 Conclusion

We have presented a stochastic modeling approach for the demand patterns of slow moving items with regard to intermittent demands forecasting and stock control. The key problem in intermittent demands forecasting and stock control is the demand pattern of slow moving items which renders traditional deterministic exponential smoothing techniques inappropriate.

Stochastic models are required to capture the intermittent demand pattern. It turns out that stochastic time series models are well suited. In many cases the intermittent demand patterns even appear to be purely random in the sense that interdemand times and demand sizes are iid random variables corresponding to a white noise process.

A procedure has been suggested for fitting real data to suitable stochastic models based on which forecasting and stock control become well-founded and automated.

While this modeling procedure already seems to be mature enough to address items separately without any substantial improvements necessary, further research should deal with aggregated models for multiple items.

With regard to stock control for single items, one potential application of the stochastic model has been demonstrated and similar approaches with an aggregated model are highly desirable. Hence, stochastic models for intermittent demands forecasting and stock control appear to be promising and have already proven useful but a lot of future work is still required and already ongoing.

References

- [1] G.E.P. Box and G.M. Jenkins. *Time Series Analysis: Forecasting and Control*. Holden Day, 1970.
- [2] G.E.P. Box, G.M. Jenkins, and G. C. Reinsel. *Time Series Analysis: Forecasting and Control*. John Wiley & Sons, 4th edition, 2008.
- [3] J.E. Boylan and A.A. Syntetos. *The accuracy of a modified Croston procedure*. International Journal of Production Economics, 507:511–517, 2007.
- [4] P.J. Brockwell and R.A. Davis. *Introduction to Time Series and Forecasting*. Springer, 2nd edition, 2002.
- [5] C. Chatfield. *The Analysis of Time Series: An Introduction*. Chapman & Hall, 6th edition, 2004.
- [6] J.D. Croston. *Forecasting and stock control for intermittent demands*. Operational Research Quarterly, 23(3):289–303, 1972.
- [7] R.V. Hogg, A.T. Craig, J.W. McKean. *Introduction to Mathematical Statistics*. Prentice Hall, 6th ed., 2004.
- [8] G. Kirchgässner and J. Wolters. *Introduction to Modern Time Series Analysis*. Springer, 2007.
- [9] G. M. Ljung and G. E. P. Box. *On a measure of lack of fit in time series models*. Biometrika, 2(65):297–303, 1978.
- [10] W. C. Navidi. *Statistics for Engineers and Scientists*. McGraw Hill, 2nd edition, 2008.

- [11] A.V. Rao. *A comment on: Forecasting and stock control for intermittent demands*. Operational Research Quarterly, 24(4):639–640, 1973.
- [12] B Sani and B.G. Kingsman. *Selecting the best periodic inventory control and demand forecasting methods for low demand items*. Journal of the Operational Research Society, 48:700–713, 1997.
- [13] L. Shenstone and R.J. Hyndman. *Stochastic models underlying Croston's method for intermittent demand forecasting*. Journal of Forecasting, 24:389–402, 2005
- [14] E.A. Silver, D.F. Pyke, and R. Peterson. *Inventory Management and Production Planning and Scheduling*. John Wiley & Sons, 1998.
- [15] A.A. Syntetos and J.E. Boylan. *On the bias of intermittent demand estimates*. International Journal of Production Economics, 71:457–466, 2001.
- [16] A.A. Syntetos and J.E. Boylan. *The accuracy of intermittent demand estimates*. International Journal of Forecasting, 21:303–314, 2005.
- [17] R. Teunter and B. Sani. *On the bias of Croston's forecasting method*. European Journal of Operational Research, Article in Press, 2008.
- [18] T. Wild. *Best Practice in Inventory Management*. Butterworth-Heinemann, 2nd edition, 2002.
- [19] P. H. Zipkin. *Foundations of Inventory Management*. McGraw-Hill, 2000.

Corresponding author: W. Sandmann
University of Bamberg,
Department of Information Systems and Applied
Computer Science
96045 Bamberg, Feldkirchenstraße 21, Germany
werner.sandmann@uni-bamberg.de

Received & Accepted: MATHMOD 2009 -

Revised: September 10, 2009 -

Accepted: July 10, 2010 -

Decision Making with a Random Walk in a Discrete Time Markov Chain

R. Chelvier, G. Horton, C. Krull, B. Rauch-Gebbensleben

University of Magdeburg, Computer Science Department, Magdeburg, Germany

SNE Simulation Notes Europe SNE 20(3-4), 2010, 37-42, doi: 10.11128/sne.20.tn.09989

The paper describes a Markov model for multi-criteria and multi-person decision making. The motivation results from a demand observed in the early stages of an innovation process. Here, many alternatives need to be evaluated by several decision makers with respect to several criteria. The model derivation and description can be split into the evaluation process and the decision process. The pair wise comparisons can be combined by weighting them according to the importance of the criteria and decision makers, resulting in a discrete-time Markov chain. A random walk on this DTMC models the decision process, where a longer state sojourn time implies a better alternative. We believe that this model meets the demands in the early stages of an innovation process.

1 Description of the problem

We consider the problem of evaluating alternatives in the early stages of an innovation process. In this application area alternatives need to be evaluated by several decision makers with respect to different criteria. There are possibly many alternatives that need to be considered; therefore it is necessary to make the evaluation process fast and simple. The problem parameters are the following:

- A possibly large number of alternatives may be involved
- Several decision makers may be involved
- Several evaluation criteria (both quantifiable and soft) may be involved
- The evaluation criteria may be weighted according to relevance
- The opinions of the decision makers may be weighted according to expertise
- Little or no information is available about the alternatives; the decision makers base their evaluations on intuition or guesswork
- The decision makers have to decide fast due to the possibly large number of alternatives

The following example describes the intended application. During an innovation workshop a large number of ideas are produced. Each idea is described only with a title and a short characterisation. An innovation team must identify the top ten ideas to bring them forward to the first stage of a stage-gate process [4]. Little or no quantifiable information is available

about the ideas, therefore it is not possible to rank the ideas based on objective criteria. Instead, only subjective impressions are available at this stage, enabling decisions of the form “A is better than B” with respect to a given criterion. Each member of the team might have a different area of expertise or competence, and to each of them can be assigned a different weight for different criteria. For each pair wise comparison the decision maker and the criteria is noted.

The following questions need to be addressed: How to model an evaluation process and a decision process with the specified parameters? How to deal with inconsistent or non-transitive evaluations, which can occur due to the subjective nature of the evaluations? How to determine the top alternatives?

2 State of the art

In the field of multi-criteria decision making (MCDM) many methods have been developed for specialised applications. Detailed information about MCDM can be found, for example in [10], [8] and [1]. Thirty available methods are discussed in [7]. Two more general methods which can be used in the early stages of an innovation process are AHP (the Analytic Hierarchy Process) [2, 11] and cost-benefit analysis (CBA) [3].

However, AHP and CBA are not directly applicable to the intended application. AHP does not support multiple decision makers, unless additional aggregation strategies are applied to merge the individual evaluation result [9, 13, 6].

Furthermore, AHP requires consistent transitive evaluations in order to compute a valid result. CBA needs measurable and quantifiable criteria to compute a valid result.

However, inconsistent evaluations and soft criteria are often present in the early stages of an innovation process. Accordingly, AHP and CBA cannot be the preferred methods to evaluate alternatives under these circumstances.

3 A DTMC-based model for evaluation and decision process

This section describes how to model both the evaluation process and the decision process. The evaluation process is similar that in AHP but with only one level of “is better than”. In the given application it is not applicable to use more than one level of difference, because there is no strict differentiation possible to get more detailed decisions for two reasons. Firstly, little or no information about the alternatives may be available and secondly the use of soft criteria.

The model of the decision process is described using an analogue situation.

3.1 Evaluation process

Based on the assumption that little or no information about the alternatives is available, the evaluation process is implemented as pair wise comparisons between all alternatives, concerning all criteria. These comparisons only ask for a decision of the following form “is better than”. This solution allows comparisons according to non-measurable evaluation criteria, such as taste or preference.

We assign weights to the decision makers according to their expertise and to the evaluation criteria according to their relevance, and scale these weights to sum up to one. We build a weighted directed graph, where each comparison adds an edge from the less preferred alternative to the better one. The edge weight results from the weight of the criterion and the decision maker. After adding the edges corresponding to the comparisons to the graph, we can scale the edge weights such that the sum of all outgoing edges of a node is one. The resulting graph is a discrete-time Markov chain, where the edges lead from the less preferred to the better alternatives.

A detailed description of the mathematics behind the evaluation process can be found in [4].

In the evaluation process we have participants p_k with $k = 1 \dots K$, criteria c_l with $l = 1 \dots L$ and alternatives a_m with $m = 1 \dots M$. Each participant p_k makes pair wise comparisons between two alternatives a_{m1} and a_{m2} with respect to criterion c_l . We denote this as follows: $p_k(c_l): a_{m1} > a_{m2}$.

Next we need the coefficient α_{kl} to assign weights to each evaluation made by the participants. Each coefficient α_{kl} contains information about the relevance of participant p_k with respect to criterion c_l and describes the importance of criteria c_l where larger values imply greater importance. In the matrix A of dimension $K \times L$ we store the coefficients that satisfy

$$0 \leq \alpha_{kl} \leq 1 \text{ and } \sum_{k=1}^K \sum_{l=1}^L \alpha_{kl} = 1.$$

Each evaluation of the participant p_k with respect to criterion c_l is represented in the matrix E_{kl} of dimension $M \times M$. We build the matrix as follows:

$$E_{kl(m_1, m_2)} = \begin{cases} 1/(\delta_{m_1} + 1) & p_k(c_l): a_{m_2} > a_{m_1} \\ 0 & \text{otherwise} \end{cases}$$

where δ_{m_1} represents the number of non-zero entries in the m_1 -th row of matrix E_{kl} . The main diagonal coefficients are set as follows:

$$E_{kl(m_1, m_1)} = 1 - \sum_{m_2=1, m_2 \neq m_1}^M E_{kl(m_1, m_2)}.$$

Finally, we need a matrix P of dimension $M \times M$. P contains the complete set of evaluations and is computed from the weighted sum of all E_{kl} . P is also a stochastic matrix and computed by

$$P = \sum_{k=1}^K \sum_{l=1}^L \alpha_{kl} E_{kl}.$$

The result of the evaluation process is a DTMC containing all evaluations of the participants as weighted edges.

3.2 Decision process

We believe that after having built this DTMC from the decision makers' evaluations, the decision process corresponds to a random walk on this resulting Markov chain. We assume that the sojourn time of better alternatives is larger than for inferior ones. To determine the preferred alternatives, the DTMC is solved, resulting in the steady state probability vector. The more incoming edges with large weights one node has, the more comparisons were made preferring that alternative. The more outgoing edges with large weights one node has, the fewer comparisons preferred that alternative.

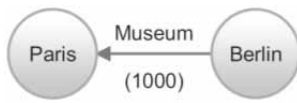


Figure 1. 1000 students evaluated Paris better than Berlin for the museums.

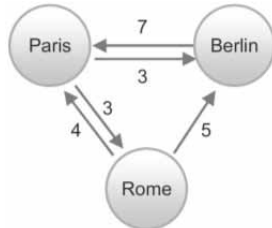


Figure 2. Example bus departure frequencies (x times a week)

Consequently the better alternatives with more incoming edges have a larger probability in the result. The resulting probability vector is then interpreted as a ranking of the alternatives, where larger probabilities show a higher rank.

To obtain the best alternative we take a look to an analogue situation: Assumed all students from a university meet after summer and evaluate impressions from their holiday trips in Europe with respect to several criteria. As a result they designed a visitor guide from statements like “I’ve visited Berlin for its museum and then I’ve visited Paris. Paris was even better than Berlin.” This result could be visualized like shown in Figure 1.

Furthermore we assume a bus company with special travel offers for students. This company may use the visitor guide to adjust the connection frequencies between the capitals in Europe. In our example (see Figure 2), the connection frequency from Berlin to Paris is higher than the other way, if Paris gets more positive evaluations in comparison with Berlin (all criteria merged):

Finally we have a student who makes a tour in Europe. In every capital he takes the first bus which is departing to another capital on his arrival at the bus station. Now, some friends of the student want to join his tour. In which capital do they have the highest possibility to meet the student?

Because the student decides on each bus station randomly, we believe the student acts like a “random walker” in a directed graph. With the different bus departure frequencies the solution is to meet the student in the capital with the highest sojourn time of the random walker.

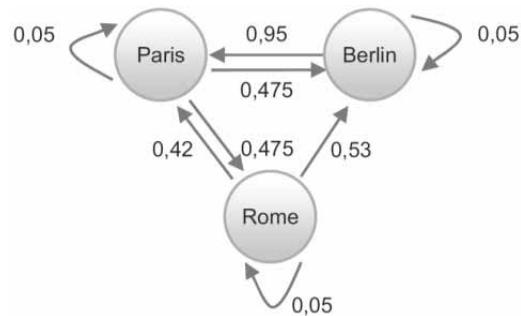


Figure 3. DTMC with normalized weights (corresponding to the departure frequency)

Since the sojourn time of a random walker corresponds to the values in a steady state probability vector of a discrete-time Markov chain (DTMC) his friends should build and solve a DTMC. DTMCs are well-researched mathematical models with many applications in Science and Engineering. A DTMC is described by a stochastic matrix P and a probability vector π . The steady-state solution of the DTMC contains the probabilities of each of the system states and is given by the solution of the linear system of equations $\pi P = \pi$. Markov chains are drawn as weighted directed graphs, where the nodes represent the states and the edges represents the possible state transitions. The weights associated with the edges describe the one-step probabilities for each state transition. A state or set of states of a Markov chain is called absorbing, if it contains only incoming edges [12].

The capitals are represented as nodes and the bus connections as directed edges. The weight of each edge corresponds to the departure frequency of the buses for each connection (Figure 3). Figure 3 also contains nodes with self-pointing edges. A self-pointing edge with value 1 means, no bus will depart from this capital (consequently the student will stay there). Self-pointing edges with values lower than one represent the probability for the student to spend another day in the capital (in this example, the value is assumed to be five per cent for each capital).

Before computing the ranking vector π , we need to consider one case that distinguishes our approach from a genuine random walker: The student decides to pick a capital randomly and not to decide using the bus station. In this case we assume the probability to travel to a certain capital is equal for all capitals (see Figure 4). To solve this problem we build a new matrix F of dimension $M \times M$ with interconnections to all capitals with the weights $1/n$:

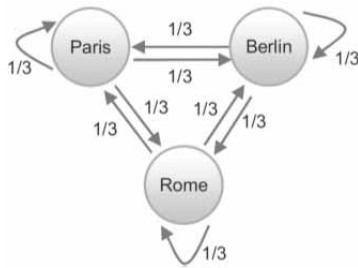


Figure 4. Graph representing matrix F with the weight $1/n$ (n : number of capitals)

Now we can compute matrix $R = (1 - \varepsilon)P + \varepsilon F$ using the parameter ε with $0 < \varepsilon \ll 1$ which describes the probability that student picks the next capital at random not using the buses. The matrix R is stochastic and irreducible. Furthermore R defines a Markov chain without absorbing states (Figure 5).

The algorithm to compute the ranking vector π contains both, the evaluation process and the decision process. The algorithm is given by a sequence of five steps:

1. Choose the coefficients α_{kl} .
2. Choose the value for ε .
3. Enter the values for the pair wise comparisons a_{m1} and a_{m2} into E_{kl} and calculate the values for the main diagonal of E_{kl} .
4. Compute R by adding all E_{kl} weighted by the coefficients.
5. Solve the DTMC, computing the steady state solution π .

After termination of the algorithm the result of each alternative a_m is equivalent to the value π_m . The larger the value π_m is, the higher is the rank of a_m .

Starting with the initial probabilities $\pi_0 = (1/3, 1/3, 1/3)$ and $\varepsilon = 0.01$ the solution vector of the example is $\pi = (\text{Paris}, \text{Berlin}, \text{Rome}) = (0.44, 0.34, 0.22)$.

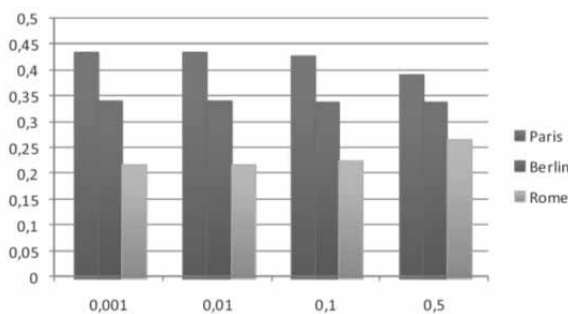


Figure 6. Influence of ε on the values in the solution vector π .

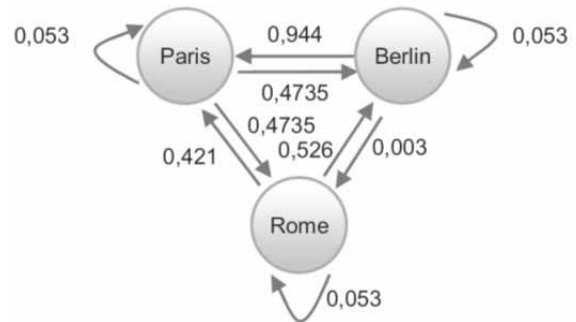


Figure 5. DTMC from Matrix R with $\varepsilon = 0.01$

Consequently the friends of the travelling student have the highest possibility to meet the student in Paris.

Even though ε changes the values for π_m , this influence can be neglected as experiments have shown. A large value of ε only dampens the vector solutions, but does not influence the ranking positions. Experiments have shown that the larger the value of ε , the less the gap between the values in π (see Figure 6).

The same approach we illustrated for the travelling student, we chose to determine the ranking of alternatives in the early stages of an innovation process. Therefore we built a Markov chain R based on the weighted pair wise comparisons made by the participants. Afterwards we send the random walker through the Markov chain who wants to visit all preferred alternatives. To avoid to be caught in a local maximum he can choose his next randomly, not using recommendations. Therefore we use the matrix F with entries of size $1/M$. The parameter ε now contains the probability that the random walker decides to continue his journey on a randomly chosen alternative. After computing matrix R , we compute the sojourn time of a random walker for each alternative and received the solution vector π .

Finally, the steady state solution of the DTMC then yields the ranking of the alternatives.

4 Discussion of the advantages in our approach

We believe that using our DTMC based decision process we can solve the following problems. The top alternatives are identified by their larger values in the probability vector, which correspond to the sojourn time of a random walker.

A DTMC can also handle contradicting and intransitive comparison results; it can contain edges running in opposite directions. By using the weights assigned to the decision makers and criteria to weight the edges, we can easily combine their comparisons made into one DTMC. The steady state solution of the DTMC then yields the ranking of the alternatives.

Furthermore, our approach has the following advantages. The method allows quantifiable and soft evaluation criteria. The method enables decision makers to evaluate alternatives with little or no information available. The method combines a decision making problem with an established mathematical method (discrete-time Markov chains). By combining only some of the comparisons made, it is possible to compute intermediate results during the evaluation process. The method is simple to use, because we only ask for “better than” decisions. The method can easily represent levels of competence of the decision makers with respect to the criteria.

We will now discuss all of these points one by one and show why they apply to our method.

The model can handle contradicting and intransitive comparisons. This statement we can confirm because a DTMC can handle contradicting and intransitive comparison results; it can contain edges running in opposite directions. This case can occur when evaluations concerning different criteria are made. E.g.: Car 1 is cheaper than car 2 and car 2 is more comfortable than car 1. This will lead to edges in opposite directions in the DTMC. However the weight of the edges will be different depending on the weights assigned to each of the criteria.

The model can handle inconsistent evaluations. This statement can also be confirmed, because a DTMC can handle these evaluations. Two inconsistent evaluations can occur, when decision makers have contradicting opinions concerning soft criteria. For example: Expert 1 thinks, car 1 is more comfortable than car 2 because of a better driving seat. Expert 2 prefers car 2 over car 1, concerning comfort, because he likes the more spacious backseat. This again results in two edges running in opposite directions, but having different weights according to the decision makers’ expertise.

The model can handle soft and quantifiable evaluation criteria. This statement can be confirmed as well. “Better than” comparisons allow soft as well as measurable evaluation criteria.

The advantage of this simple decision is that they can be done much easier within a short period of time. “Wrong” decisions should be cancelled out by the mass of other evaluators, criteria and transitive relationships.

The model can handle multiple decision makers. To achieve that, we add up weighted decision matrices of each decision maker. The more we value the opinion of a decision maker, the more weight we can give his evaluations. This results in more expertise or more influence in the decision process. Therefore this statement was confirmed as well.

Intermediate results during the evaluation process are available. We can confirm this statement, because adding one decision of a decision maker keeps the properties of a DTMC intact. Therefore it can be solved having added only some of the edges corresponding to the evaluations. The remainder of the probability of one node stays in that node itself. This implies the initialization of all the matrices to the Identity matrix. These intermediate results allow us to stop the evaluation process before all pair wise comparisons have been made, and getting a ranking. The quality of that ranking has to be determined otherwise (discussed in Section 5).

In some of our experiments we observed the so-called rank reversal effect. One case where it occurs is, when we insert a new node similar to an existent node (e.g. two equal cars in different colours). In this case the new rank of both nodes can be smaller than the original node’s rank. This occurs because the incoming edges probabilities are divided between the similar nodes. In our opinion, when adding or removing alternatives and their respective evaluations, we change the nature of the problem. We found an analogue case which can be explained by the following example: The ranking of the German Football League is also obtained by pair wise comparisons, the games. Assuming, we remove Hoffenheim and all the games they played from the evaluations, the following phenomena can occur. Teams that won against Hoffenheim loose points and teams that lost against Hoffenheim do not loose points. This can change the ranking. When removing a contestant without removing their games or other teams’ points, the global ranking is not affected. For a DTMC this removal would mean adding edges which correspond to evaluations that were not made, but that were implicitly there because of transitivity.

Since we do not want to add evaluations, which were not actually entered by a decision maker, rank reversal is a property of our model.

5 Conclusion and outlook

In this paper we described a model for the decision process in the early stages of an innovation process. We used pair wise comparisons and weights for decision makers and criteria to combine them to form a discrete-time Markov chain. A random walk on this chain models the actual decision process. The solution of the Markov chain yields the probability vector, which gives us a ranking of the alternatives. In contrast to existing methods, our model can easily handle inconsistent evaluations, soft criteria and multiple decision makers.

Since intermediate rankings can also be computed, we see improvement potential in the model by reducing the number of comparisons necessary to reach a certain goal. As far as our experience goes the most interesting goals in the early stages of an innovation process are the following:

- Identify the best alternative: Due to limited resources or project type only the best alternative is needed.
- Identify the top x alternatives: This comes directly from the properties of the stage gate process in innovation management. Limited resources restrict the number to only some innovation projects. Here the ranking of these top x among each other is unimportant.

In these cases the evaluation process can be aborted if the evaluation goal is reached. To be able to implement this, we need to find heuristics to decide which comparison should be asked for next. This might involve ordering the possible judgements according to their effect on the ranking vector. Then, the algorithm can prompt the decision makers to input the more influential judgements first. Our assumption is that the more judgements are made, the less effect on the ranking vector is measurable. That means that the probability of a rank exchange decreases. As first experiments showed, the number of necessary judgements to reach one of these evaluation goals decreases considerable in comparison to obtaining an accurate ranking result by entering all possible pair-wise comparisons.

We think that our method can be applied to many other applications with equal conditions, also beyond the innovation management.

References

- [1] Belton, V. and Stewart, T.J.: *Multiple Criteria Decision Analysis*, Kluwer Academic Publishers, Boston, Dordrecht, London, 2002.
- [2] Caklovic, L., Piskac, R., and Segó, V.: *Improvement of AHP method*, Mathematical Communications, Supplement 1; 13-21, 2001.
- [3] Chakravarty, S.: *Cost-benefit analysis*. In: The New Palgrave: A Dictionary of Economics, 1987, volume 1, pp. 687-90.
- [4] Chelvier, R., Dammasch, K., Horton, G., Knoll, S.-W., Krull, C., and Rauch-Gebbensleben, B.: *A Markov Model for Multi-Criteria Multi-Person Decision Making*. In: Proc. 5th Int. Conf. on Innovations in Information Technology, Al-Ain, United Arab Emirates. 2008.
- [5] Cooper, R.: *The New Product Process: A Decision Guide for Managers*. Journal of Marketing Management, 1988, volume: 3.
- [6] Eisenführ, F. and Weber, M.: *Rationales Entscheiden*, 2nd edition, Springer-Verlag, Berlin, Heidelberg, 1994.
- [7] Guitouni, A., Martel, J.-M., and Vincke, P.: *A Framework to Choose a Discrete Multicriterion Aggregation Procedure*, Defence Research Establishment Valcatier (DREV), 1998.
- [8] Hokkanen, J.: *Aiding Public Environmental Decision Making by Multicriteria Analysis*, Report 72, University of Jyväskylä, 1997.
- [9] Meixner, O. and Haas, R.: *Computergestützte Entscheidungsfindung: Expert Choice und AHP: innovative Werkzeuge zur Lösung komplexer Probleme*, Wirtschaftsverlag Carl Ueberreuter, 2002.
- [10] Roy, B.: *A French-English Decision Aiding Glossary*. European Working Group "Multicriteria Aid for Decision", Series 3, Number 1, 2000.
- [11] Saaty, T.: *The Analytic Hierarchy Process*. 1980, McGraw-Hill, New York.
- [12] Stewart W.J.: *Introduction to the Numerical Solution of Markov Chains*, Princeton University Press, Princeton, NJ, 1994.
- [13] Vetschera, R.: *Entscheidungsunterstützende Systeme für Gruppen: Ein rückkopplungsorientierter Ansatz*, Physica-Verlag, Heidelberg, 1991.

Corresponding author: R. Chelvier,

University of Magdeburg,
Computer Science Department
Universitätsplatz 2, 39106 Magdeburg, Germany;
rene@sim-md.de

Received & Accepted: MATHMOD 2009

Revised: September 10, 2009

Accepted: July 10, 2010

CPL Clustering Based on Linear Dependencies

Leon Bobrowski, Białystok Technical University, Warsaw, Poland

SNE Simulation Notes Europe SNE 20(3-4), 2010, 43-48, doi: 10.11128/sne.20.tn.09991

Discovering linear dependencies in data sets is discussed in the paper as a part of data mining approach [1]. The proposed method is based on the minimization of a special type of convex and piecewise linear (CPL) criterion functions defined on a given data set C [2]. The division of the set C into a family of linearly dependent clusters C_k allows to form a family of local regression type models. As a result, each subset C_k can be characterized C_k by its own linear model. The K -plans algorithm which is similar to the K -means algorithm can be used for dividing the set C into a family of linearly dependent clusters C_k . Also a different approach to this problem, based on the CPL criterion functions is discussed here.

Introduction

Data mining is a process of extracting hidden patterns from data [1]. Generally, data mining techniques can be useful in transformation of data sets into needed information. Such techniques are commonly used in a wide range of applications, such as marketing, fraud detection and scientific discovery. The term *patterns* could stand for regularities, trends, association rules or clusters in the explored data set.

A fundamental role in the cluster analysis is played by the K -means algorithm [2]. The K -means algorithm can be used for the purpose of dividing set C into a family of a priori given number K of clusters C_k ($k = 1, \dots, K$). The central points \mathbf{w}_k can be identified for each subset C_k during the K -means procedure through the minimization of convex and piecewise linear (CPL) criterion functions [3]. Modification of the K -means algorithm into the K -plans algorithm has been proposed recently [4].

The proposed method is based on the minimization of a special type of the CPL criterion functions defined on a given data set C [2]. The basis exchange algorithms which are similar to linear programming allow one to find the minimum of these CPL function efficiently, even in the case of large multidimensional data sets [3]. The minimization of the CPL criterion function during the K -plans procedure allows to identify the actual values of the parameters \mathbf{w}_k and θ_k of the central hyperplane

$$H(\mathbf{w}_k, \theta_k) = \{\mathbf{x}[n]: \mathbf{w}_k[n]^T \mathbf{x}[n] = \theta_k\}$$

for each subset C_k . In the next step of the K -plans algorithm the division of the set C into the subsets C_k is modified and adopted to the actual central hyperplane $H(\mathbf{w}_k, \theta_k)$.

The central hyperplane $H(\mathbf{w}_k, \theta_k)$ defines the local, linear dependency characteristic for a given subset C_k . As a result, each subset C_k can be characterized by its own linear model of dependencies.

The procedure of hidden linear dependencies extraction from data set different from the K -plans is also described and analysed in the presented paper. The presented procedure is based on monotonicity properties of the CPL criterion function.

This procedure allows to identify a family of K subsets C_k and local, linear dependencies without assuming a priori the value of the number K . The number K of linear models results from a structure of the explored data set C .

The proposed approach can be used for solving a variety of data mining problems. One of them is discovering and analysing linearly dependent patterns (*models*) in data sets. Data aggregation into linearly dependent subsets C_k can be combined in this approach with feature selection

1 Feature vectors and central points

Let us take into considerations the set C of m feature vectors $\mathbf{x}_j[n] = [x_{j1}, \dots, x_{jn}]^T$ belonging to a given n -dimensional feature space $F[n]$ ($\mathbf{x}_j[n] \in F[n]$):

$$C = \{\mathbf{x}_j[n]\}, \text{ where } j = 1, \dots, m \quad (1)$$

Components x_{ji} of the vector $\mathbf{x}_j[n]$ could be the numerical results of n standardized examinations of given objects O_j ($x_{ji} \in \{0,1\}$ or $x_{ji} \in R$).

Each vector $\mathbf{x}_j[n]$ can be treated as a point of the n -dimensional feature space $F[n]$.

In accordance with the K -means algorithm, feature vectors $\mathbf{x}_j[n]$ are divided into subsets C_k on the basis of actual central points (means) $\mathbf{w}_k[n] = [w_1, \dots, w_n]^T$ ($\mathbf{w}_k[n] \in R^n$):

$$\begin{aligned}
 & (\forall j \in \{1, \dots, m\}; \forall k' \in \{1, \dots, k\}) \\
 & \text{if } \rho(\mathbf{x}_j[n], \mathbf{w}'_k[n]) < \rho(\mathbf{x}_j[n], \mathbf{w}'_{k'}[n]) \quad (2) \\
 & \text{then } \mathbf{x}_j[n] \in C_k \text{ and } j \in J_k
 \end{aligned}$$

where $\rho(\mathbf{x}_j[n], \mathbf{w}_k[n])$ is the distance between the feature vector $\mathbf{x}_j[n]$ and the central point $\mathbf{w}_k[n]$, and J_k is the set of indices j of those vectors $\mathbf{x}_j[n]$ which have been allocated into the subset C_k .

The subsets C_k generated in accordance with the rule (2) allow to redefine new central points $\mathbf{w}'_k[n]$. The central points $\mathbf{w}'_k[n]$ of the subset C_k are computed through the minimization of the criterion function $Q_k(\mathbf{w}[n])$ defined on the elements $\mathbf{x}_j[n]$ of the subsets C_k :

$$Q_k(\mathbf{w}[n]) = \sum_{j \in J_k} \alpha_j \|\mathbf{x}_j[n] - \mathbf{w}[n]\| \quad (3)$$

where $\|\mathbf{x}_j[n] - \mathbf{w}[n]\|$ is the norm of the vector $\mathbf{x}_j[n] - \mathbf{w}[n]$ with the price α_j ($\alpha_j > 0$), and $\mathbf{w}'_k[n]$ is the minimum point of the function $Q_k(\mathbf{w}[n])$:

$$(\forall \mathbf{w}[n]) \quad Q_k(\mathbf{w}[n]) \geq Q_k(\mathbf{w}'_k[n]) \quad (4)$$

The new central points $\mathbf{w}'_k[n]$ allow to define new subsets C'_k in accordance with the rule (2). The K -means procedure stops when the difference between two successive central points $\mathbf{w}_k[n]$ and $\mathbf{w}'_k[n]$ is sufficiently small (in accordance with a given parameter $\varepsilon > 0$):

$$(\forall k \in \{1, \dots, K\}) \quad \|\mathbf{w}'_k[n] - \mathbf{w}_k[n]\| \leq \varepsilon \quad (5)$$

The minimization procedure of the function $Q_k(\mathbf{w}[n])$ (eq. 3) depends on the choice of the norm $\|\mathbf{x}_j[n] - \mathbf{w}[n]\|$. Commonly used is the Euclidean norm L_2 [3]:

$$\begin{aligned}
 & \|\mathbf{x}_j[n] - \mathbf{w}[n]\|_{L_2} = \\
 & = [(\mathbf{x}_j[n] - \mathbf{w}[n])^T (\mathbf{x}_j[n] - \mathbf{w}[n])]^{1/2} \quad (6)
 \end{aligned}$$

The L_1 and L_∞ norms are also used in the K -means algorithm:

$$\|\mathbf{x}_j[n] - \mathbf{w}[n]\|_{L_1} = \sum_{i=1, \dots, n} |x_{ji} - w_i| \quad (7)$$

$$\|\mathbf{x}_j[n] - \mathbf{w}[n]\|_{L_\infty} = \max_i |x_{ji} - w_i| \quad (8)$$

where $\mathbf{w}[n] = [w_1, \dots, w_n]^T$.

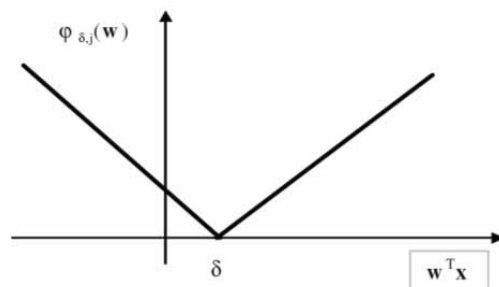


Figure 1. The penalty function $\varphi_j(\mathbf{w})$ (11)

The minimum point $\mathbf{w}'_k[n]$ (4) of the function $Q_k(\mathbf{w}[n])$ can be found analytically in the case the Euclidean norm L_2 . The criterion function $Q_k(\mathbf{w}[n])$ is convex and piecewise linear (CPL) in the case the norms L_1 (7) and L_∞ (8). The basis exchange algorithms allow one to find the minimum (4) in this case [3].

In accordance with the K -plans algorithm, feature vectors $\mathbf{x}_j[n]$ are divided into subsets C_k on the basis of actual central hyperplanes $H(\mathbf{w}_k, \theta_k)$:

$$H(\mathbf{w}_k, \theta_k) = \{\mathbf{x}[n]: \mathbf{w}_k[n]^T \mathbf{x}[n] = \theta_k\} \quad (9)$$

where the parameters $\mathbf{w}_k[n]$ and θ_k can be found through the minimization of the convex and piecewise linear (CPL) criterion function $\Phi_k(\mathbf{w}[n])$ (which is described later).

The distance $\rho_H(\mathbf{x}_j[n]; \mathbf{w}'_k[n], \theta_k)$ of the feature vectors $\mathbf{x}_j[n]$ from the hyperplanes $H(\mathbf{w}_k, \theta_k)$ can be computed in accordance with the following formula:

$$\begin{aligned}
 & \rho_H(\mathbf{x}_j[n]; \mathbf{w}'_k[n], \theta_k) = \\
 & = \left| \frac{\mathbf{w}_k[n]^T \mathbf{x}_j[n]}{\|\mathbf{w}_k[n]\|} - \theta_k \right| / \|\mathbf{w}_k[n]\| \quad (10)
 \end{aligned}$$

The distance function $\rho_H(\mathbf{x}_j[n]; \mathbf{w}'_k[n], \theta_k)$ can be used for the division of feature vectors $\mathbf{x}_j[n]$ into subsets C_k in accordance with the formula (2).

2 Convex and piecewise linear (CPL) criterion functions $\Phi_k(\mathbf{w}[n])$

Let us consider convex and piecewise linear (CPL) penalty functions $\varphi_j(\mathbf{w})$ defined on the feature vectors \mathbf{x}_j from the set C (1) [4]:

$$\begin{aligned}
 & (\forall \mathbf{x}_j[n] \in C) \\
 & \varphi_j(\mathbf{w}) = \begin{cases} \delta - \mathbf{w}[n]^T \mathbf{x}_j[n] & \mathbf{w}[n]^T \mathbf{x}_j[n] \leq \delta \\ \mathbf{w}[n]^T \mathbf{x}_j[n] - \delta & \mathbf{w}[n]^T \mathbf{x}_j[n] > \delta \end{cases} \quad (11)
 \end{aligned}$$

where δ is some parameter (margin) ($\delta > 0$).

The penalty functions $\varphi_j(\mathbf{w})$ are equal to the absolute values $|\delta - \mathbf{w}[n]^T \mathbf{x}_j[n]|$ (Figure 1).

The criterion function $\Phi_k(\mathbf{w}[n])$ is defined as the weighted sum of the penalty functions $\varphi_j(\mathbf{w}[n])$ (11) related to the vectors $\mathbf{x}_j[n]$ from the subset C_k :

$$\Phi_k(\mathbf{w}[n]) = \sum_{j \in J_k} \alpha_j \varphi_j(\mathbf{w}[n]); \quad \alpha_j > 0 \quad (12)$$

The positive parameters α_j in the function $\Phi_k(\mathbf{w}[n])$ can be treated as the prices of particular vectors \mathbf{x}_j .

The criterion function $\Phi_k(\mathbf{w}[n])$ (12) is convex and piecewise linear as the sums of such type of functions $\alpha_j \varphi_j(\mathbf{w}[n])$.

Each feature vector $\mathbf{x}_j[n]$ from the set C defines the hyperplane h_j in the parameter (weight) space R^n :

$$(\forall \mathbf{x}_j[n] \in C) \quad h_j = \{\mathbf{w}[n]: \mathbf{x}_j[n]^T \mathbf{w}[n] = \delta\} \quad (13)$$

The hyperplanes h_j (13) are linked to the penalty functions $\varphi_j(\mathbf{w}[n])$ (11). The function $\varphi_j(\mathbf{w})$ is equal to zero if and only if, the vector $\mathbf{w}[n]$ is situated on the hyperplane h_j .

Any set of n linearly independent feature vectors $\mathbf{x}_j[n]$ ($j \in J_k$) can be used for designing the non-singular matrix $\mathbf{B}_k[n] = [\mathbf{x}_{j(1)}, \dots, \mathbf{x}_{j(n)}]$ with the columns composed from these vectors.

The non-singular matrix $\mathbf{B}_k[n]$ is called the k -th basis of the feature space $F[n]$. The vectors $\mathbf{x}_{j(n)}$ ($j \in J_k$) from this set define those n hyperplanes h_j which pass through the below point (*vertex*) $\mathbf{w}_k[n]$:

$$\begin{aligned} \mathbf{B}_k[n]^T \mathbf{w}_k[n] &= \boldsymbol{\delta}[n] = [\delta, \dots, \delta]^T = \\ &= \delta [1, \dots, 1]^T = \delta \mathbf{1}[n] \end{aligned} \quad (14)$$

or

$$\mathbf{w}_k[n] = (\mathbf{B}_k[n]^T)^{-1} \boldsymbol{\delta}[n] = \delta (\mathbf{B}_k[n]^T)^{-1} \mathbf{1}[n] \quad (15)$$

In the case of “short” vectors $\mathbf{x}_j[n]$, when the number m of the vectors $\mathbf{x}_j[n]$ is much greater than the vectors’ dimensionality n ($m \gg n$), there may exist many bases $\mathbf{B}_k[n]$ (14) and many vertices $\mathbf{w}_k[n]$ (15).

It can be proved that the minimal value Φ_k^* of the criterion functions $\Phi_k(\mathbf{w}[n])$ is situated in one of the vertices $\mathbf{w}_k[n]$ [6]:

$$\begin{aligned} (\exists \mathbf{w}_k^*[n]; \mathbf{w}[n]) \\ \Phi_k(\mathbf{w}[n]) \geq \Phi_k(\mathbf{w}_k^*[n]) = \Phi_k^* \end{aligned} \quad (16)$$

The optimal parameter vector $\mathbf{w}_k^*[n]$ is used for the definition of the below hyperplane $H(\mathbf{w}_k^*[n], \delta)$ (Eq. 9) in the feature space $F[n]$:

$$H(\mathbf{w}_k^*[n], \delta) = \{\mathbf{x}[n]: \mathbf{w}_k^*[n]^T \mathbf{x}[n] = \delta\} \quad (17)$$

Theorem 1.

The minimal value $\Phi_k(\mathbf{w}_k^*[n])$ (16) of the criterion function $\Phi_k(\mathbf{w}[n])$ (12) with $\delta \neq 0$ is equal to zero ($\Phi_k(\mathbf{w}_k^*[n]) = 0$), if and only if all the feature vectors $\mathbf{x}_j[n]$ from the subset C_k are situated on some hyperplane $H(\mathbf{w}[n], \theta)$ (Eq. 9), with $\theta \neq 0$.

Proof: Let assume that the feature vectors $\mathbf{x}_j[n]$ from the subset C_k are situated on some hyperplane $H(\mathbf{w}[n], \theta)$ with $\theta \neq 0$. In this case, the following equations are fulfilled:

$$\begin{aligned} (\forall \mathbf{x}_j[n] \in C) \\ \mathbf{w}[n]^T \mathbf{x}_j[n] = \theta \text{ or } (\delta/\theta) \mathbf{w}[n]^T \mathbf{x}_j[n] = \delta \end{aligned} \quad (18)$$

thus

$$(\forall \mathbf{x}_j[n] \in C_k) \quad \varphi_j((\delta/\theta) \mathbf{w}[n]) = 0 \quad (19)$$

On the other hand, if the conditions $\varphi_j(\mathbf{w}[n]) = 0$ (8) are fulfilled for all the feature vectors \mathbf{x}_j from the subset C_k , then these vectors have to be situated on the hyperplane $H(\mathbf{w}[n], \theta)$.

If all the feature vectors $\mathbf{x}_j[n]$ from the subset C_k are situated on some hyperplane $H(\mathbf{w}[n], \theta)$ with $\theta = 0$, then the minimal value $\Phi_k(\mathbf{w}_k^*[n])$ of the criterion function $\Phi_k(\mathbf{w}[n])$ is equal to zero ($\Phi_k(\mathbf{w}_k^*[n]) = 0$) only if $\delta = 0$. \square

It has been proved that the minimal value $\Phi_k(\mathbf{w}_k^*[n])$ of the criterion function $\Phi_k(\mathbf{w}[n])$ does not depend on linear, non-singular data transformations (the *invariance property*) [4]:

$$\Phi'(\mathbf{w}'_k[n]) = \Phi(\mathbf{w}_k^*[n]) \quad (20)$$

where $\Phi'(\mathbf{w}'_k[n])$ is the minimal value (16) of the criterion functions $\Phi'(\mathbf{w}[n])$ (12) defined on the transformed feature vectors $\mathbf{x}'_j[n]$:

$$(\forall \mathbf{x}_j[n] \in C_k) \quad \mathbf{x}'_j[n] = \mathbf{A}[n] \mathbf{x}_j[n] \quad (21)$$

where $\mathbf{A}[n]$ is a non-singular matrix of dimension $(n \times n)$ ($\mathbf{A}^{-1}[n]$ exists).

The minimal value $\Phi_k(\mathbf{w}_k^*[n])$ of the criterion function $\Phi_k(\mathbf{w}[n])$ defined on the centred vectors $\mathbf{x}'_i[n] = \mathbf{x}_i[n] - \mathbf{m}_k[n]$ does not depend on translations $\mathbf{x}'_j[n] + \mathbf{b}[n]$ of the centred vectors $\mathbf{x}'_j[n]$, where $\mathbf{b}[n]$ is an arbitrary vector and $\mathbf{m}_k[n]$ is the mean vector in the subset C_k .

The minimal value $\Phi_k(\mathbf{w}_k^*[n])$ of the criterion function $\Phi_k(\mathbf{w}[n])$ is characterised by two below monotonicity properties:

Property I (*monotonicity with respect to reduction of the subset C_k*): Reducing the subset C_k to C'_k by neglecting some feature vectors $\mathbf{x}_j[n]$ can not result in an increase of the minimal value $\Phi_k(\mathbf{w}_k^*[n])$ of the criterion function $\Phi_k(\mathbf{w}[n])$:

$$(C'_k \subset C_k) \Rightarrow (\Phi_k^* \leq \Phi_k^*) \quad (22)$$

where the symbol Φ_k^* means the minimal value (16) of the criterion function $\Phi_k(\mathbf{w}[n])$ defined on the elements $\mathbf{x}_j[n]$ of the subset C'_k .

The relation (22) can be justified by the remark that neglecting some feature vectors $\mathbf{x}_j[n]$ results in neglecting some non-negative components $\varphi_j(\mathbf{w}[n])$ in the criterion function $\Phi_k(\mathbf{w}[n])$.

Property II (*monotonicity with respect to reduction of the feature space $F[n]$*): Reducing the feature space $F[n]$ to $F'[n']$ by neglecting some features x_i can not result in a decrease of the minimal value $\Phi_k(\mathbf{w}_k^*[n])$ of the criterion function $\Phi_k(\mathbf{w}[n])$:

$$(F'[n'] \subset F[n]) \Rightarrow (\Phi_k^* \geq \Phi_k^*) \quad (23)$$

where the symbol Φ_k^* means the minimal value (16) of the criterion function $\Phi_k(\mathbf{w}[n])$ defined on the vectors $\mathbf{x}_j[n']$ from the feature space $F'[n']$.

The relation (23) results from the fact that the neglecting of some features x_i is equivalent to imposing an additional constraints in the form of the condition $w_i = 0$ on the parameter space R^n .

The monotonicity properties (22) and (23) constitute the basis for the proposed procedure of hidden linear dependencies extracting from data set.

3 CPL criterion functions $\Psi_k(\mathbf{w})$ with feature costs

Reduction of unimportant features x_i in the cost sensitive manner can be supported by the modified CPL criterion function $\Psi_k(\mathbf{w}[n])$ in the below form [4]:

$$\Psi_k(\mathbf{w}[n]) = \Phi_k(\mathbf{w}[n]) + \lambda \sum_{i \in I} \gamma_i \phi_i(\mathbf{w}[n]) \quad (24)$$

where $\Phi_k(\mathbf{w}[n])$ is given by (12), λ is the feature cost level ($\lambda \geq 0$), γ_i is the cost of the feature x_i ($\gamma_i > 0$), $I = \{1, \dots, n\}$, and the cost functions $\phi_i(\mathbf{w})$ are defined by the unit vectors $\mathbf{e}_i[n] = [0, \dots, 1, \dots, 0]^T$:

$$\begin{aligned} (\forall i \in \{1, \dots, n\}) \phi_i(\mathbf{w}[n]) &= |w_i| = \\ &= \begin{cases} -\mathbf{e}_i[n]^T \mathbf{w}[n] & \mathbf{e}_i[n]^T \mathbf{w}[n] < 0 \\ \mathbf{e}_i[n]^T \mathbf{w}[n] & \mathbf{e}_i[n]^T \mathbf{w}[n] \geq 0 \end{cases} \quad (25) \end{aligned}$$

The criterion function $\Psi_k(\mathbf{w}[n])$ is the convex and piecewise linear (CPL) as the sum of the CPL functions $\Phi_k(\mathbf{w}[n])$ (12) and $\lambda \gamma_i \phi_i(\mathbf{w}[n])$. The optimal point $\mathbf{w}'_\lambda[n]$ constitutes the minimal value of the criterion function $\Psi_k(\mathbf{w}[n])$:

$$(\exists \mathbf{w}'_\lambda[n]; \forall \mathbf{w}[n]) \Psi_k(\mathbf{w}[n]) \geq \Psi_k(\mathbf{w}'_\lambda[n]) \quad (26)$$

Each CPL cost function $\phi_i(\mathbf{w}[n])$ tends to reach the condition $w_i = 0$ (24) through the minimization of the function $\Psi_k(\mathbf{w}[n])$ and to reducing the feature x_i . The influence of the cost functions $\phi_i(\mathbf{w}[n])$ increases with the value of the parameter λ . The increase of the cost level λ can lead to reducing additional features x_i .

Each unit vector $\mathbf{e}_i[n]$ defines the below hyperplane $h_{0,i}$ in the parameter space R^n :

$$h_{0,i} = \{\mathbf{w}[n]: \mathbf{e}_i[n]^T \mathbf{w}[n] = 0\} \quad (27)$$

The minimum point $\mathbf{w}'_\lambda[n]$ of the function $\Psi_k(\mathbf{w}[n])$ is situated in one of the vertices $\mathbf{w}'_k[n]$ ($\mathbf{w}'_\lambda[n] = \mathbf{w}'_k[n]$) defined by the equation of the below type (14):

$$\mathbf{B}_k[n]^T \mathbf{w}'_k[n] = \boldsymbol{\delta}'[n] = [\delta, \dots, \delta, 0, \dots, 0]^T \quad (28)$$

In this case, the columns of the matrix $\mathbf{B}_k[n]$ can be composed partly of some feature vectors $\mathbf{x}_j[n]$ and partly of some unit vectors $\mathbf{e}_i[n]$. The vertex $\mathbf{w}'_k[n]$ (15) is the point of intersection of hyperplanes h_j (13) defined by some feature vectors $\mathbf{x}_j[n]$ and hyperplanes $h_{0,i}$ (27) defined by unit vectors $\mathbf{e}_i[n]$. The minimum point $\mathbf{w}'_\lambda[n]$ of the function $\Psi_k(\mathbf{w}[n])$ (24) is situated in one of such vertices $\mathbf{w}'_k[n]$, which is the intersection point of n hyperplanes h_j and $h_{0,i}$.

The features x_i , which are linked to the unit vectors $\mathbf{e}_i[n]$ in the optimal basis $\mathbf{B}_k[n]$ fulfil the below equation and can be reduced without changing of the minimal value $\Psi_k(\mathbf{w}'_\lambda[n])$:

$$\begin{aligned} \mathbf{e}_i[n]^T \mathbf{w}'_k[n] = 0 &\Rightarrow w_{ki} = 0 \Rightarrow \\ &\Rightarrow \text{feature } x_i \text{ is reduced} \end{aligned} \quad (29)$$

where $\mathbf{e}_i[n] = [0, \dots, 0, 1, 0, \dots, 0]^T$ is the i -th unit vector, and $\mathbf{w}'_k[n] = [w_{k1}, \dots, w_{kn}]$.

The number of the reduced features x_i can be increased by an increasing the feature cost level λ in the criterion function $\Psi_\lambda(\mathbf{w}[n])$.

4 Extracting hidden linear dependencies

The procedure of hidden linear dependencies extracting from data set \mathcal{C} (1) can be based on the CPL criterion functions $\Phi_k(\mathbf{w}[n])$ (12) and $\Psi_k(\mathbf{w}[n])$ (24). The monotonicity properties (22) and (23) are particularly important in the proposed procedure. These monotonicity properties (22) and (23) are valid not only for the minimal value $\Phi_k(\mathbf{w}'_k[n])$ (16) of the function $\Phi_k(\mathbf{w}[n])$ but also for the minimal value $\Psi_k(\mathbf{w}'_\lambda[n])$ (26) of the function $\Psi_k(\mathbf{w}[n])$.

Algorithm 1. The multistage procedure of extracting of hidden linear dependency is described by the below successive steps:

1. Two small, positive parameters (margins of precision) τ_1 and τ_2 are defined ($\tau_2 \geq \tau_1 > 0$), the value $k = 1$ and the initial set $C'_k = \mathcal{C}$ (1) of all the feature vectors $\mathbf{x}_j[n]$ are fixed.
2. There is the computed minimal value $\Phi_k(\mathbf{w}'_k[n])$ (16) of the criterion function $\Phi_k(\mathbf{w}[n])$ (12). The function $\Phi_k(\mathbf{w}[n])$ is defined on all the feature vectors $\mathbf{x}_j[n]$ from the set C'_k .
3. The minimal number of the feature vectors $\mathbf{x}_j[n]$ is omitted from the set C'_k in order to reach the condition $\Phi_k(\mathbf{w}'_k[n]) \leq \tau_1$. Such vectors $\mathbf{x}_j[n]$ are reduced which caused the smallest increase of the value $\Phi_k(\mathbf{w}'_k[n])$. The remaining feature vectors $\mathbf{x}_j[n]$ form the k -th *linearly dependent cluster* C_k .
4. The maximal number of the features x_i is omitted from the feature space $F[n]$, while preserving the condition $\Phi'_k(\mathbf{w}'_k[n'])$ in the new feature subspace $F'[n']$ ($F'[n'] \subset F[n]$). The vector $\mathbf{w}'_k[n']$ constitutes the minimum (26) of the function $\Psi_k(\mathbf{w}[n])$

The dimensionality of the feature vectors $\mathbf{x}_j[n]$ from the cluster C_k is reduced from n to n' by a successive increase of feature cost level λ (24).

5. The below linear relation between features x_i from the feature subspace $F'[n']$ ($x_i \in F'[n']$) is formed on this basis:

$$w'_{\lambda,i(1)}x_{j,i(1)} + \dots + w'_{\lambda,i(n')}x_{j,i(n')} = \delta \quad (30)$$

where $\mathbf{x}_j[n'] = [x_{j,i(1)}, \dots, x_{j,i(n')}]^T$ is the feature vector ($\mathbf{x}_j[n'] \in F'[n']$) reduced during the previous stage, and $\mathbf{w}'_\lambda[n'] = [w'_{\lambda,i(1)}, \dots, w'_{\lambda,i(n')}]^T$ is the optimal vector with all the components $w'_{\lambda,i}$ different from zero ($w'_{\lambda,i} > 0$).

6. The set C'_k is reduced by neglecting such feature vectors $\mathbf{x}_j[n]$ which constitute linearly dependent cluster C_k . If the set C'_k is not empty ($C'_k \neq \emptyset$), then the value of the parameter k is increased by one ($k \rightarrow k + 1$) and the next stage is started from the step 2.

The above procedure allows to extract K linearly dependent clusters C_k from the data set \mathcal{C} . Each cluster C_k is represented by K linear relations (30). As opposed to the K -means algorithm, the number K is not fixed at the beginning of this procedure. The number K of the clusters C_k reflects the structure of the data set \mathcal{C} . Let us remark that the relation (30) allows to form n' regression type models. Each component $x_{j,i(k)}$ can represent dependent variable (feature) $x_{i(k)}$ and the remaining $n' - 1$ components $x_{j,i(k')}$ can represent dependent variables. Such regression type models have local properties. This means that each model represents feature vectors $\mathbf{x}_j[n]$ from one particular cluster C_k .

5 Concluding remarks

The problem of extracting linearly dependent patterns from data sets is considered in the paper. The proposed approach is based on the minimization of two convex and piecewise linear (CPL) criterion functions $\Phi_k(\mathbf{w}[n])$ and $\Psi_k(\mathbf{w}[n])$.

Extraction of hidden, linearly dependent patterns is considered here as a problem of cluster analysis. The K -plans algorithm, similarly to the K -means algorithm has been proposed as one of the tools for solving this problem. In this approach each linearly dependent cluster C_k is represented by some central hyperplane $H(\mathbf{w}_k[n], \theta_k)$.

The alternative approach is based on what is described in Section 4 as a sequence of the functions $\Phi_k(\mathbf{w}[n])$ and $\Psi_k(\mathbf{w}[n])$ minimizations. In this approach there is no need to fix the number K of clusters C_k beforehand. The extraction of linearly dependent clusters C_k is linked here to designing the regression type models (30). As a result, the data set C is represented by the family of K linear models (30). Such representation allows to expose the internal linear structure hidden in the set C .

The usefulness of the extracted linearly dependent patterns and models should be verified in many ways. In accordance with data mining or exploratory analysis standards, experts in the fields should have the final judgments concerning the extracted patterns.

References

- [1] Hand D., Smyt P., Mannila H. *Principles of Data Mining*, MIT Press, Cambridge, MA 2001
- [2] Duda O.R., Hart P.E., Stork D.G. *Pattern Classification*, J. Wiley, New York, 2001.
- [3] Bobrowski L., Bezdek J.C. *C-means clustering with the L_1 and L_∞ norms*, IEEE Transactions on Systems Man and Cybernetics, Vol. 21(3), pp. 545-554, 1991.
- [4] Bobrowski L. *CPL clustering with feature costs*, ICDM2008, Leipzig, Germany
- [5] Bobrowski L. *Design of piecewise linear classifiers from formal neurons by some basis exchange technique*. Pattern Recognition, 24(9), pp. 863-870, 1991
- [6] Bobrowski L. *Eksploracja danych oparta na wypukłych i odcinkowo-liniowych funkcjach kryterialnych (Data mining based on convex and piecewise linear (CPL) criterion functions)* (in Polish), Technical University Białystok, 2005

Corresponding author: L. Bobrowski
Faculty of Computer Science
Białystok Technical University
ul. Wiejska 45A, 15-351 Białystok, Poland
leon@ibib.waw.pl

Received & Accepted: MATHMOD 2009 -

Revised: September 10, 2009 -

Accepted: July 10, 2010 -

The Proposal of the Alternately Evolving Genetic Algorithm and its Application

Zhanghui Chen, Kangrui Zhou, Yingtao Liu, Wenbin Zhan
Huazhong University of Science and Technology, China

SNE Simulation Notes Europe SNE 20(3-4), 2010, 49-54, doi: 10.11128/sne.20.tn.09993

In view of the shortcomings of the usual genetic algorithm when solving multi-objective combinatorial optimization problems, the paper proposes an alternately evolving genetic algorithm. It adopts alternate strategy and optimizes multiple objectives one by one circularly. The solving results of the Sudoku puzzle indicate that this strategy can make the excellent patterns of different objectives all grow rapidly, and the results' comparison verify its feasibility and excellence in the general convergence. The sensitivity of the algorithm's parameter is also analyzed.

Introduction

A genetic algorithm (GA) is a random searching algorithm based on biological evolution and natural selection. It follows the "survival of the fittest" principles of the Darwinian evolution to implement probabilistic optimization of particular objectives. Due to its parallelism and globally searching ability, GA has been widely applied to function optimization, combinatorial optimization, artificial intelligence and some other fields. Especially in some multi-objective combinatorial optimization problems, GA is usually better than other algorithms.

When dealing with multi-objective combinatorial optimization problems, the usual genetic algorithm usually employs a comprehensive weighed objective

to replace multiple objectives through constructing an evaluation function, and then uses unified genetic operators to optimize the weighted objective in a unified coding scheme. This method is effective in many cases. But for some combinatorial problems, the objectives may conflict with one another in the evolutionary process. That is to say, when one target gets met, others may deviate from the requirements very much. In these cases, the efficiency of the general evolution will be very low and the evolutionary process may be trapped in a state of random wander. By way of contrast, this paper puts forward an alternately evolving genetic algorithm from the opposite direction, which adopts alternate strategy to optimize multiple objectives one by one circularly. The solving results of the Sudoku puzzle verify the strategy's reasonableness and excellence.

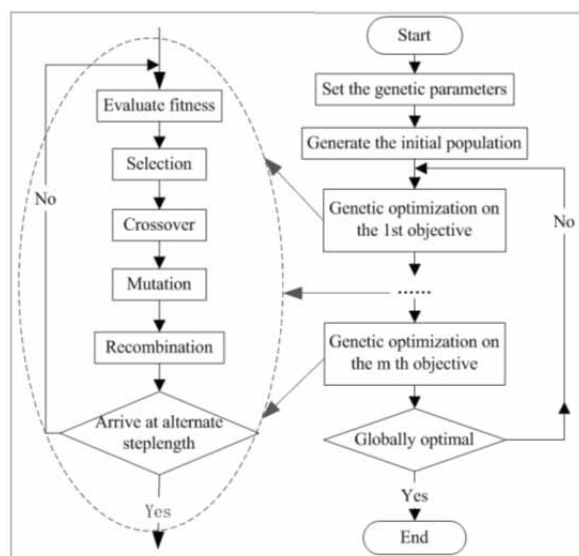


Figure 1. The flow char of alternately evolving GA.

1 The proposal of the alternately evolving genetic algorithm

1.1 Multi-objective combinatorial optimization

Multi-objective combinatorial optimization refers to the optimization of multiple objective functions of combinatorial variables in order to achieve the globally optimal solution or satisfactory solution. It is widely used in many fields, such as the university timetable, train schedules, Latin square, Sudoku and some other scheduling problems. Generally it can be modelled by the following expression:

$$\begin{aligned} & \min f_1(X), \min f_2(X), \dots, \min f_m(X) \\ & \text{s. t. } \begin{cases} g_1(X) \leq b_1 \\ g_2(X) \leq b_2 \\ \vdots \\ g_n(X) \leq b_n \end{cases} \end{aligned} \quad (1)$$

Where $X = (x_1, x_2, \dots, x_n)$ are combinational variables; $f_k(X)$ is the k -th objective function; and $g_i(X)$ is the i -th combinational constraint condition.

1.2 The proposal of the alternately evolving genetic algorithm

It can be seen from formula (1) that multi-objective combinatorial optimization problems will become very complex with the increase of the number of objective functions and constraint conditions. It has been proved in theory that most of combinatorial optimization problems belong to NP complete problem, whose searching space will increase exponentially and even explode as the problem scale increases. The NP complete character of combinatorial optimization and the complexity of multi-objective optimization cause many classical algorithms not applicable any more. But GA can afford very well for its good parallelism and heuristic, and has been widely applied to most of combinatorial optimization problems. When solving these problems, GA usually changes multiple objectives into a comprehensive weighed objective, namely:

$$\min a_1 f_1(X) + a_2 f_2(X) + \dots + a_m f_m(X) \quad (2)$$

where a_k ($k = 1, 2, \dots, m$) refers to the weighted coefficient of the k -th objective function.

By doing this, it then adopts a unified coding scheme to transform solution space into genetic space, and then promote individuals to a better state by a unified fitness evaluating operator, selection operator, crossover and mutation operator. This method that changes multiple objectives into a single objective is not applicable to some cases, such as problems with conflicting objectives. If we employ a comprehensive objective to evolve these problems, the evolutionary process may enter a state of random wander because of conflicts. From the point of view of Pattern Theorem, different objectives have different excellent patterns in a unified coding scheme.

But the genetic operators especially the crossover operator can only consider patterns of one objective, and make the objective's excellent patterns grow exponentially, while other objectives stay in a state of random evolution.

Consequently, the overall evolutionary efficiency will be very low and multiple objectives cannot get rapidly optimized at the same time.

In view of the shortcomings of the usual GA, the paper proposes an alternately evolving strategy from another standpoint. It adopts alternate method to optimize multiple objectives one by one circularly. Take the problem in formula 1 for example, and its detailed implementing steps are the following:

1. Take $f_1(X)$ as the only objective function to evolve, and turn to step 2 when the evolutionary process arrives at a certain depth.
2. Take $f_2(X)$ as the only objective function to evolve, and turn to step 3 when the evolutionary process arrives at a certain depth.
- ...
- m . Take $f_m(X)$ as the only objective function to evolve, and turn to step $m + 1$ when the evolutionary process arrives at a certain depth.
- $m + 1$. Determine whether the m objective functions all achieve optimum or meet requirement. If so, terminate the procedure; if not, turn to step 1 and start the next alternate evolution.

In the above evolutionary process, every goal's evolution can have its own coding scheme and genetic operators. And it may be needed to transform coding scheme when the step turn to the next one. In order to describe each step's evolutionary depth, we can define alternate step length S_k , which refers to the iteration times when the k -th objective evolves independently in the step k . Similarly, we can define the process of evolving all the m objectives one time as an alternate cycle.

The advantage in the foregoing alternate strategy is that every step is single-objective optimization, and when proper genetic operators are adopted for every objective, the convergent speed will be very fast. But every step doesn't consider other objectives, so these objectives' evolution will be in a random state in the step; that is, may converge or may diverge. In order to make the overall evolutionary trend be convergent, it is very important to control each step's evolutionary depth S_k . If evolve too deeply, the divergent trend will be greater than the convergent one; but if too shallowly, the evolutionary process will be very slow and be similar to the usual GA. Therefore, it is needed to adjust S_k enough times to search for a better one. The flow char of alternately evolving GA is shown in Figure 1.

	2	4	8	6	1			
1	7			2				5
7		1			3		8	
9				5				4
	8		9			5		1
3				8			1	7
			6	9	2	4	5	

Figure 2. 9 × 9 grid

2 Experiments

In order to verify the feasibility and high efficiency of the alternately evolving GA, take Sudoku as an example which the algorithm is used to solve. Sudoku is a numbers game popular in Japan, the UK and the USA. In essence, it is a special kind of Latin square and can be modelled by multi-objective combination optimization.

Because of Latin square’s NP complete character, the solution space of Sudoku is quite huge, which causes many conventional algorithms to be ineffective. In the following paragraphs, we firstly give a brief introduction to Sudoku, and then adopt the usual GA and the alternately evolving GA to solve it respectively, and verify the excellence of the alternately evolving GA through the experimental results.

2.1 Sudoku

The most common Sudoku puzzle consists of 9 × 9 grid and 3 × 3 blocks for a total of 81 cells. One example of 9 × 9 Sudoku puzzle is shown in Figure 2, in which different blocks are marked up by different colours.

When a puzzle with a set of pre-filled numbers is designed, the task is to place the numbers 1 through 9 in each cell, such that the following rules hold:

1. Constraints on rows: each row must contain the numbers 1 – 9 once and only once;
2. Constraints on columns: each column must contain the numbers 1 – 9 once and only once;
3. Constraints on blocks: each block must contain the numbers 1 – 9 once and only once.

As can be seen, the objective of the puzzle is to make the nine groups of the numbers 1 – 9, totally 81 digits, rationally arrange on the 9 × 9 board, such that each row, column and block have no repeated digits.

2.2 The specific design of the alternately evolving GA for Sudoku

Considering Sudoku’s characteristics, the following algorithm can be designed to solve it.

The coding scheme

Adopt the symbolic matrix coding scheme, which directly maps the 9 × 9 grid into a 9 × 9 matrix shown as follows: The matrix element x_{ij} refers to the number filled in the row i and column j of the 9 × 9 grid.

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \cdots & x_{19} \\ x_{21} & x_{22} & x_{23} & \cdots & x_{29} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{91} & x_{92} & x_{93} & \cdots & x_{99} \end{bmatrix} \quad (3)$$

The comprehensive objective function

For each individual in the population, its fitness can be obtained in the following way:

1. Determine the candidate set of every empty cell on the 9 × 9 board.
2. For each individual, each cell’s value can be determined this way: if the cell has been pre-filled, then its value is the pre-filled one; if the cell is empty, its value must be randomly chosen from its candidate set. By doing this, all the cells’ values can be determined and an initial individual is generated.
3. Repeat step 2 to generate more individuals until the initial population has been yielded.

Optimization on rows

1. *Fitness function.* The optimization on rows means only considering the constraints on rows without taking into account the other two types of constraints. The optimization’s goal is to meet the rows’ constraints as much as possible. Therefore, the fitness function is:

$$f_1 = 1 / \sum_{i=1}^9 m_i \quad (5)$$

where $\sum_{i=1}^9 m_i$ refers to the total punishment on rows.

2. *Selection operator.* Adopt the combined strategy of roulette wheel selection and preserving the fittest individuals to the next generation.

3. *Crossover operator.* Adopt one-point crossover, which treats each row of two-dimensional matrix as a point, and treats multi-row as multi-point, and then randomly determines a point to implement crossover operation.
4. *Mutation operator.* Adopt simple mutation and the mutated cell's new value is randomly generated from its candidate set.

Optimization on columns

Compared with optimization on rows, optimization on columns means only considering the constraints on columns without taking into account the other two types of constraints. The optimization's goal is to meet the columns' constraints as much as possible. Its fitness function is:

$$f_2 = 1 / \sum_{j=1}^9 n_j \tag{6}$$

where $\sum_{j=1}^9 n_j$ refers to the total punishment on the columns.

The genetic operators of the columns' optimization are similar to the rows'. Simply, it only needs to treat each column of two-dimensional matrix as a point when implementing crossover.

Optimization on blocks

Compared with the two optimizations above, the fitness function of optimization on blocks is:

$$f = 1 / \sum_{k=1}^9 l_k \tag{7}$$

where $\sum_{k=1}^9 l_k$ refers to the total punishment on the blocks.

The genetic operators of blocks' optimization are similar to the foregoing. Simply, it only needs to treat each block as a point and line up these points when implementing crossover.

Genetic parameters

All of the genetic parameters need debugging repeatedly to obtain proper values. Through many experiments, some parameters are set as follows:

Number of the population's individuals	300
Preserving probability of the fittest individuals	0.10
Crossover probability	0.85
Mutation probability	0.03
Number of possible mutated points	81
Three kinds of alternate step length for the three objectives	10, 10, 10

5	2	4	8	6	1	9	7	3
1	7	3	4	2	9	8	6	5
6	9	8	7	3	5	1	4	2
7	5	1	2	4	3	6	8	9
9	3	6	1	5	8	7	2	4
4	8	2	9	7	6	5	3	1
2	4	5	3	1	7	3	9	6
3	6	9	4	8	5	2	1	7
8	1	7	6	9	2	4	5	8

Figure 3. The result of the usual GA. The cells marked up by other colours indicate there are repeated numbers.

5	2	4	8	6	1	3	7	9
1	7	3	4	2	9	8	6	5
6	9	8	7	3	5	1	4	2
7	5	1	2	4	3	9	8	6
9	3	6	1	5	8	7	2	4
4	8	2	9	7	6	5	3	1
2	4	5	3	1	7	6	9	8
3	6	9	5	8	4	2	1	7
8	1	7	6	9	2	4	5	3

Figure 4. The result of the alternately evolving GA.

2.3 Experiment results and their comparison

For the Sudoku puzzle in Figure 2, adopt the usual GA and the alternately evolving GA to solve it respectively. Their solving results are shown in Figure 3.

By comparing Figures 3 – 6, it can be known that:

1. In the solving process of the alternately evolving GA, although every evolution's alternation always makes punishment fluctuate, the general trend of the three kinds of punishment is convergent and the population will evolve to optimal state in the end when all of the genetic parameters are set properly. These show that the alternately evolving GA is feasible.
2. The evolution of the usual GA falls into local optimum at about 30 generations, while the one of the alternately evolving GA can converge to global optimum. Therefore, the convergent capability of the alternately evolving GA is better than the one of the usual GA.

3. The alternately evolving GA need to evolve every objective one by one circularly, so for some simple or small-scale problems, its convergent time will be longer than the usual GA's. But for some complex and large-scale problems, the usual GA always converges slowly and even falls into local optimum, so at this moment the advantage of the alternately evolving GA in convergent time will present gradually as problems' complexity increases.

4. Sudoku is a typical multi-objective combinatorial optimization problem. It can be predicted that the alternately evolving GA is also effective for other multi-objective combinatorial optimization problems.

2.4 Sensitivity analysis

To confirm the effect of each objective's evolutionary depth to the general convergent performance, set step length to 3 and 30 respectively and other parameters keep constant. The results of the two kinds of experiment are shown in Figures 7 and 8.

As can be seen from the two figures, when the three kinds of alternate steplength are all set to 3, each objective's evolutionary degree is so shallow that the general evolution stays in a state of random shake, and the general performance is similar to the usual GA's.

When the three kinds of alternate steplength are all set to 30, each objective's evolutionary degree is so deep that the population may fall into local optimum at the first evolution and all of individuals may be exactly the same, and the

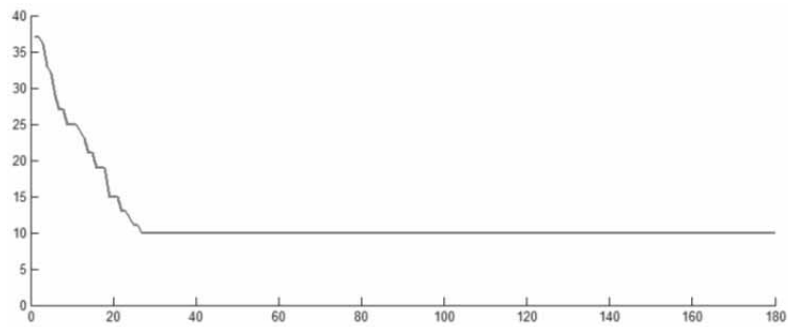


Figure 5. The comprehensive punishment's change in the solving process of the usual GA. The graph's abscissa refers to evolving generations, and the ordinate refers to comprehensive punishment. It can be seen from the graph that the population's evolution falls into local optimum at about 30 generations.

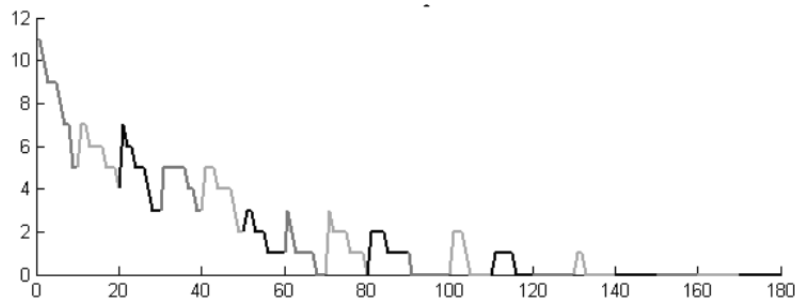


Figure 6. The change of each objective's punishment in the solving process of the alternately evolving GA. The graph's abscissa refers to evolving generations, and the ordinate refers to row objective's punishment, column objective's punishment or block objective's punishment. The curve's fluctuation in the graph is the very result of alternate evolution, in which the red curve represents the row objective's evolution and its corresponding ordinate refers to row objective's punishment; the green curve represents the column objective's evolution and its corresponding ordinate refers to column objective's punishment; the black curve represents the block objective's evolution and its corresponding ordinate refers to block objective's punishment. As can be seen from the graph, although every evolution's alternation always makes punishment fluctuate, the general trend of the three kinds of punishment is convergent. And when the three kinds of alternate step length are all set to 10, the three kinds of punishment all converge to 0 at about 150 generations, that is to say, the population arrives at optimum.

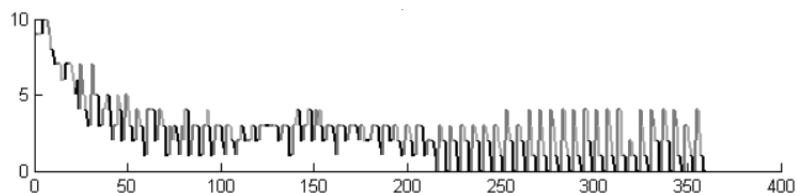


Figure 7. The change of each objective's punishment in the solving process of the alternately evolving GA when the three kinds of alternate steplength are all set to 3.

later evolution is a uniform shake. These indicate that each objective's evolutionary depth has a great effect to the general convergent performance, so it is necessary to experiment enough times to find the best steplength.

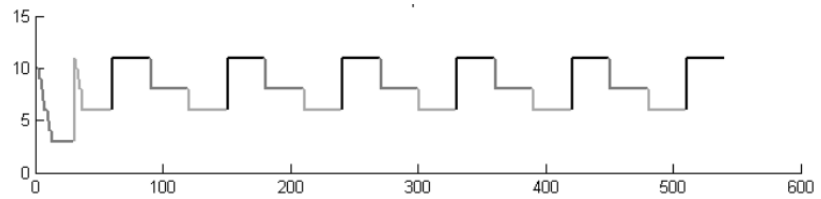


Figure 8. The change of each objective's punishment in the solving process of the alternately evolving GA when the three kinds of alternate steplength are all set to 30.

3 Conclusions

This paper considering the shortcomings of the usual genetic algorithm when solving multi-objective combinatorial optimization problems, proposes an alternately evolving strategy.

The solving results of the Sudoku puzzle indicate that the alternately evolving GA's general convergent capability is better than the usual GA's, and the advantage of the former in convergent speed will present gradually as problems' complexity increases. These show that the alternately evolving GA is feasible and excellent.

The alternately evolving GA adopted in solving Sudoku, is just an easy model of the alternately evolving GA, and there is much room for improvement, such as:

1. Adopt adaptive steplengths, which can change with objectives and evolving state;
2. Adopt different coding schemes and genetic operators for different objectives.

These measures can improve general convergent capability and speed.

References

- [1] Timo Mantere, Janne Koljonen. Solving, rating and generating Sudoku puzzles with GA. IEEE Congress on Evolutionary Computation, 2007, 1382-1389.
- [2] Miguel Nicolau, Conor Ryan. Solving Sudoku with the GAuGE system. Springer Verlag, 2006, 213-224.
- [3] Rhydian Lewis. On the combination of constraint programming and stochastic search: the Sudoku case. Springer-Verlag, 2007, 96-107.
- [4] Alberto Moraglio, Julian Togelius. Geometric particle swarm optimization for the Sudoku puzzle. GECCO'07, July 7-11, 2007, 118-125.
- [5] Zong Woo Geem. Harmony Search Algorithm for Solving Sudoku. Springer Verlag, 2007, 371-378.
- [6] Zhanghui Chen, Xiaohui Huang, Wenyi Ren, Lie Kang. The diploid code genetic algorithm used to solve UTP. Unpublished.

Corresponding Author: Zhanghui Chen,
Huazhong University of Science and Technology
Center for Computational Materials Science and
Measurement Simulation, ZS3#507, Huazhong
Univ. of Science and Technology, Wuhan, China
11k2j3p4o5i6@163.com

Received & Accepted: MATHMOD 2009 -

Revised: September 10, 2009 -

Accepted: July 10, 2010 -



MATLAB-based Robot Control Design Environment for Research and Education

L. Žlajpah, B. Nemec, D. Omrčen, "Jožef Stefan" Institute, Ljubljana, Slovenia

SNE Simulation Notes Europe SNE 20(3-4), 2010, 55-66, doi: 10.11128/sne.20.en.09995

Research in the field of robotics is tightly connected to simulation tools for many reasons. On one side, simulation supports the development of new advanced control algorithms and on the other side it is always not feasible to build a whole robot system to test some algorithms or it is not safe to perform tests on a real system (at least in the first design stages). In the paper we present an integrated environment for the design and testing of advanced robot control schemes, including visual tracking, force feedback on a single robot or in multi-robot applications. The kernel of our simulation environment is MATLAB/Simulink. The main capabilities are: the simulation of the kinematics and dynamics of manipulators, the integration of different sensor systems like vision and force sensors, scenarios for complex robot tasks, the visualization of robots and their environment and the integration of real robots in the simulation loop. The advantage of our system is the simplicity, which allows easy integration of different robots, sensors and other devices. Some of these can be easier simulated by using other tools. Hence, other simulation tools can be used for the simulation of different parts of the system and then these subsystems are integrated in our simulation environment. The other important feature is easy final testing of developed control algorithms. Namely, for final testing of the control algorithms the models in the simulation scheme are just replaced by interface blocks for real system and the user does not need to consider implementation details. Finally, to show the efficiency and usability of our control design environment we outline some typical experimental examples using our robots. We explain some typical control design procedures from the "pure" simulation to the testing of algorithms on real robots.

SNE 20/2, August 2010

Introduction

The ways and methods in robotics research and development have always been influenced by the tools used. This is especially true when one considers the profound impact of recent technologies on robotics, especially the development of computers which become indispensable when designing the complex systems like robots. Not many years ago, computing cost was still a significant factor to consider when deriving algorithms and new modeling techniques [1, 2, 3]. Nowadays, distributed computing, network technology and the computing power developed by commercial equipment open new possibilities for doing systems design and implementation. However, in spite of all that the creativity of a human designer cannot be left out in the design process. The best solution seems to be to provide the designer with proper tools which significantly increase his efficiency. Among them, the simulation has been recognized as an important tool in designing the new products, investigating their performances and also in designing applications of these products. For complex systems as robots the simulation tools can certainly enhance the design, development, and even the operation of the robotic systems. Augmenting the simulation with visualization tools and interfaces, one can simulate the operation of the robotic systems in a very realistic way.

Currently, many different simulation tools for robotic systems are available. They differ from each other depending on which aspect of the robot research they support, how open they are or on which platforms they work. However, many tools are not always fulfilling all the requirements of the research and teaching activities in robotic laboratories like reconfigurability, openness and ease of use, etc.

Reconfigurability and openness are features already recognized by many as essential in the development of advanced robot control algorithms [4, 5, 6]. Not only is it important to have easy access to the system at all levels (e.g. from high-level supervisory control all the way down to fast servo loops at the lowest level), but it is a necessity to have open control architectures where software modules can be modified and exteroceptive sensors like force/torque sensors and vision systems can be easily integrated. Reconfigurability should also be reflected when more fundamental changes to the controller architecture are required, in the necessity of quickly being able to make modifications in the original design and verify the effect of these modifications on the system. In other words, the user should be able to quickly modify the structure of the control without having to alter the simulation system itself.

In the last decade the software has become more and more easy to use. This is still one of the main major issues when selecting a software tool. First of all, the tools are used by many users in a laboratory and not all of them have the same expertise. To boost the knowledge exchange, it is of benefit that they work with the same tools. Next, testing of different control algorithms on real robotic systems is in general not very user friendly: the algorithms usually have to be rewritten for the real-time execution and the different implementation details have to be considered [4, 7]. This forces the user to devote a large part of the design time to topics not connected with the main issues of the control design, especially when he is not interested in software implementation issues. The ease of use becomes even more important when students are working with robots. In most cases they work in a laboratory for a shorter period, they are focused on their projects and they could become frustrated if they have to learn a lot of things not directly connected to their tasks. Finally, in research laboratories different robot systems are used equipped with more or less open proprietary hardware and software architecture. Therefore, it is much desired that the control design environment is unified, i.e. the same tools can be used for all robot systems.

The simulation tools for robotic systems can be divided into two major groups: tools based on general simulation systems and special tools for robot systems [8]. Tools based on general simulation systems are usually represented as special modules, libraries or user interfaces which simplify the building of robot systems and environments within these general simulation systems (e.g. SolidWorks [9]). On the other hand, special simulation tools for robots cover one or more tasks in robotics like off-line programming and design of robot work cells (e.g. Robcad [10]) or kinematic and dynamic analysis [11, 12]. They can be specialized for special types of robots like mobile robots, underwater robots, parallel mechanisms, or they are assigned to predefined robot family. Depending on the particular application different structural attributes and functional parameters have to be modelled.

For the use in research and teaching laboratories, robot simulation tools focused on the motion of the robotic manipulator in different environments are important, especially those for the design of robot control systems [13, 11, 12, 14]. Recently, Microsoft Robotics Studio (MSRS) [13] has been launched with a general aim to unify robot programming for hobby-

ist, academic and commercial developers and to create robot applications for a variety of hardware platforms. The system enables both remotely connected and robot-based scenarios using .NET and XML protocols. Simulation engine enables real-time physics simulation and interaction between simulated entities. Each part of control loop can be substituted with the real or simulated hardware. Although the system is still under the development, it is not easy to add new entity, for example a new robot or a new sensor. One of the major drawbacks seems to be the low data throughput rate, which does not allow the realization of complex control laws at high sampling frequency. Therefore, it is not clear yet if MSRS is appropriate for research robotics, especially for complex systems. Real time requirements are better solved in another programming/simulation framework, MCA2 [15]. MCA is a modular, network transparent and realtime capable C/C++ framework for controlling robots and other hardware. The main platform is Linux/RTLinux, but support for Win32 and MCA OS/X also exists. However, it is still a complex system and therefore less appropriate for education and students projects.

MATLAB is definitely one of the most used platforms for the modelling and simulation of various kind of systems and it is not surprising that it has been used intensively for the simulation of robot systems. "The Robotics Toolbox" [11] provides many functions that are required in robotics and addresses areas such as kinematics, dynamics, and trajectory generation. The Toolbox is useful for simulation as well as for analyzing the results from experiments with real robots, and can be a powerful tool for education. However, it is not very good for the simulation in Simulink and for the hardware-in-the-loop simulation "SimMechanics Toolbox" [12] extends Simulink with the tools for modelling and simulating mechanical systems. With SimMechanics, one can model and simulate mechanical systems with a suite of tools to specify bodies and their mass properties, their possible motions, kinematic constraints, and coordinate systems and to initiate and measure body motions.

In the following, we present our approach to the integrated environment for the design and testing of robot control systems. Our framework is not intended as an alternative to the MSRS or MCA2. It is not as complex as MSRS and it does not possess physic simulation capabilities. On the other hand, real time capabilities cannot be compared to the MCA2.

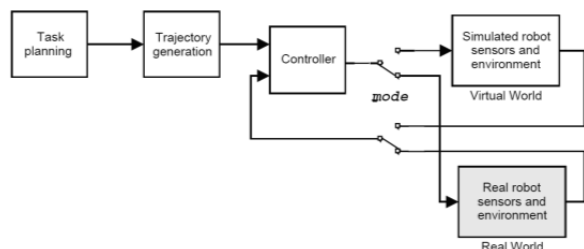


Figure 1. A block diagram of the integrated environment.

The advantage of our system is the simplicity, which allows easy integration of new entities and is also very appropriate for the education and research robotics. First we present our concept of the use of simulation tools in the control design for research and education. Next, the experimental setup in our laboratory is described. Finally, to show the efficiency and usability of our control design environment we outline some typical experimental examples using our service robot.

1 The concept

The importance of simulation tools in the development of robot control systems has been recognized by our team very early. We have been using different simulation tools for over 20 years and many of them have been developed in our laboratory. In the last decade we have been using for the control design MATLAB/Simulink based integrated environment based on Planar Manipulators Toolbox for dynamic simulation of redundant planar manipulators [7]. It enables the use of different sensors in the control loop and also the real-time implementation of the controller and hardware-in-the-loop simulation. Figure 1 shows the general simulation scheme in this environment.

A crucial feature inherited in this scheme is indicated by the mode switches. Namely, the user can easily switch between using model or a real system in the simulation loop. This is one of the main features which we need for development of robot control systems. The Planar Manipulators Toolbox has proved to be a very useful and effective tool for many purposes, but it has been primarily designed for kinematic and dynamic simulation of planar manipulators and to develop and test control algorithms on the lower control level, especially for redundant manipulators. In the last years, the scope of our research is oriented more in the development of control systems for humanoid and service robots.

These robots have in general a more complex mechanical structure with many degrees-of-freedom.

So, complex kinematic and dynamic models are necessary to simulate them.

Furthermore, the control methods and algorithms we are developing are now usually a part of the higher robot control levels and the low level close-loop control algorithms are assumed to be a solved issue. These high level control algorithms can become very complex and may even require parallel computation distributed over more computers.

Considering all new requirements, which are:

- to simulate the kinematics and dynamics of arbitrary chosen kinematic chain describing different manipulators,
- to enable integration of different sensor systems like vision and force sensors,
- to enable simulation of scenarios for complex robot tasks,
- to include the model the robots' environments,
- to visualize the robots and their environment and
- to enable integration of real robots in the simulation loop,

we had to reconsider the concept of the control design environment we will use in future. Based on our good experience with MATLAB/Simulink we have decided that this environment will be the kernel of our simulation tools. However, some of the above requirements can be easier fulfilled by using other tools. For example, the visualization of the robot and the environment can be easily done by dedicated graphics tools. Furthermore, advanced robot control strategies rely intensively on feedback sensor information. The most complex sensor system is the vision system, which can have several configurations and can be implemented on a single computer or on a computer cluster composed of many computers running different operating systems.

To integrate such a diversity of hardware components in unique framework we have decided to use the Ethernet communication and the UDP protocol. In this way, we have maximal possible "degree-of-openness" of the system. Figure 2 shows a typical scheme of our robot integrated environment.

In this scheme, each block can represent a real system or a model of that system.

Note that because we are using ethernet communication between the blocks, different software tools on different platforms can be used to simulate specific parts of the system.

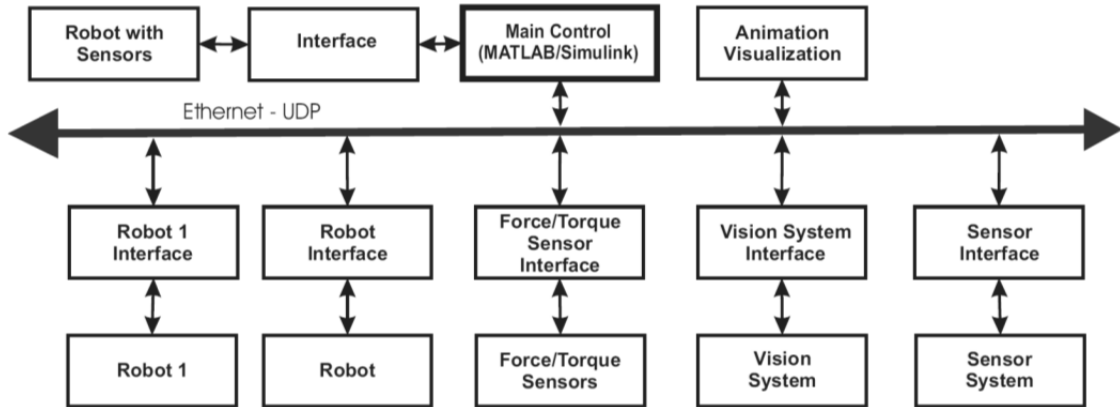


Figure 2. A functional block diagram of the robot integrated environment in Robotics Laboratory including the robot PA10, mobile platform Nomad XR 4000 and sensor systems

Consequently, the simulation environment can consist of several interacting applications, each representing a part of the system.

2 The experimental setup

The experimental setup in our Robotics Laboratory consists of several robots: the Mitsubishi PA10 robot which can be mounted on the Nomad XR 4000 mobile platform, humanoid robot Fujitsu HOAP3 and a 7 DOF humanoid head. They serve to research new approaches in the service and humanoid robotics.

In the following examples we use the Mitsubishi PA10 robot (see Figure 3). The PA10 robot is a general purpose seven degrees of freedom robot arm with brushless AC Motors and harmonic drive transmission in each joint.

The robot has an open architecture as well as in the hardware as in the software, and this provides the possibility to control and modify any aspect of the robot's behavior and to include new sensor information to the control system. The MHI controller is a four layer controller based on ARCNET which allows to control the robot in velocity mode at 100Hz. In same applications, it turned out that the MHI robot controller is not appropriate due to the limited sampling frequency, speed and acceleration limits and redundancy resolution algorithms used for the robot control. Therefore, we have developed an interface which communicates with the robot power system via ARCNET, which enables direct access to the velocity and torque motor inputs with sampling rates up to 700 Hz.

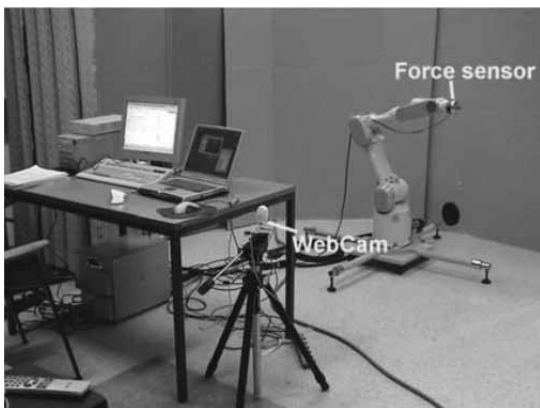


Figure 3. Experimental setup (Mitsubishi PA10, vision system and force sensor)

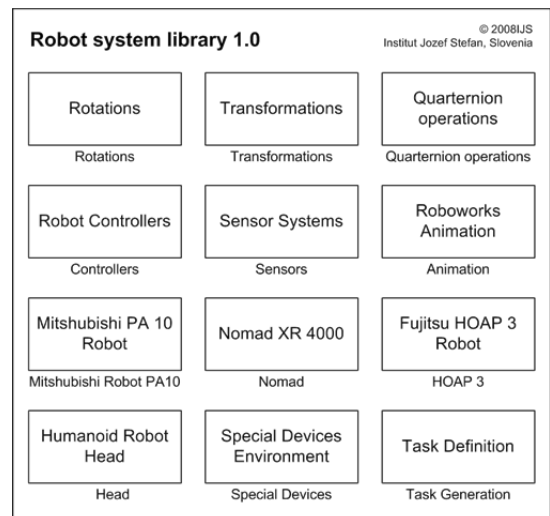


Figure 4. Simulink Robot systems library

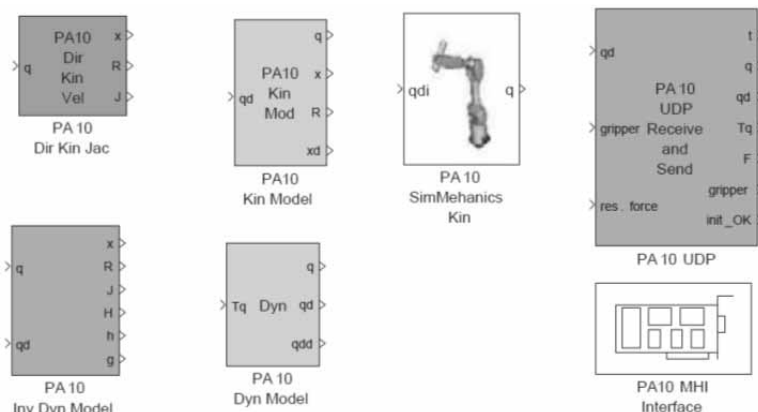


Figure 5. PA10 robot blocks

3 Simulink block library

In Simulink, a system is modelled by combining input-output blocks. To gain the transparency we try to represent a system by the block structure with several hierarchical levels, i.e. by combining different basic blocks subsystems are built which become a single block at the higher level.

In Figure 1 typical robot subsystems can be seen: the trajectory generation, the controller, the model of the manipulator and the environment and the animation of manipulator motion.

Figure 4 shows the Robot systems block library. The goal of the library is to provide blocks which are needed to simulate robotic systems and cannot be modelled with standard blocks.

First of all, this are the blocks for robot kinematic and dynamic models, the blocks for sensors systems, the typical transformations present in robot systems and the special interface blocks for robots, sensors and all other communications.

Additionally, the library includes some blocks with standard subsystems like task space controllers, trajectory generation modules, etc.

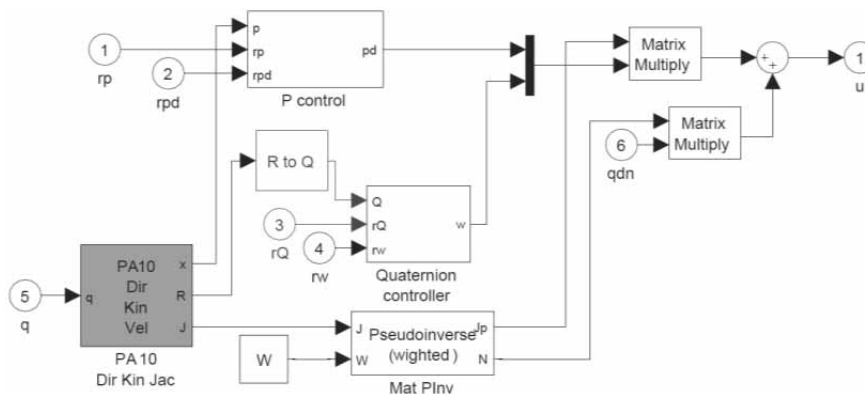


Figure 7. A block diagram representing the task space controller.

3.1 Robot models

Let the configuration of the manipulator be represented by the vector q of n joint positions, and the end-effector position (and orientation) by m -dimensional vector x of task positions. The joint and task coordinates are related by the following expressions

$$x = p(q), \quad \dot{x} = J(q) \dot{q},$$

$$\ddot{x} = J(q) \ddot{q} + \dot{J} \dot{q} \quad (1)$$

where J is the Jacobian matrix, and the overall dynamic behaviour of the manipulator is described by the following equation

$$\tau = H(q)\ddot{q} + h(\dot{q}, q) + g(q) - \tau_F \quad (2)$$

where τ is the vector of control torques, H is the symmetric positive-definite inertia matrix, h is the vector of Coriolis and centrifugal forces, g is the vector of gravity forces, and vector τ_F represents the torques due to the external forces acting on the manipulator.

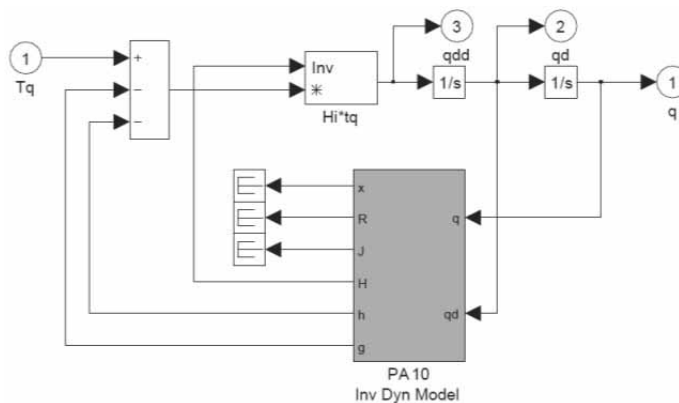


Figure 6. The dynamic model of the PA10 robot

The robot model blocks in the library (see Figure 5) represent the basic terms of the system as given in the above equations. Hence, the modelling of the robot is actually only the transformation of the model equations into block diagrams. In the library there are model blocks for all robots we are using. For example, the dynamic model described by the Eq. (2) for the PA10 robot, i.e. the block PA10 Dyn Mod in the library, is built using the basic block PA10 Inv Dyn Mod as shown on Figure 6, where the model matrices H , and g of the robot mechanism are calculated. The same principle is used for other robots and model types. Therefore, if one wants to use a dynamic model for another robot, he has only to substitute the block PA10 Inv Dyn Mod with an adequate block for the desired robot. In the same way, the other common subsystem which includes models is built. Figure 7 shows the task space controller for PA10 robot used later in examples. Here, a kinematic model of a robot is used to obtain direct kinematic transformation (end-effector position vector and rotational matrix) and the Jacobian matrix of the robot, which are needed in the control algorithm.

For hardware-in-the-loop simulation, it is necessary to use hardware interfaces with corresponding software drivers to include a real robots into the control loop. Usually, in case of robotic manipulators interfaces for actuators and sensors are needed. In the past, it was common to use D/A converters for controlling the actuators and joint positions were measured via incremental encoders. However, contemporary robot controllers and sensor systems enable the communication via different networks protocols like ethernet, Profibus, CAN, etc. In the Robot systems library, each robot has special interface blocks which allow simple integration of them into the simulation loop. For example, for the PA10 robot we have prepared drivers to control the robot using the MHI controller and using the Ethernet and UDP protocol.

Additionally, we have developed external applications for the simulation of our robots, which have the same interface as real robots. Using this applications, the control system realized in Matlab/Simulink is the same for the model or the real robot, i.e. the same interface blocks are used when a model or a real robot is included in the simulation loop. This enables easy and safe testing of control algorithms and the tests can be made even if the real robot is not available. When animation and visualization are also included, the simulation is even more realistic.

3.2 Integration of sensors

Advanced robotics is characterized by the variety of complex sensory system, e.g. vision sensors, force sensors, acoustic sensors, laser scanners, proximity sensor, etc. Therefore it is extremely important to apply as accurate as possible sensor models into the simulation environment. Models of sensors are completely transparent to the design environment, i.e. real sensor can be substituted with the simulated one and vice versa in the control loop.

The integration of sensors depends on their characteristics. Complex sensor systems like vision and acoustic sensors, or more advanced laser proximity sensors require relatively high computational power for signal processing. In many cases, it is difficult to accomplish all required data processing on the local computer. Often we have to apply a remote computer or even a remote computer cluster in order to obtain required computational power.

In such a case, the subsystems are connected through ethernet with UDP protocol. We have developed a special protocol classes for different sensors, actuators and other subsystems. However, the performance is also affected by the communication delays. Therefore, it is favourable to process signals of high frame-rate sensor, such as joint encoders, tachometers, force sensors, etc. on the local computer.

3.3 MATLAB robot language interpreter

When designing and testing complex robot tasks, it has turned out that standard Simulink blocks which can generate arbitrary trajectory cannot provide all the flexibility needed for complex robot tasks, especially for experimental work in service and humanoid robotics where the desired motion depends on the system/environment states. Commonly used solution for the definition of robot tasks are robot languages. Therefore, we have developed a MATLAB/Simulink block which can interpret the robot language. Included in the simulation it serves as the robot motion generator and supervisor. The developed interpreter module for Matlab Robot Language (MatRoL) is BASIC-like programming language extended with special commands for the robot control and supports all MATLAB interpreter commands.

In this way we have the advantage of a simple robot task definition and access to comprehensive MATLAB computation capabilities. The usage of the robot language is also favourable for the education.

Students can learn and accomplish their laboratory exercises much faster using robot language and the integrated environment allows safe testing of their algorithms on models and final tests on the real robots.

MatRoL is entirely written in Matlab. It has common instructions for the program flow (IF THEN ELSE, FOR NEXT, REPEAT UNTIL, GOTO <label>, GOSUB <procedure_name> RETURN) and special commands for the robot control (FRAME, MOVE, APPROACH, DEPART, SPEED, ACCELERATION, FORCE, NULLMOVE, TRAJECTORY). Additionally, all Matlab commands can be executed within a MatRoL program as an instruction. In this way we can use powerful Matlab matrix computation capability for controlling robot pose and for complex computation generally needed when vision and force sensors are applied. MatRoL supports various interpolation modes in Cartesian and task space, and supports also redundant robot systems, e.g. a special command NULLMOVE is used to define self-movement when kinematically redundant mechanisms are used.

Each robot in the simulation environment has its own MatRoL block, i.e. a special program. Program synchronization is done by assigning global variables, which can be signals, vectors, frames, or other. The MatRoL supports frame the orientation definition in roll-pitch-yaw angles, Euler angles and quaternions, while the interpolation is accomplished using the quaternions. The script in Listing 1 has been used for the vision based manipulation as explained later in Section 4.2.

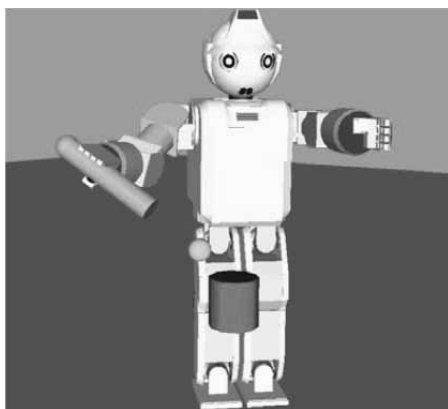


Figure 9. Animation of the HOAP 3 humanoid robot using RoboWorks.

```

1 % Case: Visual servo
2 TRACE 1
3 ! points(1).d=[0.0,0.0,1.1,0,-pi/2,0]';
4 ! points(10).d=[0,0,0,0,0,0]';
5 ! mat(10).d=eye(3);
6 SPEED 0.5
7 ACC 0.5
8 MOVE 1
9 ! ATK_Send('LetterA');
10 GRIP 1
11 GOSUB vservo
12 GOSUB grip
13 STOP
14 % -----
15 LABEL vservo
16 ! disp('VISUAL SERVO');
17 LABEL loop
18 ! [xc,var(1)]=Vis_Ser(points(10).d,mat(10).d);
19 IF (var(1) < 0.005)
20     GOTO exit
21 ENDIF
22 GOTO loop
23 LABEL exit
24 ! disp('DONE');
25 RETURN
26 % -----
27 LABEL grip
28 SPEED 0.1
29 DELAY 1
30 SPEED 0.1
31 TDEPART 0 0.03 0
32 DELAY 0.5
33 TDEPART 0 0 0.15
34 DELAY 0.5
35 GRIP 0
36 DELAY 0.5
37 TDEPART 0 0 -0.01
38 SPEED 1
39 move 1
40 TDEPART 0 0.04 0
41 GRIP 1
42 RETURN
43 END

```

Listing 1. MatRoL script for the vision-based manipulation

3.4 Visualization and animation

It is very important to visualize the simulation results. Especially in robotics it is necessary to “see” the motion of the robot and objects in the working environment. In our system we rely on external software for the visualization and animation of robots. In general, joint angles of robotic manipulators as well as the position and orientation of the other simulated objects in the scene are passed to the visualization tools using TCP/IP or UDP protocol.

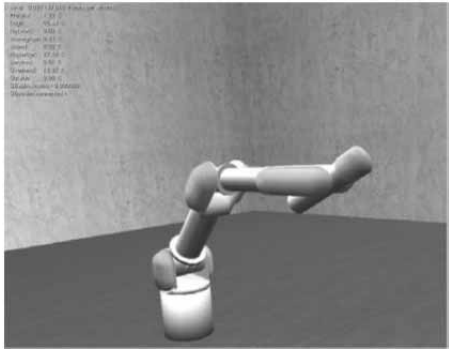


Figure 10. Animation of the PA10 robot in Blender.

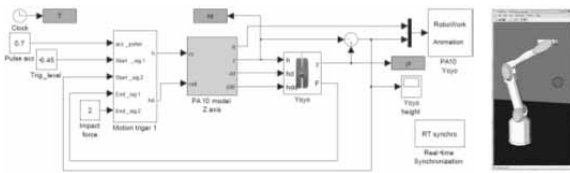


Figure 11. Yoyo simulation: top level block scheme in Simulink and animation of the PA10 robot and yo-yo.

Currently, we have integrated into our simulation environment two visualization software packages - RoboWorks [16] and Blender[17].

Roboworks incorporates simple, but efficient modeler. Because of its simplicity RoboWorks package is the favourable tool for the visualization of simpler systems, i.e. one or two robots in non-complex environment. Figure 9 shows the animation of our HOAP 3 humanoid robot and also in the following examples the RoboWorks environment has been used for the visualization.

For more complex scenes we use Blender, an open source multi-platform 3D computer animation program, which has a lot of features that are potentially interesting for engineering purposes, such as the simulation and programming of robots, machine tools, humans and animals, and the visualization and post-processing of all sorts of data that come out of such biological or artificial “devices”.

Blender supports also scripts (via Python interfaces to the core C/C++ code), hence it can be extended in many different ways. Among others, Blender has the capability of placing moving cameras at any link of the kinematic chain, it supports the real time photo realistic rendering for the virtual reality simulation and has also a physics engine for the simulation of the interactions between entities.

3.5 Real-time simulation

The real-time performance of the control algorithm is very important when dealing with low-level control. However, when developing higher level control algorithms real-time may be also important especially when high sample frequency improves the performance of the system. Therefore, when manipulator-in-the loop simulation is performed, the simulation system which controls the robot system has to provide real-time capabilities and enable high sample frequencies. There are many real-time operating systems as Real Time Linux, QNX, EYRX, SMX, etc. Disadvantages of these operational systems are time-consuming software development and incompatibility with other systems. The algorithms are usually written in C or some other low-level programming language, where more sophisticated control algorithms requires more time and increase the chance of error. Due to the above mentioned disadvantages of some real-time operation systems, we use the MATLAB/Simulink and the xPC Target operation system whenever possible [18]. xPC Target enables real-time simulation and hardware-in-the-loop simulation using corresponding interfaces. It is a good prototyping tool that enables to connect MATLAB/Simulink models to physical systems and to execute simulation in real-time on PC-compatible hardware. As xPC Target supports also UDP communication, this was also one of the reasons to select the UDP for the communication between different applications in the simulation environment. Nevertheless, using MATLAB/Simulink and xPC Target environment brings some disadvantages. Most of the hardware used for a robot control, which is available on the market, does not provide drivers for xPC Target. Therefore, we had to develop drivers for our robots and sensors.

4 Case studies

To show the efficiency, flexibility and usability of our control design environment we outline some typical experimental examples using the Mitsubishi PA robot and the mobile platform. We explain the complete design of the control system different simulation schemes used in this procedure from the "pure" Simulink simulation schemes, where the complete system is simulated in MATLAB/Simulink, to the hardware-in-the-loop schemes, where a real robot and sensor systems are part of the simulation loop and only the controller is realized in MATLAB/Simulink.

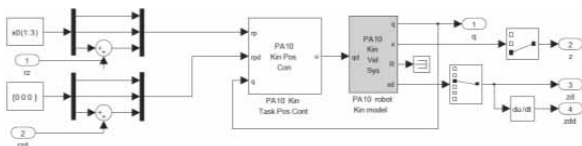
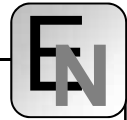


Figure 12. PA10 model with kinematic task space position controller

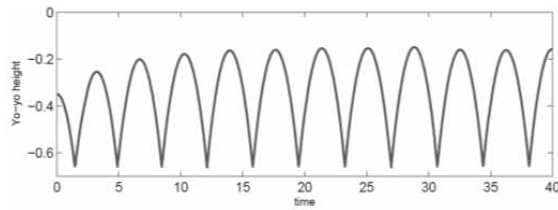


Figure 16. Swinging yo-yo motion - Simulation results

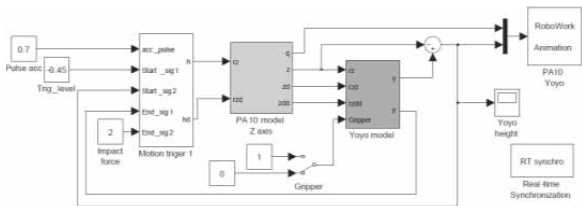


Figure 13. The case with kinematic PA10 robot model and external yo-yo simulator.

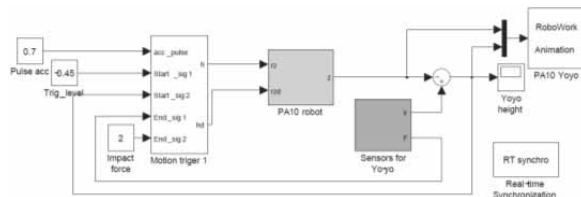


Figure 17. Hardware-in-the-loop simulation (real PA10 robot, force sensor and vision systems are in the simulation loop)



Figure 14. Interface for external yo-yo simulator (Yo-yo model block)

4.1 Playing yo-yo

In the first example we use the Mitsubishi PA10 robot arm to play yo-yo. The objective of playing the yo-yo is to keep the amplitude of the yo-yo at a desired level. The yo-yo is tied to the tip of the robot. To be able to play the yo-yo it is necessary to know the position of the yo-yo and the force in the string or the velocity of the yo-yo (depending on the control algorithm). A webcam has been used to measure the position of the yo-yo.

To measure the string force a JR3 force/torque sensor mounted on the end-effector of the robot was used. The experimental setup is shown in Fig. 3.

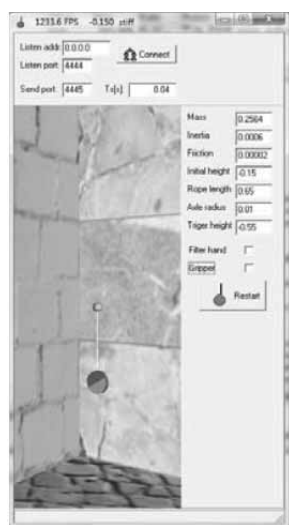


Figure 15. External yo-yo simulator.

The control should be implemented on PC's in MATLAB/Simulink environment and we wanted to use the PA10 motion control board which allows to control the end-effector positions of the robot. In the first step of the control design when different control strategies have to be tested, we simulated the whole system in Simulink. We used the PA10 kinematic model and we had to develop a Simulink model of the yo-yo. The top level simulation scheme is shown in Figure 11. The main three blocks are the controller, the robot model and a special model of the yo-yo [19]. As we want to move the robot end-effector only in the vertical direction the z-axis motion (x and y positions are fixed to the initial values), we have to use a kinematic task space controller. This subsystem can be easily composed by combining blocks in our Simulink library as it is shown in Figure 12.

After the best control strategy has been verified using this simulation scheme, the next step is to test the control when the sensor systems information is obtained via Ethernet connection. Therefore, we have developed a special yo-yo simulator, which receives the hand position and sends the position of the yo-yo and the string force via Ethernet connection using UDP protocol (see Figure 15). The simulation scheme is the same except that instead of yo-yo Simulink model the corresponding UDP interface blocks are used (see Figures 13 and 14).

As the external yo-yo simulator is a real time simulator, also in Simulink real-time simulation should be used.

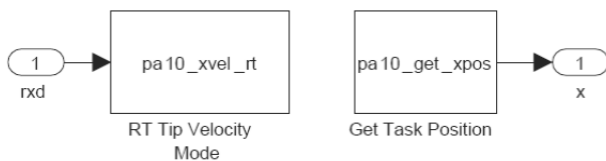


Figure 18. The interface block for PA10 task space position control

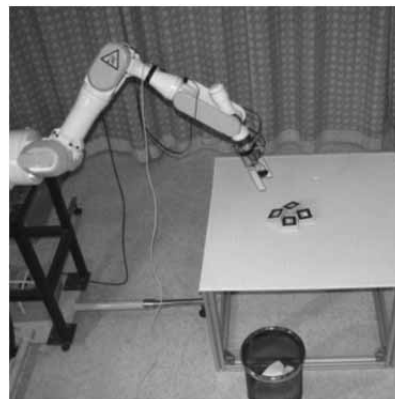


Figure 21. Experimental setup for vision based manipulation of objects

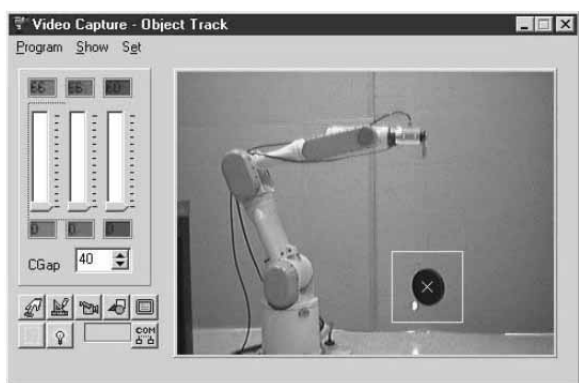


Figure 19. Capturing and identification of the yo-yo position with the webcam.

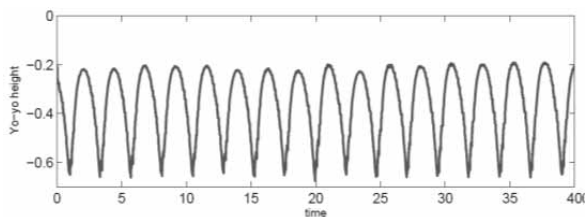


Figure 20. Swinging yo-yo motion - Experimental results.

As the sampling frequency in this case is rather low (100 Hz for robot control and 25 Hz for vision system) and the computation time of the Simulink model is small enough, we can use a special block for real-time synchronization. After tuning the controller parameters the simulation results for the yo-yo swinging height as shown on Figure 16 have been obtained.

Finally, when the designed control algorithms give satisfactory simulation results, we can test the control strategy on a real system. In manipulator-in-the-loop simulation, the model of the PA10 robot is replaced by the corresponding interface blocks. The position of the yo-yo is now obtained from the vision system and the force sensor via the same interface blocks as when the yo-yo simulator has been used. Figure 19 shows the user interface of the webcam based vision system.

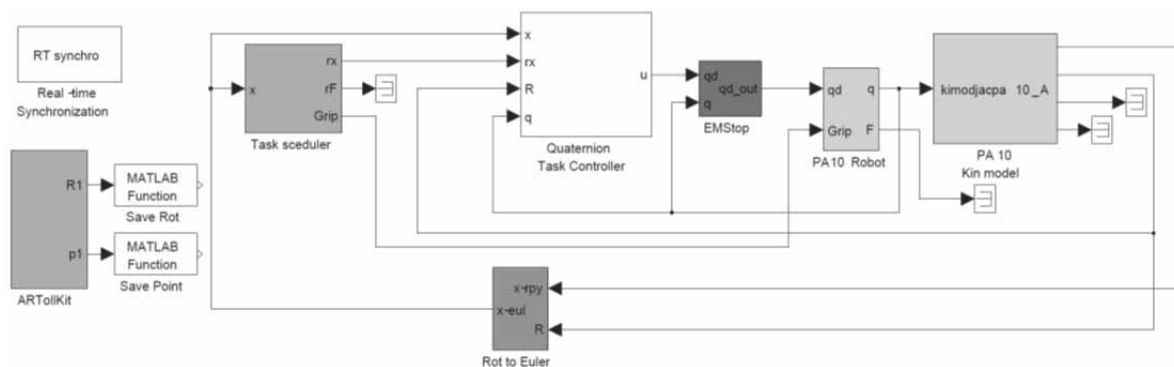


Figure 22. Vision based manipulation: Simulink block scheme

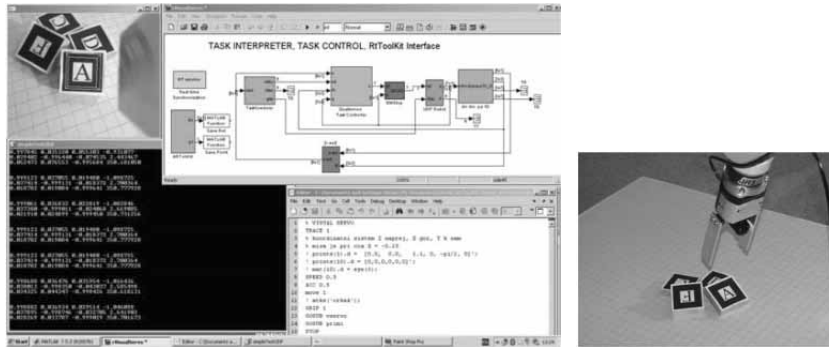
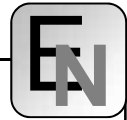


Figure 23. Vision based manipulation experiment: Robot is picking cube “A”

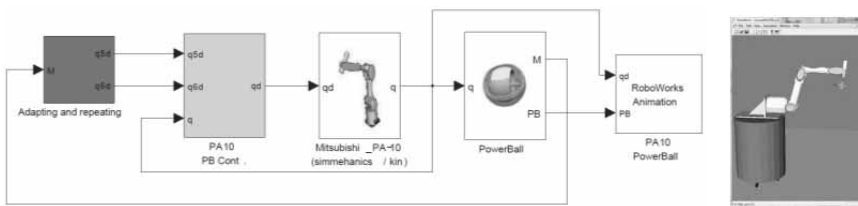


Figure 24. Power@Ball simulation: top level block scheme in Simulink and animation of the PA10 robot on Nomad platform and the Power@Ball

As explained before, special Simulink drivers for interfacing the PA10 robot control board, the JR3 force sensor and the webcam based vision system are already part of our Simulink library. Therefore, the user just replaces the model blocks. The corresponding scheme is shown in Figures 17 and 18.

From the top level scheme it can easily be seen that the controller part of the system has not been changed and is the same as in the previous simulation schemes. The final experimental results are shown on Figure 20. By comparing them with the simulation results on Figure 16 one can see that they are very similar. This confirms that simulation tools can be an important tool when designing control system.

4.2 Vision-based manipulation

In the second example we show the visual tracking experiment. The task for the robot has been to compose a text using cubes marked with letters.

The cubes have been randomly dispersed on the table. The robot has to identify a cube with the desired letter using vision, to grasp this cube and to place it on the table in order to compose the desired text. Note that cubes were arbitrary rotated in all three axes. Therefore, the visual tracking algorithm has to track not only the position of a cube but also the object orientation. Figure 23 shows the experimental setup.

To detect the object position and orientation we have used a USB webcam and the “Ar-ToolKit” - an open source software library for building Augmented Reality (AR) applications [20]. These are applications that involve the overlay of virtual imagery on the real world.

Although, augmented reality is generally not needed in robotics, ArToolKot was chosen because of its object recognition capabilities. Ar-ToolKit is capable of calculating 3D object position and orientation using single camera. The pose estimation is based on exact knowledge of the observed object geometry and its projection in the camera.

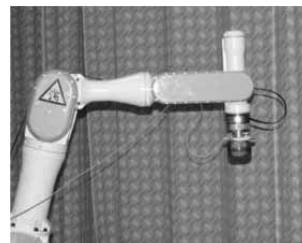


Figure 25. Experimental setup for spinning up the Power@Ball with PA10 robot

4.3 Playing “Power@Ball”

In the third example the robot should perform the spinning of a Power@Ball – a hand held gyroscopic toy or exerciser. To accelerate the rotor of the device with a

robot, we first measured the way a human does it. Using the results from the motion capture, we transferred the movement of the wrist to the end-effector movement of the robot. For a successful spin-up a synchronization of the exerted torque with the control velocity of the circular motion is necessary. Figure 25 shows the experimental setup. Different control approaches using feedback information from the velocity counter and force/torque sensor were applied.

First, they have been tested using SimMechanics model of the PA10 and the model of the Power@Ball. Figure 24 shows the block scheme and the animation of the system in RoboWorks. Finally, the experiment with a real robot in the loop has been done

The model of the robot and the Power@Ball has been replaced with the interface blocks as explained before. Figure 25 shows the experimental setup.

5 Conclusions

The concept of the presented control design environment is a result of our experience in the use of robots in research and education. It has proved to be a very useful and effective tool for fast and safe development and testing of advanced control schemes and task planning algorithms, including force control and visual feedback. The main part is implemented in MATLAB/Simulink and we have developed models for the robots and sensors used in our laboratory. To integrate the variety of components in a unique framework we have decided to allow the use of different tools for their simulation. So, the simulation environment can be composed of more than one application and the Ethernet is used for the communication between them. In this way, our environment is very open and can be very easily extended and adapted to different requirements and applied to any types of robotic manipulators. We have augmented the simulation with the animation and we show the importance of the possibilities offered by the simulation in the “virtual” world. One of the most important features of our simulation environment is that the testing on real robots is made very easy — the model real systems is simply replaced in the simulation loop by proper interface blocks. For that purpose, we have developed interfaces for the robots and sensors. Additionally, we have developed external applications which simulate certain robot subsystem and use the same interface as a real system. In this way, the user can test algorithms using the final control system but on a system on models which is very easy. Last but not least, it is an efficient tool for educational purposes. Thus, it should be of interest to the researchers involved in the development of advanced robot systems, and for teaching laboratories.

References

- [1] Yehong Zh., R.P. Paul. *Robot Manipulator Control and Computational Cost*. Technical Report MS-CIS-88-10, University of Pennsylvania Department of Computer and Information Science, 1988. <http://repository.upenn.edu/cis-reports/621>.
- [2] R.G. Fenton, F. Xi. *Computational analysis of robot kinematics, dynamics, and control using the algebra of rotations*. IEEE Trans. on Systems, Man, Cybernetics, (6):936 – 942, June 1994.
- [3] J.-C. Latombe. *Controllability, recognizability, and complexity issues in robot motion planning*. In Proc. 36th Ann. Symp. on Foundations of Computer Science, pp. 484 – 500, Los Alamitos, CA, USA, 1995.
- [4] J.M. Lambert, B. Moore, M. Ahmadi. *Essential Real-Time and Modeling tools for Robot Rapid Prototyping*. In Proc. 6th Int. Symp. on Artificial Intelligence and Robotics & Automation in Space i-SAIRAS 2001, Quebec, Canada, 2001.
- [5] G. Alotto, B. Bona, T. Calvelli. *Prototyping Advanced Real-Time Robotic Controllers on Linux RTAI Systems with Automatic Code Generation*. In Proc. Int. Conf. Mechatronics and Robotics 2004, Aachen, Germany, 2004.
- [6] V. Lippiello, L. Villani, B. Siciliano. *An open architecture for sensory feedback control of a dual-arm industrial robotic cell*. An International Journal Industrial Robot, 34(1):46–53, 2007.
- [7] L. Žlajpah. *Integrated environment for modelling, simulation and control design for robotic manipulators*. Journal of Intelligent and Robotic Systems, 32(2):219 – 234, 2001.
- [8] L. Žlajpah. *Simulation in robotics*. Math. comput. simul., doi:10.1016/j.matcom.2008.02.017, 2008.
- [9] *RobotWorks – A Robotics Interface and Trajectory generator for SolidWorks*, www.robotworkseu.com.
- [10] Tecnomatix. *ROBCAD/Workcell*, User’s manual, 1988.
- [11] P. I. Corke. *A Robotics Toolbox for MATLAB*. IEEE Robotics & Automation Magazine, 3(1):24–32, 1996.
- [12] The Mathworks. *SimMechanics, User’s Guide*, 2005.
- [13] Microsoft Robotics Studio. MSDN. <http://msdn2.microsoft.com/en-us/robotics/default.aspx>.
- [14] Cyberbotics Ltd. *Webots User Guide*, 2005.
- [15] Modular Controller Architecture 2 - MCA2: <http://mca2.org/>
- [16] Roboworks: http://www.newtonium.com/public_html/Products/RoboWorks/RoboWorks.htm.
- [17] Blender: <http://www.blender.org/>.
- [18] D. Omrcen. *Developing Matlab/Simulink and XPC target real-time control environment for humanoid jumping robot*. In 16th Int. Workshop on Robotics in Alpe-Adria-Danube Region - RAAD 2007, pp. 18–23, Ljubljana, Slovenia, 2007.
- [19] L. Žlajpah. *Robotic yo-yo: modelling and control strategies*. Robotica, 24(2):211 – 220, 2006.
- [20] ARToolKit: <http://www.hitl.washington.edu/artoolkit>.

Corresponding author: L. Žlajpah,
"Jožef Stefan" Institute
Jamova cesta 39, 1000 Ljubljana, Slovenia
leon.zlajpah@ijs.si

Received & Accepted: MATHMOD 2009: -

Revised: January 2010 -

Accepted: may 22, 2010

SOFTWARE NOTE

**RapOpt – An Automation Tool for
Production-orientated Run-to-run Model Evaluation**

Norman Violet, Niko Rossner, Thomas Heine, Rudibert King

Berlin Institute of Technology, Chair of Measurement and Control, Germany

SNE Simulation Notes Europe SNE 20(3-4), 2010, 67-74, doi: 10.11128/sne.20.sw.09997

Mathematical models have been proven to be a key factor in optimizing production processes in recent years. However, in the case of biochemical processes the design is usually done using heuristics, since these systems show complex internal regulation mechanisms and strongly nonlinear behavior. This makes it difficult to find an appropriate model. In those cases, where a structured biochemical model has been successfully identified, the yield of the process can be increased significantly. Obtaining a suitable production model is usually a difficult and time consuming task, especially for biochemical systems. In this contribution the concept for an automation tool is presented which starts with the few noisy measurements of initial experiments to perform a model evolution from run to run. Thereby, the first unstructured model candidates are used for an optimal production-orientated process design whose realization will provide additional information about the dynamic behavior within the production area, thus, leading to new and improved model candidates. Due to the difficult measurement situation in biochemical processes many different model candidates may show a similar fit to the data why it is unwise to focus on one model candidate for process design, only. Furthermore, the use of more than one model candidate for the design procedure represents a kind of robustness for the planning. This cyclic procedure enables an optimal production design corresponding to the available measurement information at any time.

SNE 20/3-4, December 2010

Introduction

Mathematical models of biological productions play an important role for process planning and optimization [1]. Here, the main task of a model is the prediction of optimal substrate feeds in order to maximize the economical yield. Usually a human modeler will plan a number of experiments using his or her experience and heuristics. To obtain a mathematical description of the system the trends of the measurements are analyzed manually and the most important state variables and reaction schemes are postulated. Then a mathematical model is formulated using balance equations and conservation laws. The velocity of each reaction step has to be described using empirical kinetic equations. After the model implementation the values of the model parameters have to be determined in a time consuming optimization-based numerical identification. In an iterative way, the human modeler changes the reaction schemes and the kinetic terms until the model shows an appropriate fit to the experimental data. Because of this tedious procedure not all promising reaction kinetics will be tested.

Thus, there might exist many other different model structures which would fit the few existing measurements similarly well or even better. After the identified model was used to plan new feeding profiles, often a significant extrapolation outside the domain of identification experiments takes place. Here, this model is often no longer valid. Therefore, the modeling procedure has to be repeated. The result of this error-prone and time consuming scheme is highly depending on the expertise of the human modeler.

In the last decade many software tools have been presented to simplify the modeling procedure [2]. Commercial tools like AspenPlusTM, ChemCADTM or gPromsTM are usually highly specialized on a certain field of application and rely on established methods, while academic institutions rather use prototypic realizations of new approaches. They often focus on the automation of major modeling steps [3, 4, 5, 6, 8, 9, 10, 11]. Besides balance equations with reaction kinetics, many recent tools like Simpathica [12, 13], TAM-B [9, 11], ProMoT [14, 7] and BioChem [20, 19] also consider temporal logic in order to integrate

information from heuristic observations in the mathematical model. While TAM-B uses this information to refine the reaction scheme of an ODE system, BioChem uses the temporal logic to describe additional constraints for a canonical S-System [21]. The tool ProMoT is based on network theory and provides a graphical interface with *drag-and-drop* functionality, which allows to quickly build a model out of standard elements stored in a library. It offers different input and output standards, providing a wide variety of interfaces for further processing. The software tool RapOpt [22] presented in this paper focuses on a data-driven continuous model evolution starting with the measurements from the first experiments. In order to test the fitting of different models, RapOpt will interchange individual kinetics within a given basic structure, automatically code and compile the model files and thus create a multi-model system environment. The refinement of each model in every iteration cycle is orientated towards product maximization. The paper is organized as follows. In section 1 the progress of the run-to-run model evolution will be introduced in general, whereas only the central functionalities are described in this contribution. The final section is devoted to an example of a multi model process design and its experimental realization is presented.

1 Run-to-Run model evolution with RapOpt

1.1 Definition of a model family

In order to initialize RapOpt, the user has to define the system's states that should be considered. For the first crude, unstructured model it is assumed that all substrates may influence the reaction rates of the specified states. Additionally, measured data is required which can either be obtained from initial experiments in Erlenmeyer flasks and/or from the database of a process control system of a fermenter.

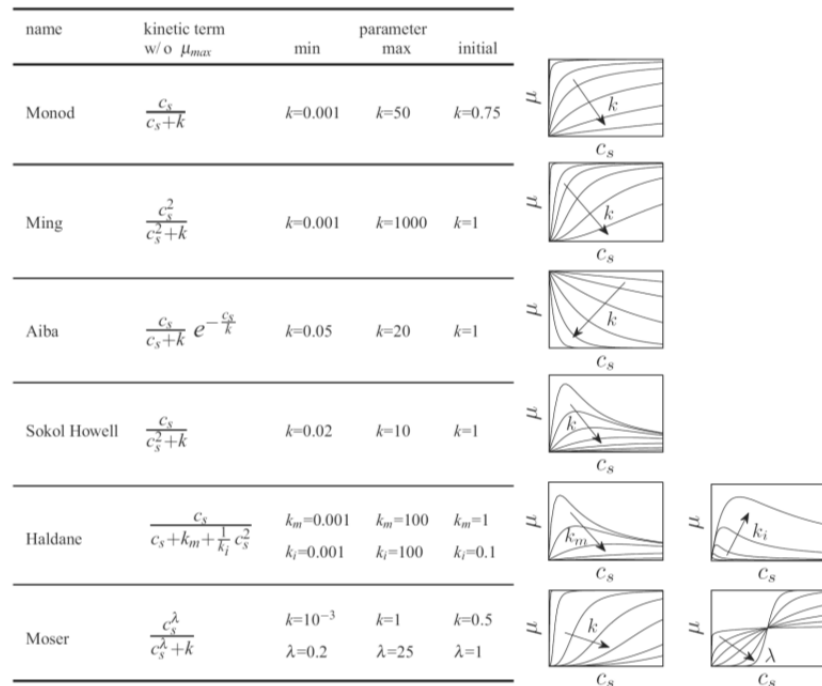


Figure 1. Examples of commonly used rate equations for biological systems that are stored in the kinetic library without the corresponding maximal specific growth rate μ_{max} . The library also contains information about the parameter bounds and the initial parameter values. To the right, a set of curves is shown to illustrate the shape of the kinetic when varying its parameter(s).

Furthermore, the user has to define those dependencies in the reaction rates which are supposed to be interchanged by RapOpt as well as the permitted kinetics for this process (see Figure 1). Choosing all dependencies as changeable will cause a huge number of model candidates as it will be explained later in this section. At this point, the model family is completely defined and the investigation of its individual members concerning the available data will follow. To clarify the definition of a model family, a short example is introduced.

The growth of the biomass m_x is an autocatalytic process whose specific growth rate μ_x is influenced by three substrate concentrations c_{S1} , c_{S2} and c_{S3} . The balance equation of a (fed-)batch fermentation without cell death then reads

$$\dot{m}_x = \mu_{max} \underbrace{g_1(c_{S1})g_2(c_{S2})g_3(c_{S3})}_{\mu_x} m_x \quad (1)$$

In this basic structure of the model family, the reaction rate μ_x is a product of the three kinetics g_1, g_2 and g_3 depending on the different substrates c_{S1} , c_{S2} and c_{S3} . The a priori unknown kinetics for the specific growth rate - and analogously of every other unknown reaction rate - can be described using empiri-

cal kinetic expressions as shown in Figure 1. Besides the name of the kinetic and the mathematical expression, the library also contains meaningful initial values for the parameters in order to guarantee a typical behavior during simulation. Moreover, minimal and maximal values are given. These avoid the degeneration of kinetics. To create all members of a model family, all interchangeable kinetic terms will be permuted automatically by RapOpt using the list of the permitted kinetics, beginning with the least parameterized terms. In order not to create senseless models, a simple logic is implemented that for example avoids the use of strictly inhibiting kinetics in a growth rate when substrate is the dependency. Methods from TAM-B [9, 18, 11] shall be used to eliminate inappropriate models in future. Nevertheless, the kinetic library contains more than 50 different empirical expressions to describe building rates, which can lead to a huge number of models. In the example, see Eq. (1), $50^3 = 125\,000$ candidates for μ_x can be generated, which have to be identified.

In the case that only very few measurements are available initially, the tool just activates the three most often used kinetic terms which contain at most two parameters to create different initial model candidates. In most cases the few initial measurements can be fitted similarly well with different kinetics, even if some reaction rates contain only one parameter. All created models constructing the model family and their corresponding parameter files will be coded automatically by RapOpt in a MATLAB m-file for easy interpretable documentation as well as coded and compiled in C for accelerated simulation. A short `select_model` command allows the user to change between different models, whereby a multi-model environment can be easily embedded in existing MATLAB programs which will be detailed below.

1.2 Parameter identification for all members of a model family

As pointed out in the previous section, the permutation of kinetics can lead to a large number of model candidates, for each of which the parameters have to be identified in a nonlinear optimization procedure. The numerical burden for a nonlinear parameter identification depends on the degree of the nonlinear couplings, the optimization algorithm, and on the quality of the initial values and measured data. In RapOpt, the time-consuming calculations for all model candidates are accelerated using three short-cut methods beside a multiple shooting approach [17].

Sequential Parameter Identification The dependencies among the design parameters can be cut off with a sequential identification procedure. Normally in an identification, the ODE system has to be simulated to calculate the value of the maximum likelihood (ML)-objective. In a nonlinear system, each equation of the ODE system is usually coupled to many other equations. By using interpolated measurements instead of the simulations for all measured states, these equations can be decoupled, such that a parameter in one equation does not affect the results of other equations. Thus, the identification problem is partitioned in a series of identifications. The first problem of this sequence only contains a few design parameters. By replacing the data interpolations with the model simulations step by step, the forthcoming identifications grow piecewise until the original identification problem is solved.

Determination of Initial Values The sequential identification is used for the first model candidate only and provides a well fitted initial model. For all further model candidates the similarity between the different models is used to generate initial parameter values for the next identification. This presumes a designated order, in which consecutive models only differ in one kinetic term. The parameters of the current identification are initialized with the optimal parameters of the former model which ensures a converging identification procedure. The new parameters of the changed expression (e.g. Figure 1) are determined within the given bounds such that the kinetic term is as close as possible to that of the former candidate. For this process no time-consuming simulation of the ODE system is necessary.

As an example, Figure 2 shows how some kinetics from Figure 1 are equalized to a Monod equation with given parameter $k = 1$ within the given range $0 \leq c_s \leq 1$. In RapOpt this range will be determined according to the experimental data used for parameter identification. The kinetic *Moser* is not shown in this figure, because by setting $\lambda = 1$ it can be exactly transformed in a Monod kinetic. As a result of this kinetic equalization, the initial simulation of the new model is very close to the final simulation of the previous model candidate.

Sequence of Identification Depending on the number of model candidates, there might not be enough time to identify all of them. Therefore, most promising models have to be identified first. In RapOpt this is achieved using a hierarchical tree structure.

The top level consists of the model candidate with the most simple kinetic terms (usually all *Monod*) as a parent for further variations. The second layer is derived from the first by replacing one dependency with each of the activated reaction equations and thus creating several children. Therefore, in the appearing tree structure, adjoining models along the branch differ in one term only. Moreover, all children of a model only differ in one term as well. The following heuristic is used to choose the most promising model candidates for the next identifications. After the identification of the first model, all of its direct children will be identified. The child with the best fit to the measured data will be the new parent, whose children will then be tested. This identification procedure continues until all leaves of one branch of this systematic tree have been reached. At this point, every dependency was interchanged with all activated kinetics from the library even though not all permutation have been identified yet. Then, the procedure continues with the model in the data tree which shows the second best fit to the measurements, and so on. This strategy is based on the assumption, that a better model always arises from a good model by further changes in individual dependencies and therefore many promising models are identified at an early stage. However, the best model can be located somewhere in the tree. Still a process design with appropriate model candidates can be started already as explained next while the identification procedure continues to find even better models.

1.3 Multi-model trajectory planning

At an early stage of process development only few measurement information is available. Many of the model candidates created in Section 1.1 and 1.2 will fit the measurements similarly well with differences in the objective values in the order of the measurement noise. Nevertheless, they all have different structures, with different parameter sensitivities according to the measurements given. A design procedure that is based on the best model only, runs the risk of showing a bad extrapolation of the model behavior around the planned trajectory either caused by a wrong mathematical structure or by parameters that had been insensitive during identification and have now a significant effect in process design. Moreover, judging a model by its objective value is delusive, due to the fact that a gradient-based optimization could have stopped in a local minimum.

Name	Parameter
Monod	$k = 1$
Ming	$k = 0.655$
Sokol Howell	$k = 1.185$
Haldane	$k_m = 0.988, k_i = 38.126$
Moser	$k = 1, \lambda = 1$

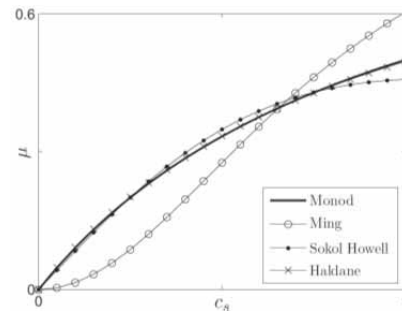


Figure 2. Parameters of different kinetics adjusted such that they fit the Monod kinetic ($k = 1$) in a least squares sense.

Not rarely, a better objective can be found if the optimization is restarted at the last minimum, because of an untrained Hessian matrix. By considering more than one model in the planning procedure, these problems can be addressed. The extrapolation to an extreme dynamical behavior, that a single model could predict, is now partly covered by the others.

Moreover, the difficulty of finding the best model is circumvented by optimizing the feeding profiles according to the yield predicted by all models. The simplest objective function for a multi-model trajectory planning would be to maximize the mean or median of the product amount. More robust trajectories can be obtained if the objective is formulated using the minimal product yield obtained over all models.

1.4 Preparing the next evolution

As more measurement information becomes available from run to run, two different evolutions can take place. At first, an automated data analysis will be done in order to investigate whether or not the ODE structure has to be refined by intra-cellular storages of nutrients and products to slowly obtain a more and more structured compartment model. If new states are postulated, the procedure will restart using the simplest kinetics. Otherwise, more complex kinetic terms will be tried out for the best models of the last cycle. The percentage of models that should be carried over to the next cycle can be investigated as follows.

Let m be the number of permitted kinetics of the last cycle and n the number of new kinetics for the forthcoming. If d is the number of dependencies wherein the kinetics are inserted, then md models were formerly considered and $(m+n)d$ would have to be identified in the next cycle. Reusing a certain fraction X of old structures can reduce the number of models if

$$X \cdot m^d \cdot (n+1)^d < (m+n)^d \quad (2)$$

$$X < (m+n)^d / (m \cdot (n+1))^d \quad (3)$$

holds. It has to be observed that reusing models will lead to several identical models, when new kinetics are inserted for terms that formerly distinguished the models from each other. Therefore, a routine has to be implemented that eliminates all duplicated models. Nevertheless, this method promotes the actual evolution, since further modifications are based on the properties of the fittest models only.

2 Experimental part

The development of the RapOpt software-tool for a run-to-run optimization was followed in parallel to the synthesis of an optimization, based on a multi-model approach. Therefore, the first experimental results obtained from a multi-model planning which is presented here, did not make use of all possibilities concerning automatic modelling as described above. Instead, some modeling steps had been done manually to describe the growth and production behavior of the bacteria *Paenibacillus polymyxa*, see below. Early experiments with this organism have pointed out that a simple unstructured model has to be refined by a storage term for phosphate, giving rise to the following low-structured model family.

$$\dot{m}_x = \mu_x m_x \quad (4)$$

$$\dot{m}_{am} = -Y_{am/x} \mu_x m_x + c_{am,feed} u_{am}(t) \quad (5)$$

$$\begin{aligned} \dot{m}_{ph} = & -(Y_{ph/x} \mu_x + Y_{ph/pp} \mu_{pp}) m_x + \\ & + c_{ph,feed} u_{ph}(t) \end{aligned} \quad (6)$$

$$\begin{aligned} \dot{m}_c = & -(Y_{c/x} \mu_x + Y_{c/ML} v_{pp} \mu_{ML} + Y_{main}) m_x \\ & + c_{c,feed} u_c(t) \end{aligned} \quad (7)$$

$$\dot{m}_{pp} = (\mu_{pp} - Y_{pp/ML} v_{pp} \mu_{ML}) m_x \quad (8)$$

$$\dot{m}_{ML} = \mu_{ML} m_x \quad (9)$$

$$\dot{V}_x = u_{am}(t) + u_{ph}(t) + u_c(t) \quad (10)$$

$$\mu_x = \mu_{max,x} g_1(c_{am}) g_2(c_{ph}) g_3(c_c) \quad (11)$$

$$\mu_{ML} = \mu_{max,ML} h_1(c_{am}) h_2(c_c) h_3(c_{pp}) \quad (12)$$

$$\mu_{pp} = \mu_{max,pp} \text{ming}(c_{ph}) \text{aiba}(c_{pp}) \quad (13)$$

$$v_{pp} = \text{aiba}(c_{ph}) \quad (14)$$

The balance equations for biomass m_x and the product macrolactin m_{ML} contain variable unknown kinetics g_i , h_j that depend on the concentrations of the substrates glucose (index 'c'), ammonium ('am'), phosphate ('ph') and polyphosphate ('pp'). For each of these variable kinetics one of the three allowed kinetics *Monod*, *Moser*, *Ming* for the growth and *Monod*, *Haldane*, *Ming* for the production from Figure 1 was inserted by RapOpt to define an individual member of the model family. Since there are 6 variable kinetic terms in the model family and three different permitted kinetic terms, $3^6 = 729$ different models had been set up automatically. After the automatic computational implementation of all models had been completed as described in the previous section, a parameter identification for every single model had been carried out. The result of these identifications are shown in Figure 3 as a histogram, where the number of models with a certain amount of the objective Φ_{MLE} is given.

The histogram clearly illustrates that many models can describe the measurements with similar quality. As argued in the previous section, using the model with the best objective value ignores the fact that the measured data might not be informative enough to clearly discriminate one model from the others as well as the problem of local minima in the objective of the parameter identification. Nevertheless, it is obvious that a lot of model structures are not able to fit the underlying measurement information. The question remains how many of the suitable model candidates should be used for the upcoming process design. In this case, the best 4 model structures according to the objective value were added to the 5 manually built models that already existed.

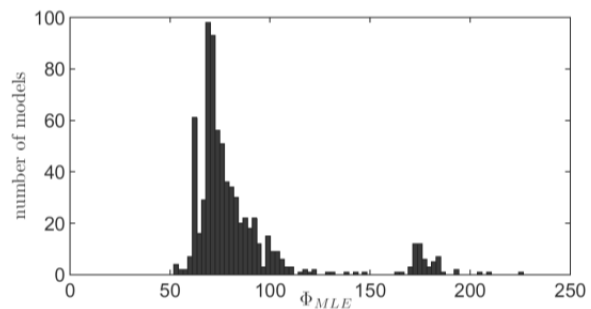


Figure 3. Histogram of the final ML-objective for the parameter identifications of all 729 models.

no.	growth				production				Φ_{ML}	
	$\mu_{max,x}$	$g_1(c_{am})$	$g_2(c_{ph})$		$g_3(c_c)$	$\mu_{max,m}$	$h_1(c_{am})$	$h_2(c_c)$		$h_3(c_{pp})$
1 <i>m.</i>		Ming	Moser		Ming		Monod	Ming	Sokol Howell	
	0.249	$k = 0.0158$	$k = 0.088$	$\lambda = 0.909$	$k = 0.0059$	0.005	$k = 0.0413$	$k = 0.037$	$k = 0.897$	
361		Monod	Moser		Monod		Monod	Monod	Haldane	
	0.347	$k = 0.786$	$k = 0.095$	$\lambda = 0.783$	$k = 0.012$	0.115	$k = 0.025$	$k = 0.031$	$k_m = 12.06$	$k_i = 0.018$
316		Monod	Moser		Monod		Ming	Monod	Haldane	
	0.335	$k = 0.724$	$k = 0.091$	$\lambda = 0.812$	$k = 0.007$	0.095	$k = 0.003$	$k = 0.037$	$k_m = 6.717$	$k_i = 0.018$
2 <i>m.</i>		Monod	Haldane		Monod		Ming	Monod	Haldane	
	0.403	$k = 0.470$	$k_m = 0.078$	$k_i = 1.150$	$k = 0.003$	0.110	$k = 0.002$	$k = 0.041$	$k_m = 9.433$	$k_i = 0.018$
3 <i>m.</i>		Monod	Haldane		Monod		Monod	Monod	Haldane	
	0.410	$k = 0.398$	$k_m = 0.068$	$k_i = 0.940$	$k = 0.002$	0.099	$k = 0.010$	$k = 0.043$	$k_m = 9.592$	$k_i = 0.020$
4 <i>m.</i>		Monod	Moser		Monod		Ming	Monod	Sokol Howell	
	0.326	$k = 0.594$	$k = 0.100$	$\lambda = 0.867$	$k = 0.004$	0.057	$k = 0.001$	$k = 0.040$	$k = 14.477$	
5 <i>m.</i>		Monod	Haldane		Ming		Ming	Monod	Haldane	
	0.398	$k = 0.379$	$k_m = 0.068$	$k_i = 0.970$	$k = 0.001$	0.099	$k = 0.001$	$k = 0.051$	$k_m = 10.75$	$k_i = 0.021$
568		Ming	Moser		Monod		Ming	Monod	Haldane	
	0.262	$k = 0.376$	$k = 0.087$	$\lambda = 0.544$	$k = 0.001$	0.090	$k = 0.019$	$k = 0.041$	$k_m = 10.27$	$k_i = 0.030$
424		Monod	Moser		Monod		Ming	Ming	Haldane	
	0.317	$k = 0.469$	$k = 0.082$	$\lambda = 0.826$	$k = 0.004$	0.105	$k = 0.005$	$k = 0.048$	$k_m = 11.06$	$k_i = 0.019$

Table 1. Inserted kinetic terms of nine model candidates that were used for multi-model trajectory planning. For the mathematical expressions of the kinetic terms see Figure 1. Model numbers denoted by *m.* had been built manually.

no.	v_{pp}	μ_{pp}			yield coefficients						
	aiba	μ_{max}	ming	aiba	$Y_{am/x}$	$Y_{ph/x}$	$Y_{ph/pp}$	$Y_{c/x}$	$Y_{c/main}$	$Y_{c/ML}$	$Y_{pp/ML}$
1 <i>m.</i>	0.2503	4.340	1.9254	2.9309	0.1848	0.0228	12.936	1.7753	0.0782	15.232	0.3716
361	0.0500	0.005	0.1090	22.095	0.1891	0.0468	10.237	1.9250	0.0626	19.048	0.1401
316	0.0883	0.025	0.1183	21.638	0.1897	0.0471	8.0091	1.8984	0.0642	19.104	0.1442
2 <i>m.</i>	0.0624	0.048	0.1113	22.034	0.1932	0.0475	10.753	1.6069	0.0642	21.728	0.1445
3 <i>m.</i>	0.0573	0.005	0.1099	22.450	0.1969	0.0484	11.778	1.8260	0.0702	18.953	0.1419
4 <i>m.</i>	0.0728	0.061	0.0834	19.965	0.1892	0.0467	13.746	1.6376	0.0639	21.134	0.1306
5 <i>m.</i>	0.0561	0.054	0.1104	22.682	0.1958	0.0487	11.705	1.8195	0.0700	19.046	0.1395
568	0.0616	0.061	0.0892	22.752	0.1868	0.0476	11.223	1.7779	0.0636	20.289	0.1318
424	0.0534	0.049	0.0940	22.308	0.1940	0.0488	12.344	1.6834	0.0701	20.000	0.1442

Table 2. Parameter values of those kinetics that had not been interchanged (Eq. (13) and (14)) as well as the identified yield coefficients of all nine models. Model numbers denoted by *m.* had been build manually.

Hence, in this point we left the route given in Section 1, as RapOpt missed some functionalities when the experiment was scheduled. In Table 1 the kinetics and their corresponding parameter values for the interchanged dependencies of these models as well as its objective value after parameter identification are listed. Table 2 shows the yield coefficients and the parameters of the kinetics that had not been interchanged.

During the identification of the parameters these 9 models showed a very similar outcome as shown in Figure 4 for one of the experiments used in the identification process.

It points out that different model structures show a similar fit to the measurements after parameter identification. When using these models for a multi-model trajectory planning, the individual trajectories differ a lot, as can be seen in Figure 5 as grey solid lines. The underlying process design is based on an objective that maximizes the mean of all nine predicted products, shown by the black solid line. When the process was run (measurements shown as open circles) none of the models at this early state could fully describe the behavior. Especially the predicted yield, here the antibiotic macrolactin, differs among the candidates.

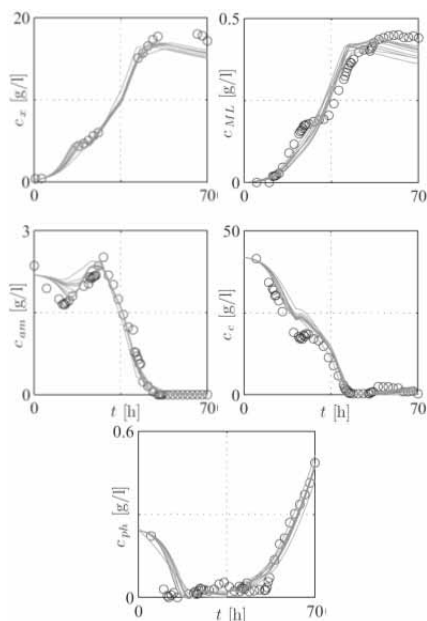


Figure 4. Simulation run of all nine models for one of the experiments used for parameter identification. Measurements of the concentration of biomass c_x , macrolactin c_{ML} , ammonium c_{am} , glucose c_c and phosphate c_{ph} are given by open circles, the different model simulations by grey solid lines.

However, the obtained yield of macrolactin was higher than the one obtained by our partners of the biological department in first tests and taking into account that macrolactin is a secondary metabolized product these first results are promising. Moreover, these data are now used in the aforementioned model evolution.

Furthermore it is noticeable that the simulated phosphate concentrations show differences between the individual model and a bad fit in the first part of the experiment. The range can be partly explained by the obviously different implementations of the assumed polyphosphate reactions which cannot be fitted since there are no measurements available. Beside of that it is reasonable to assume that phosphate is involved in a reaction not covered by these early model candidates. The model evolution will take care of that later on. The difference between planned trajectories of c_{am} and the measurements at the end of the experiment can be explained by the violation of the glucose concentration constraint; the models were not able to cope with this condition. It has to be noted that the feeding container for the glucose substrate run dry during the last 10 hours of the experiment.

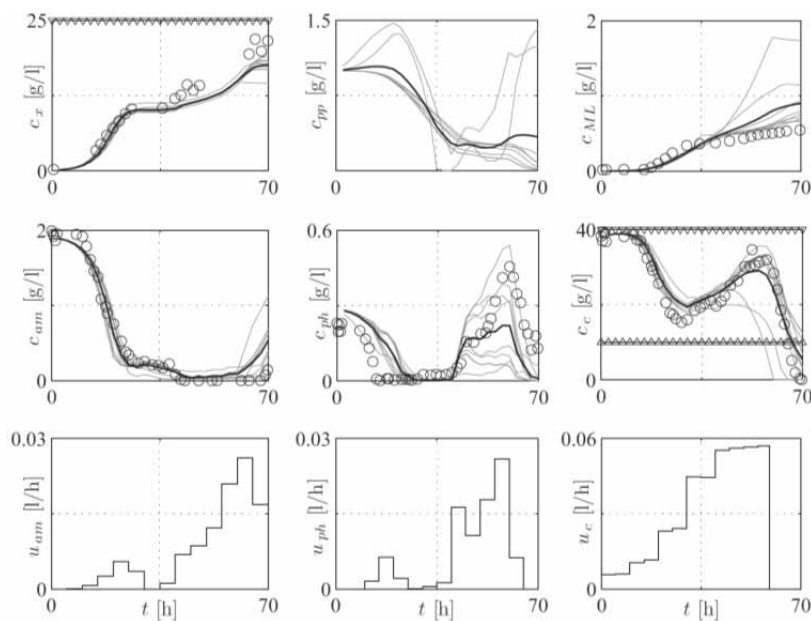


Figure 5. Optimized trajectories of the different models in a process design that considers all nine models. First and second row as in Figure 4. Third row: optimal feeding profiles used for ammonium, phosphate and glucose feed. Constraints are shown as lines with triangular.

This explains the feeding profile of u_c in Figure 5 and the violation of the lower boundary of c_c . The last part of the experiment was re-calculated in this Figure to reflect the realized feeding profile to allow a better comparison between planned trajectories and measurements.

3 Conclusion

In this contribution, it was shown how RapOpt enables an automatic modeling based on a multi-model approach. Moreover, by connecting this automated modeling procedure with a multi-model process design an evolutionary model development was established that focuses from the beginning on the product output of the process. Therefore, the modeling procedure can be performed in parallel to the actual production process leading quickly to a certain amount of product while the model will be continuously refined.

Acknowledgement

This work was partly supported by AIF, project 14775N. Additional financial support from the Deutsche Forschungsgemeinschaft and the Cluster of Excellence Unifying Concepts in Catalysis (EXC 314/1) are gratefully acknowledged.

References

- [1] Roubos J.A. *Bioprocess Modelling and Optimization: Fed-Batch Clavulanic Acid Production by Streptomyces*. PhD Thesis, Delft University of Technology, Netherlands, 2002.
- [2] von Watzdorf R., Allgöwer F., Helget A., Marquardt W., and Gilles E.D. Dynamische Simulation verfahrenstechnischer Prozesse und Anlagen - Ein Vergleich von Werkzeugen. In: *Simulationstechnik 9. ASIM-Symposium*, Wiesbaden, 1994, pp. 83–88.
- [3] Bonvin D. and Rippin D.W.T. *Target Factor Analysis for the Identification of Stoichiometric Models*. In: *Chemical Engineering Science*, Vol. 45, No. 12, pp. 3417–3426, 1990.
- [4] Bettenhausen K.D., Marenbach P., Freyer S., Rettenmaier H. and Nieken U.: *Self-Organizing Structured Modelling of a Biotechnological Fed-Batch Fermentation by Means of Genetic Programming*. In: 1st Int. Conf. Genetic Algorithms in Engineering Systems: Innovations and Applications, GALESIA, Vol. 414, pp. 481–486, 1995.
- [5] Bogaerts Ph. and Vande Wouwer A. *Systematic Generation of Identifiable Macroscopic Reaction Schemes*. In: 8th Int. Conf. Computer Applications in Biotechnology (CAB8), Quebec, pp. 13–18, 2001.
- [6] Bardow A. and Marquardt W. *Incremental and simultaneous identification of reaction kinetics: methods and comparison*. In: *Chemical Engineering Science*, Vol. 59, Issue 13, pp. 2673–2684, 2004.
- [7] Saez-Rodriguez J., Mirschel S., Hemenway R., Klamt S., Gilles E. D. and Ginkel M.: *Visual setup of logical models of signaling and regulatory networks with ProMoT*. In: *BMC Bioinformatics*, Vol. 7, No. 506, 2006.
- [8] Mandenius C.-F. *Recent developments in the monitoring, modeling and control of biological production-systems*. In: *Bioprocess and Biosystems Engineering*, Vol. 26, No. 6, pp. 347–351.
- [9] Leifheit J. and King R. *(Semi-)automatic modeling of biological reaction systems with TAM-B*. In: 9th Int. Conf. Computer Applications in Biotechnology (CAB9), Nancy, France, 2004.
- [10] Mangold M., Angeles-Palacios O., Ginkel M., Kremling A., Waschler R., Kienle A. and Gilles E.D. *Computer Aided Modeling of Chemical and Biological Systems: Methods, Tools and Applications*. In: *Industrial & Engineering Chemistry Research*, Vol. 44, No. 8, pp. 2579–2591, 2005.
- [11] Leifheit J., Heine T., Kawohl M. and King R. *Rechnergestützte halbautomatische Modellierung biotechnologischer Prozesse*. In: *Automatisierungstechnik*, Vol. 55, Issue 5, pp. 211–218, 2007.
- [12] Mishra B. *A Symbolic Approach to Modeling Cellular Behavior*. In: *Proc. HiPC 2002*, pp. 725–732, Springer-Verlag, 2002.
- [13] Antoniotti M., Policriti A., Ugel N. and Mishra B.: *Model Building and Model Checking for Biological Processes*. In: *Cell Biochemistry and Biophysics*, Vol. 38, No. 3, pp. 271–286, 2003.
- [14] Ginkel M., Kremling A., Nutsch T., Rehner R. and Gilles E. D. *Modular modeling of cellular systems with ProMoT/Divi*. In: *Bioinformatics*, Vol. 19, No. 9, pp. 1169–1176, 2003.
- [15] King R., Leifheit J. and Freyer S. *Automatic identification of mathematical models of chemical and biochemical reaction systems*. In: *CHISA*, pp. 495–510, Prague, Czech Rep., 2002.
- [16] Pohlheim H. and Marenbach P.: *Generation of Structured Process Models Using Genetic Programming*. In: *Lecture Notes in Computer Science, AISB Workshop on Evolutionary Computing*, Vol. 1143, pp. 102–109, Springer-Verlag, 1996.
- [17] Bock H.G., Diehl M.M., Leineweber D.B. and Schlöder J.P. *A direct multiple shooting method for real-time optimization of nonlinear DAE processes*. *Nonlinear Model Predictive Control*, pp. 245–268, 2002.
- [18] Leifheit J. and King R. *Systematic structure and parameter identification for biological reaction systems supported by a software-tool*. In: 16th IFAC World Congress, Vol. 5, pp. 211–218, Prague, Czech Rep., 2005.
- [19] Fages F. and Soliman S. *Abstract interpretation and types for systems biology. Theoretical Computer Science*, Vol. 403, Issue 1, pp. 52–70, 2008.
- [20] Calzone L., Chabrier-Rivier N., Fages F. and Soliman S. *Machine learning biochemical networks from temporal logic properties*. In: *Transactions on Computational Systems Biology VI*, Vol. 4220, pp. 68–94, Springer-Verlag, 2006.
- [21] Voit E. O.: *Canonical Nonlinear Modeling, S-system Approach to Understanding Complexity*. Van Nostrand Reinhold, New York, 1991.
- [22] Violet N., Heine T., Rossner N., Valentin M., Böckelmann U., Szwedzyk U. and King R. *Usage of a software tool for automatic modeling and application of the results to the design of fermentations with Paenibacillus polymyxa*. *European BioPerspectives*, 2008.

Corresponding author: Rudibert King

Berlin Institute of Technology
 Chair of Measurement and Control, Sec. ER2-1,
 10623 Berlin, Hardenbergstrasse 36a, Germany
rudibert.king@tu-berlin.de

Received & Accepted: MATHMOD 2009

Revised: January 2010

Accepted: May 22, 2010

Comparison of Computational Methods for Reaction Equilibrium

Sanoja A. Jayarathna¹, Bernt Lie¹, Morten C. Melaaen^{1,2}

¹Telemark University College, Norway; ²Tel-Tek, Norway

SNE Simulation Notes Europe SNE 20(3-4), 2010, 75-82, doi: 10.11128/sne.20.tn.09999

Specie concentrations for chemical equilibrium can be found in various ways such as solving a system of algebraic equations or solving a dynamic model to steady state. The algebraic equation system for reacting systems consists of multivariate polynomials with multiple solutions. Some traditional and modern methods for solving such systems are discussed together with their advantages and disadvantages. The dynamic model results from ordinary differential equations (ODEs) based on dynamic material balances. It is shown how the ODE system contains all the information in the algebraic equation system. Based on the comparison of methods, it is suggested that solving the dynamic model to steady state in many ways is the simplest way for computing speciation data at the equilibrium of a reacting system.

Introduction

Reactive systems are widely used in the process industry, including the CO_2 capturing field. Equilibrium concentrations of the species available in the reactive systems are considered as an important source of information regarding the reaction kinetics. Computation of the equilibrium concentrations with known equilibrium coefficients is important for checking the equilibrium with the experimental data. Equilibrium concentrations are important also for the modelling of the systems.

In chemical engineering, the values of the equilibrium concentrations are often found by solving a set of algebraic equations, [1]. The algebraic equation set consists of the relations from setting the reaction rates to zero in tandem with the atom or charge balances; the latter ensure mass conservation. Alternatively, a method like minimization of Gibbs free energy ([2], [3], [4] & [5]) can be used. Usually such methods also result a set of equations to solve for the equilibrium compositions. The set of equations to be solved is usually a set of non-linear polynomials in several variables. A number of methods are applicable for solving such systems of non-linear polynomials with their own advantages and disadvantages. The use of some of these methods for solving for the equilibrium concentrations are discussed in this paper.

Alternatively, the dynamic mole balances (ODEs) can be solved until the steady state is reached. This gives the equilibrium concentrations.

The relationship between the algebraic equation system and the ODEs is discussed, including advantages and disadvantages of the two approaches.

The paper is organized as follows: first, the algebraic equations for chemical equilibrium are discussed, together with the possibility of multiple solutions. Methods for solving such algebraic equations are then briefly covered. Next, it is shown how dynamic models for concentrations are related to the algebraic equations, with examples. The methods are illustrated through computing a speciation curve which is relevant for post combustion CO_2 capturing. Finally, some conclusions are drawn and further work is discussed.

Nomenclature

c	Concentration	$[mol/m^3]$
MEA	Monoethanolamine	$[-]$
$MEACOO^-$	Protonated MEA	$[-]$
r	Rate of reaction	$[mol/s \cdot m^3]$
t	Time	$[s]$
ν	Stoichiometric matrix	$[-]$

1 Equilibrium Concentrations with Algebraic Equations

In chemical engineering, the equilibrium concentrations are often specified as the solution of

$$r(c) = 0 \quad (1)$$

$$Vc = b \quad (2)$$

where Vc indicates a linear combination of the elements of c . Here, $r(c) = 0$ is a nonlinear set of equations, while the linear equations $Vc = b$ typically are based on the idea of conservation of mass.

The use of minimization of Gibbs free energy is one of the alternative methods to find the equilibrium compositions in a reactive mixture. The numerical value of the standard state Gibbs energy of reaction (ΔG_T^0) is used to determine the reaction coordinate (ξ) at equilibrium [3]. The Eq. 3 provides the relation between the equilibrium coefficient, the ΔG_T^0 term and the component fugacities

$$K_a = \exp \left[\frac{-\Delta G_T^0}{RT} \right] = \prod \left[\frac{\hat{f}_i}{f_i^0} \right]^{v_i} \quad (3)$$

The reaction coordinates can be found by representing the component fugacities in terms of the mole fractions. With use of the known K_a 's and simplifications to the Eq. 3 a set of polynomials as presented by Eq. 4 will be resulted

$$a\xi^k = 0 \quad (4)$$

where $a\xi^k$ is a non-linear combination of the reaction coordinates when $k \neq 1$. Ultimately, the Gibbs free energy method can also result a set of non-linear polynomials to solve for the equilibrium compositions.

In either case, numerical methods should be used if the situation is complicated for normal hand calculations. Only the $r(c) = 0 \wedge Vc = b$ case will be used for the analysis in the rest of the paper while stating that the facts are common for all the similar cases.

Use of an iterative method like Newton's method is an option for solving a system of algebraic equations, [6]. A drawback of this method is that it gives only one possible solution among all possible solutions satisfying $r(c) = 0 \wedge Vc = b$ or $a\xi^k = 0$. The solution is dependent on the initial guess used to start the iterations.

Having a good guess will not always guarantee the expected results as systems of polynomials can have both stable and unstable solutions, as well as physically unrealistic solutions (negative concentrations or complex roots).

Sup-pose the reaction is a third order polynomial in the concentration of specie A, leading to the ODE

$$\frac{dc_A}{dt} = -kc_A^3 + c_A \quad (5)$$

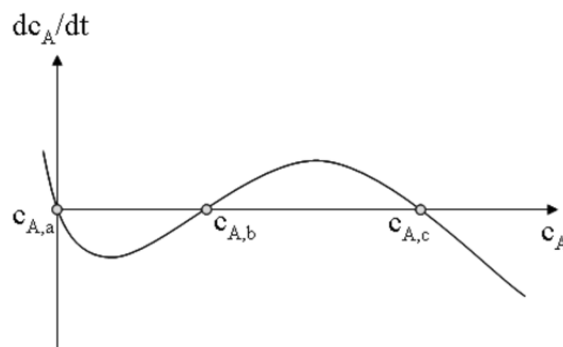


Figure 1. Possible solutions of the polynomial given by Eq. 5:

The possible solutions are $c_{A,a}$, $c_{A,b}$ and $c_{A,c}$ for c_A , which can be depicted as in Figure 1. Since dc_A/dt is negative between $c_{A,a}$ and $c_{A,b}$ the value of the concentration should decrease providing that any initial guesses between $c_{A,a}$ and $c_{A,b}$ will converge to $c_{A,a}$.

All the initial guesses between $c_{A,b}$ and $c_{A,c}$ will settle $c_{A,c}$ due to the positive values of dc_A/dt , leaving $c_{A,b}$ as an unstable solution. In general it is viable that the steady state solution of a dynamic model with polynomial reaction rate $r(c_A)$,

$$\frac{dc_A}{dt} = r(c_A)$$

is either complex (not physical), real and negative (not physically realizable), or real and positive. In the latter case, the solution may be stable or unstable. Only stable solutions are of interest.

Alternatively to Newton's method, a method such as Buchberger's algorithm for the Gröbner basis [7] can be used to solve the algebraic equation system. The Gröbner basis forms a "triangular" set of polynomial "bases" similar to triangular result of Gaussian elimination. As an illustrative example, consider the ammonia synthesis reaction, Eq. 6



The equilibrium coefficient ($K = 17.653$) is found in [3].

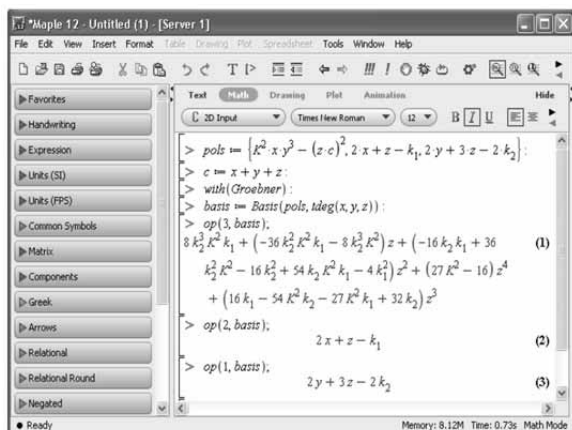


Figure 2. Calculation of Gröbner basis using Maple.

The initial N_2 and H_2 concentrations in the reactor are taken as k_1 and k_2 . The system of algebraic equations to solve for the equilibrium concentrations is given below, where Eq. 7 gives the equilibrium coefficient relating the concentrations. Nitrogen and hydrogen atom balances are given in Eqs. 8 and 9

$$K c_{N_2}^{1/2} + c_{N_2}^{3/2} + c_{NH_3} c_T = 0 \quad (7)$$

$$2c_{NT} + c_{NH_3} - 2c_{N_{2,in}} = 0 \quad (8)$$

$$2c_{H_2} + 3c_{NH_3} - 2c_{N_{2,in}} = 0 \quad (9)$$

where c_T is the total concentration given by Eq. 10

$$c_T = c_{N_2} + c_{H_2} + c_{NH_3} \quad (10)$$

The Gröbner bases can be found using Maple [8]; Figure 2 shows the steps in Maple when $c_{N_{2,n}}$ and $2c_{N_{2,in}}$ are denoted k_1 and k_{12} , respectively. In Figure 2, concentrations of N_2 , H_2 and NH_3 are represented by the symbols x , y and z , respectively.

The resulting equations imply that

$$\begin{aligned} 0 = & 8c_{N_{2,in}} c_{H_{2,in}}^3 K^2 \\ & + \left(-36c_{N_{2,in}} c_{H_{2,in}}^2 K^2 - 8c_{H_{2,in}}^3 K^2 \right) c_{NH_3} \\ & + \left(-16c_{N_{2,in}} c_{N_{2,in}} + 36c_{H_{2,in}}^2 K^2 - 16c_{H_{2,in}}^2 \right) \\ & + 54c_{N_{2,in}} c_{H_{2,in}} K^2 - 4c_{N_{2,in}}^2 c_{NH_3}^2 \\ & + \left(16c_{N_{2,in}} - 54c_{H_{2,in}} K^2 - 27c_{N_{2,in}} \right) K^2 \\ & + 32c_{H_{2,in}} c_{NH_3}^3 + \left(27K^2 - 16 \right) c_{NH_3}^4 \end{aligned}$$

$$c_{N_2} = \frac{1}{2} \left(c_{N_{2,in}} - c_{NH_3} \right)$$

$$c_{H_2} = c_{H_{2,in}} - \frac{3}{2} c_{NH_3}$$

From the equations above, c_{NH_3} is found by solving a fourth order polynomial in c_{NH_3} , which yields 4 solutions in the complex field; the solutions may be real positive or negative, or complex. The two subsequent equations give c_{N_2} and c_{H_2} directly from c_{NH_3} .

In conclusion, there are 4 triples $(c_{N_2}, c_{H_2}, c_{NH_3})$ of solutions. In order to be a physically realistic solution, every element in a triple must be real and positive. Furthermore, it is necessary to check whether the remaining physically realistic solutions are stable or unstable.

Finally, it must be determined which of the physically realistic and stable solutions belong to realistic initial concentrations $c(t = 0)$.

The disadvantages of the Gröbner basis method are: (i) much computer memory is required, (ii) the computational time is high, (iii) the method is numerically ill-posed with current algorithms, and (iv) some post treatment is required to select the physically correct solution. The advantage is that all solutions are found, and it is not necessary to “guess” any initial value in an iteration procedure.

In [9], it is recommended to instead use ideas from continuation/homotopy to find all solutions; continuation has better numerical properties than current implementations of the Gröbner basis method.

2 ODEs for Generating Equilibrium Concentrations

Solving an ODE system until steady state is reached, is an alternative to solving a set of polynomials to get the equilibrium compositions. Additional information such as forward and backward reaction rates are required, but as long as their ratio equals the equilibrium constant, the steady state equation should be correct.

As long as only the steady state solution is required, exists, and is unique, and ODE solver can be looked upon as a special kind of root finder.

Many other such methods exist, in addition to standard solvers such as the Newton method, see e.g. [10].

Batch reactors with perfect mixing and a constant volume have concentrations given by

$$\frac{dc}{dt} = v^T r \quad (11)$$

where $c \in \mathbb{R}^n$ is a vector of concentrations with elements c_j , $r(c) \in \mathbb{R}^m$ is a vector of overall reaction rates for the m reactions with elements r_i , and $v \in \mathbb{R}^{m \times n}$ is the stoichiometric matrix.

With a given initial condition $c(t_0)$, and assuming the existence of a steady state, the steady state can be found by solving this set of ODEs until $t \rightarrow \infty$, or for a sufficiently large time. The concentrations at the steady state gives the equilibrium concentrations and is easily found with known realistic initial conditions.

It is simple to show that these ODEs contain the same information as in the system of algebraic equations being discussed earlier.

2.1 Partitioning into Reaction Invariants and Reaction Variants

The following partitioning into reaction invariants and variants is discussed in [11]. Let us introduce fictitious species with concentration s_j which are linear combinations of the real species with concentration c_j . Stacking the concentrations s_j into vector s , s can then be written as

$$s = Mc$$

We can then formulate the dynamic model for s as

$$\frac{ds}{dt} = Mv^T r$$

As long as M is invertible, this differential equation for s holds exactly the same information as the differential equation for c in the previous subsection.

Let us now choose matrix M in a particular way: let M be composed of submatrices M_N and M_C ,

$$M = \begin{pmatrix} M_N^T \\ M_C^T \end{pmatrix}$$

where the columns of M_N lie in the nullspace $\mathcal{N}(v)$ of v , while the columns of M_C lie in the column space $\mathcal{C}(v^T)$ of v^T ; the nullspace $\mathcal{N}(v)$ of v consists of all possible vectors n such that $v \cdot n = 0$.

The column space of v^T contains all the linear combinations of the columns of v^T [12].

Linear algebra tells us that it is possible to choose columns in M_N and M_C such that M is invertible [12]. With this structure of M , the differential equation for s becomes

$$\begin{aligned} \frac{ds}{dt} &= Mv^T r = \begin{pmatrix} M_N^T \\ M_C^T \end{pmatrix} v^T r \\ &\Downarrow \\ \frac{ds_j}{dt} &= M_N^T v^T r \\ \frac{ds_v}{dt} &= M_C^T v^T r \end{aligned}$$

Here, s_j are the reaction invariant fictitious concentrations, while s_v are the reaction variant fictitious concentrations.

Since the columns of M_N , $m_N \in \mathcal{N}(v) \Leftrightarrow v \cdot m_{N,i} = 0$, we have

$$M_N^T v^T r = (v M_N)^T r = 0 \cdot r = 0$$

Thus,

$$\frac{ds_j}{dt} = 0 \Rightarrow s_j = \text{constant}$$

Furthermore

$$M_C^T v^T = (v M_C)^T \in \mathbb{R}^{\rho \times \rho}$$

is square and invertible with $\rho = \text{rank} v$ when v is of full rank; when v is not of full rank, superfluous reactions can be removed to ensure full rank of v . Since $M_C^T v^T$ is invertible, it follows that in steady state

$$\frac{ds_v}{dt} = M_C^T v^T r = 0 \Rightarrow r = 0$$

In conclusion, this shows that the steady state solution of the differential equation Eq. 11 for c is equivalent to the solution found by simultaneously setting

$$r(c) = 0$$

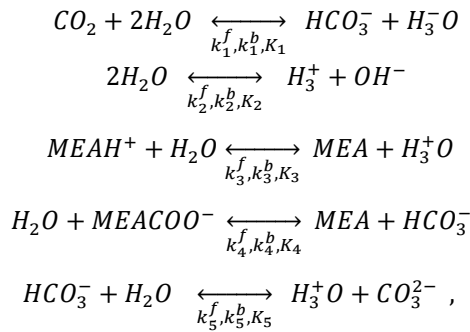
$$s_j = M_C^T v^T r = \text{constant}$$

Here, M_N^T corresponds to V in Eq. 2, while the constant corresponds to b .

Using linear algebra software such as Matlab, Maple, etc., M_N can easily be computed, while the constant is given from the initial conditions $c(t_0)$. But finding the solutions of this set of equations can be difficult as already explained.

2.2 Case Study

This example will illustrate how the ODEs are related with the algebraic equations using an example found in CO_2 capturing systems [13]:



where

$$K_i \triangleq \frac{k_i^f}{k_i^b}$$

Here K_i ; k_i^f and k_i^b are the equilibrium coefficient, forward reaction coefficient and backward reaction coefficient of reaction i , respectively. For the set of reactions considered, the stoichiometric matrix ν is

$$\nu = \begin{bmatrix} -1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & -2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & -2 \\ 0 & 1 & -1 & 0 & 0 & 0 & 1 & 0 & -1 \\ 0 & 1 & 0 & -1 & 1 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 1 & 1 & -1 \end{bmatrix}$$

when the species are considered in the order CO_2 , MEA , $MEAH^+$, $MEACOO^-$, HCO_3^- , OH^- , HCO_3^+ , CO_3^{2-} and H_2O . The overall reaction rates r_i of the system are given as

$$r_1 = k_1^f \cdot c_{CO_2} \cdot (c_{H_2O})^2 - k_1^b \cdot c_{HCO_3^-} \cdot c_{H_3O^+}$$

$$r_2 = k_2^f \cdot (c_{H_2O})^2 - k_2^b \cdot c_{OH^-} \cdot c_{H_3O^+}$$

$$r_3 = k_3^f \cdot c_{MEAH^+} \cdot c_{H_2O} - k_3^b \cdot c_{MEA} \cdot c_{HCO_3^-}$$

$$r_4 = k_4^f \cdot c_{MEACOO^-} \cdot c_{H_2O} - k_4^b \cdot c_{MEA} \cdot c_{HCO_3^-}$$

$$r_5 = k_5^f \cdot c_{HCO_3^-} \cdot c_{H_2O} - k_5^b \cdot c_{H_3O^+} \cdot c_{CO_3^{2-}}$$

In order to solve the dynamic model based on the mole balances, the forward and backward reaction coefficients are required. The values of the forward reaction coefficients can either be found in the literature (e.g. in [14]), or fixed at some chosen value.

With given forward rates, the backward reaction coefficients are chosen in such a way that the equilibrium coefficients are correct. The mass conservation of each specie contributes with an ODE to be solved in time. Since the reactor contains nine species, the ODE system consists of nine equations which can be stacked into a vector-matrix formulation as in Eq. 11.

The nullspace $\mathcal{N}(\nu)$ of ν and the column space $\mathcal{C}(\nu^T)$ of ν^T are found using the computer algebra system MuPAD within the word processor Scientific Workplace, and are given by the following basis vectors $b_{N,i}$ and $b_{C,i}$ respectively:

$$\mathcal{N}(\nu) = \left\{ \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 2 \\ -1 \\ 0 \\ 0 \\ 1 \\ -1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ -1 \\ -1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} -3 \\ 2 \\ 1 \\ 0 \\ -1 \\ 2 \\ 0 \\ 0 \\ 1 \end{bmatrix} \right\}$$

$$\mathcal{C}(\nu^T) = \left\{ \begin{bmatrix} -1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ -2 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ -2 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ -1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ -1 \\ 1 \\ 0 \\ 0 \\ 0 \\ -1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ 0 \\ 1 \\ 0 \\ -1 \end{bmatrix} \right\}$$

Using the linear combinations of basis vectors in the nullspace $\mathcal{N}(\nu)$ of ν the columns in M_N can be chosen.

The following column vectors of M_N are used here:

$$m_{N,1} = b_{N,2} - b_{N,1},$$

$$m_{N,2} = b_{N,1} - b_{N,3},$$

$$m_{N,3} = b_{N,3} - b_{N,1}$$

$$m_{N,4} = 2b_{N,1} + b_{N,2} + 3b_{N,3} + b_{N,4}.$$

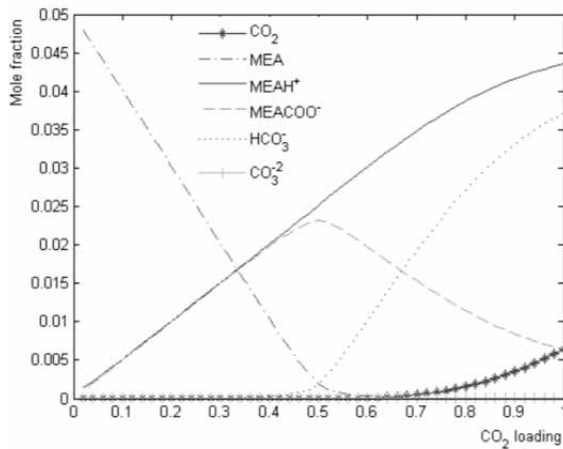


Figure 3. Speciation curves at 40 °C

Hence,

$$M_N = \begin{bmatrix} 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ -1 & 1 & 1 & 2 \\ -1 & 1 & 0 & 3 \\ -1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ -2 & 1 & 0 & 3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The columns of M_C are chosen to be the basis vectors of $\mathcal{C}(v^T)$, i.e. $m_{e,i} = 2b_{c,i}$.

$$M_C = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 1 & -1 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ -2 & 2 & -1 & -1 & -1 \end{bmatrix}$$

The reaction invariant set of equations is found for the fictitious species s_j

$$\begin{aligned} \frac{ds_j}{dt} &= M_N^T v^T r \\ \Downarrow M_N^T v^T r &= 0 \in \mathbb{R}^{4 \times 5} \\ \frac{ds_j}{dt} &= 0 \end{aligned}$$

Since $ds_j/dt = 0$, this shows that $s_j = \text{constant}$, or $V_c = b$.

The reaction variant equation set is found for the fictitious species s_v :

$$\frac{ds_v}{dt} = M_C^T v^T r$$

where matrix $M_C^T v^T$ is an invertible matrix, hence in steady state the condition $r(c) = 0$ is also satisfied.

This example shows that the ODEs, $\frac{dc}{dt} = v^T r$ contains all the information available from the system the system of algebraic equations, $r(c) = 0 \wedge Vc = b$.

3 Results of the ODEs

A dynamic model of the case study is simulated to steady state to find the equilibrium concentrations. The equilibrium coefficients for this reacting system can be found in e.g. [13], [15], and [16].

The dynamic model is simulated for a sufficiently long time and the steady state solution is taken as the equilibrium concentrations.

It can be shown that the equilibrium results depend on the initial concentrations of CO_2 and MEA , $c_{CO_2,in}$ and $c_{MEA,in}$, respectively. It is of interest to construct a so called speciation diagram, see Figure 3.

The speciation diagram shows the equilibrium mole fractions of the various species, as a function of the so called CO_2 loading, i.e. as a function of the fraction $c_{CO_2,in}/c_{MEA,in}$.

In the speciation diagram of Figure 3, a temperature of 40° C is assumed, and the results are in good agreement with the reference work [13].

4 Conclusions

The equilibrium concentrations of a reactor can be found by solving a set of algebraic equations satisfying Eqs. 1 and 2. In the case of polynomial algebraic equations, the roots lie in the complex field. However, only real and positive roots are physically realistic.

Furthermore, among the real and positive roots, some solutions represent unstable steady states, and the stable steady state roots are thus the only acceptable equilibrium concentrations.

A basic method for solving algebraic equations is Newton's method; this method only finds one root, and the root that is found depends on the chosen initial guess c^0 . There is no guarantee that the chosen c^0 leads to the most physically realistic solution. To study all solutions using Newton's method, it is necessary to use the method repeatedly, each time using a different value of c^0 .

When the algebraic equations are polynomials in several variables, the algebraic equations can be transformed to Gröbner bases using Buchberger's algorithm. Thereafter, all possible sets of solutions can be found using a standard solver for single variable polynomials.

Finally, it is necessary to postprocess the solutions in order to find the physically acceptable equilibrium concentrations. However, Buchberger's algorithm is quite demanding when it comes to computer memory and computation time. Thus, it may be better to use continuation/homotopy methods to find all solutions.

Another alternative is to solve the dynamic model of a batch reactor to steady state. Comparing the dynamic model $\frac{dc}{dt} = v^T r$ with the algebraic equations, it is easily shown that the dynamic model contains all the information from the algebraic equation system $r(c) = 0 \wedge Vc = b$. Additionally, dynamics of the reaction system is included in the dynamic model.

Clearly, simulating an ODE to steady state where we do not care about the transient behavior, is similar to using e.g. Newton's method; the ODE solver can be considered as just another root solver. Ensuring that steady state is reached with the ODE solver, is comparable to ensuring that the Newton solver has converged to the root. In both cases, we iterate on $v^T r(c^0)$ such that $v^T r(c^0) \rightarrow 0$.

The advantage of using a physically realistic dynamic model with an ODE solver, is that we know that the dynamic model has the equilibrium solution as the steady state solution, and that we can choose a realistic initial value for the ODEs such that the equilibrium solution is found.

Clearly, ODEs exist with complex dynamics such as limit cycle behavior where no steady state solution exist, or with bifurcations where the steady state solution is extremely sensitive to the chosen initial value or numerical inaccuracies. But these complex dynamic cases will also be difficult to solve with alternative methods.

The above factors indicate that both solving the algebraic equations using Newton's method, and solving the ODEs using an ODE solver should be equally accurate in steady state.

Since it is just as simple to formulate the ODEs as formulating the algebraic equations, since it is simpler to suggest physically reasonable initial values than guessing an initial iterate for the Newton solver, and since the ODE solver automatically will lead to a stable steady state, we suggest that using an ODE solver is favorable.

Using an ODE solver to find the equilibrium concentrations has been illustrated through computing the speciation diagram relevant for post combustion CO₂ capturing using MEA.

References

- [1] Timo Wark (Jr), K. (1995). *Advanced Thermodynamics for Engineers*. McGraw-Hill Inc, Singapore.
- [2] Thermodynamics of Chemical Equilibrium (2009). See www.chem1.com/acad/webtext/thermeq/TE5.html.
- [3] Elliot, J.R., and Lira, C.T. (1999). *Introductory Chemical Engineering Thermodynamics*. Prentice-Hall, Upper Saddle river, NJ.
- [4] Smith, M.J., Van Ness, H.C. and Abbott, M.M. (1996). *Introduction to Chemical Engineering Thermodynamics, fifth ed.* McGraw-Hill Inc, Singapore.
- [5] Skogestad, S. (2009). *Chemical and Energy Process Engineering*. CRC Press, Taylor and Francis group, NW, USA.
- [6] Gerald, C. F., and Wheatley P. O. (1998). *Applied numerical Analysis*. Addison Wesley Publishing Company, USA.
- [7] Cox, D., Little, J., and O'Shea, D. (1992). *Ideals, Varieties, and Algorithms. An Introduction to Computational algebraic Geometry and Commutative Algebra*. Springer, New York.
- [8] Maple (2009). See www.maplesoft.com.

- [9] Bates, D. J., Beccuti, A. G., Fotiou, I. A. and Morari, M. (2008). *An Optimal Control Application in Power Electronics Using Numerical Algebraic Geometry*. Proceedings of American Control Conference, Seattle, Washington, USA.
- [10] Bhaya, A., and Kaszkurewicz, E. (2006). *Control Perspectives on Numerical Algorithms and Matrix Problems*. SIAM, Philadelphia.
- [11] Aris, R. (1989). *Elementary Chemical Reactor Analysis*. Dover, Mineola, New York
- [12] Strang, G. (2006). *Linear Algebra and its Applications, fourth ed.* Thomson, Belmont, California.
- [13] Liu, Y., Zhang, L., and Watanasiri, S. (1999). „Representing Vapour-Liquid Equilibrium for an Aqueous MEA-CO₂ System Using the Electrolyte Nonrandom-Two-Liquid model“. *Industrial Engineering Chemical Research*, volume 38, pp 2080-2090.
- [14] Tanaka, Y. (2002). “Water Dissociation in Ion-Exchange Membrane Electro Dialysis”. *Journal of Membrane Science*, volume 203, pp 227-244.
- [15] Ausgen, D. M., Rochelle, G. T., Peng, X. and Chen, C. C. (1989). “Model of Vapour-Liquid Equilibria for Aqueous Acid Gas-Alkanolamine Systems Using the Electrolyte-NRTL Equation.“ *Industrial Engineering Chemical Research*, volume 28, pp 1060-1073.
- [16] Kent, R. L. and Eisenberg, B. (1976). „Better Data for Amine Treating“. *Hydrocarbon Processing*, pp 87-90.

Corresponding author: Berndt Lie,
Telemark University College, Postboks 203,
3901 Porsgrunn, Norway
Bernt.Lie@hit.no

Received & Accepted: SIMS 2009:-
Revised: March 2010 -
Accepted: May 22, 2010

SNE News Section

Data & Quick Info



EUROSIM 2013

8th EUROSIM Congress on Modelling and Simulation

The City Hall, Cardiff, Wales, United Kingdom 10-13 September 2013

www.eurosim2013.info

Contents

Info EUROSIM	2
Info EUROSIM Societies	3 - 6
Info ASIM, CROSSIM	3
Info CSSS, HSS	
DBSS, FRANCOSIM	4
Info ISCS, PSCS, SIMS, SLOSIM	5
Info UKSIM, LSS	
CAE-SMSG, ROMSIM	6
Reports of EUROSIM Societies	7 - 11
Report ASIM.....	7
Report SLOSIM	9
Report SIMS	10
Report PSCS	11
European Simulation Groups: YSS, KA-SIM / KA-CASE.....	12

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.

eSNE - SNE at Web SNE issues are also available at www.argesim.org as eSNE – *Electronic SNE*. Web-resolution eSNEs are free for download. Subscribers, e.g. members of EUROSIM Societies have access to SNE Archive with high-resolution eSNE copies and with sources of ARGESIM Benchmarks.

This *EUROSIM Data & Quick Info* compiles data from EUROSIM and EUROSIM societies: addresses, weblinks, officers of societies with function and email, to be published regularly in SNE issues – independent of individual reports of the societies.

SNE Reports Editorial Board

EUROSIM

Mikuláš Alexík, alexik@frtk.utc.sk
Borut Zupančič, borut.zupancic@fe.uni-lj.si
Felix Breitenecker, Felix.Breitenecker@tuwien.ac.at

ASIM: Thorsten Pawletta, pawel@mb.hs-wismar.de
CROSSIM: Vesna Dušak, vdusak@foi.hr
CSSS: Mikuláš Alexík, alexik@frtk.utc.sk
DBSS: A. Heemink, a.w.heemink@its.tudelft.nl
FRANCOSIM: Y. Hamam, y.hamam@esiee.fr
HSS: András Jávör, javor@eik.bme.hu
ISCS: M. Savastano, mario.savastano@unina.it
PSCS: Zenon Sosnowski, zenon@ii.pb.bialystok.pl
SIMS: Esko Juuso, esko.juuso@oulu.fi
SLOSIM: Rihard Karba, rihard.karba@fe.uni-lj.si
UKSIM: Richard Zobel, r.zobel@ntlworld.com
CAE-SMSG: Emilio Jimenez, emilio.jimenez@unirioja.es
LSS: Yuri Merkurjev, merkur@itl.rtu.lv
ROMSIM: Florin Stanculescu, sflorin@ici.ro

ARGESIM

Felix Breitenecker, Felix.Breitenecker@tuwien.ac.at
Anna Mathe, Anna.Mathe@tuwien.ac.at
Nikolas Popper, Niki.Popper@drahtwarenhandlung.at

INFO:

→ www.sne-journal.org
✉ office@sne-journal.org
→ www.eurosim.info

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 thirteen full members and one observer member:

ASIM	Arbeitsgemeinschaft Simulation <i>Austria, Germany, Switzerland</i>
CEA-SMSG	Spanish Modelling and Simulation Group <i>Spain</i>
CROSSIM	Croatian Society for Simulation Modeling <i>Croatia</i>
CSSS	Czech and Slovak Simulation Society <i>Czech Republic, Slovak Republic</i>
DBSS	Dutch Benelux Simulation Society <i>Belgium, Netherlands</i>
FRANCOSIM	Société Francophone de Simulation <i>Belgium, France</i>
HSS	Hungarian Simulation Society <i>Hungary</i>
ISCS	Italian Society for Computer Simulation <i>Italy</i>
LSS	Latvian Simulation Society <i>Latvia</i>
PSCS	Polish Society for Computer Simulation <i>Poland</i>
SIMS	Simulation Society of Scandinavia <i>Denmark, Finland, Norway, Sweden</i>
SLOSIM	Slovenian Simulation Society <i>Slovenia</i>
UKSIM	United Kingdom Simulation Society <i>UK, Ireland</i>
ROMSIM	Romanian Society for Modelling and Simulation, <i>Romania, Observer Member</i>

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 Fritzon (SIMS)
petfr@ida.liu.se

Treasurer Felix Breitenecker (ASIM)
felix.breitenecker@tuwien.ac.at

SNE Repres. Felix Breitenecker
felix.breitenecker@tuwien.ac.at

SNE – Simulation News Europe. 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*. EUROSIM societies are offered to distribute to their members the journal *Simulation News Europe* (SNE) as official membership journal. SNE Publisher are EUROSIM, ARGESIM and ASIM.

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

→ www.sne-journal.org, menu SNE

✉ office@sne-journal.org

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.

Chairs EUROSIM Miroslav Šnorek (CSSS)
2010 *snorek@fel.cvut.cz*

Mikulas Alexik (CSSS)
alexik@frtk.utc.sk

Organisation *chairs@eurosim2010.org*
EUROSIM 2010 *info@eurosim2010.org*
actionm@action-m.com

Information Mikulas Alexik (CSSS)
CSSS *alexik@frtk.utc.sk*

→ www.eurosim2010.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 Officers

President	Felix Breiteneker felix.breiteneker@tuwien.ac.at
Vice presidents	Sigrid Wenzel, s.wenzel@uni-kassel.de T. Pawletta, pawel@mb.hs-wismar.de
Secretary	Anna Mathe, anna.mathe@tuwien.ac.at
Treasurer	I. 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. Breiteneker, felix.breiteneker@tuwien.ac.at
Universities / Research Inst.	S. Wenzel, s.wenzel@uni-kassel.de W. Wiechert, W.Wiechert@fz-juelich.de J. Haase, Joachim.Haase@eas.iis.fraunhofer.de Katharina Nöh, k.noeh@fz-juelich.de
Industry	S. Wenzel, s.wenzel@uni-kassel.de K. Panreck, Klaus.Panreck@hella.com
Conferences	Klaus Panreck Klaus.Panreck@hella.com A. Gnauck, albrecht.gnauck@tu-cottbus.de
Publications	Th. Pawletta, pawel@mb.hs-wismar.de Christina Deatcu, christina.deatcu@hs-wismar.de F. Breiteneker, felix.breiteneker@tuwien.ac.at
Repr. EUROSIM	F. Breiteneker, felix.breiteneker@tuwien.ac.at N. Popper, niki.popper@drahtwarenhandlung.at
Education / Teaching	Ch. Deatcu, christina.deatcu@hs-wismar.de N. Popper, niki.popper@drahtwarenhandlung.at Katharina Nöh, k.noeh@fz-juelich.de
International Affairs	H. Szczerbicka, hsz@sim.uni-hannover.de O. Rose, Oliver.Rose@tu-dresden.de
Editorial Board SNE	T. Pawletta, pawel@mb.hs-wismar.de Ch. Deatcu, christina.deatcu@hs-wismar.de
Web EUROSIM	Anna Mathe, anna.mathe@tuwien.ac.at

Last data update December 2009

ASIM Working Groups. ASIM, part of GI - Gesellschaft für Informatik, is organised in Working Groups, dealing with applications and comprehensive subjects:

ASIM Working Groups

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 Katharina Nöh, k.noeh@fz-juelich.de

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

✉ vdusak@foi.hr

✉ CROSSIM / Vesna Dušak
Faculty of Organization and Informatics Varaždin, University of Zagreb
Pavlinska 2, HR-42000 Varaždin, Croatia

CROSSIM Officers

President	Vesna Dušak, vdusak@foi.hr
Vice president	Jadranka Božikov, jbozikov@snz.hr
Secretary	Vesna Bosilj-Vukšić, vbosilj@efzg.hr
Executive board members	Vlatko Čerić, vceric@efzg.hr Tarzan Legović, legovic@irb.hr
Repr. EUROSIM	Vesna Dušak, vdusak@foi.hr
Edit. Board SNE	Vesna Dušak, vdusak@foi.hr
Web EUROSIM	Jadranka Božikov, jbozikov@snz.hr

Last data update March 2009



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

✉ snorek@fel.cvut.cz

✉ CSSS / Miroslav Šnorek, CTU Prague
FEE, Dept. Computer Science and Engineering,
Karlovo nám. 13, 121 35 Praha 2, Czech Republic

CSSS Officers

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

Last data update December 2008

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 Pierreal, pierreal@imfa.fr*
- Pole Modelling and simulation of continuous systems. Pole Contact: *Yskandar Hamam, y.hamam@esiee.fr*

→ www.eurosim.info

✉ y.hamam@esiee.fr

✉ FRANCOSIM / Yskandar Hamam
Groupe ESIEE, Cité Descartes,
BP 99, 2 Bd. Blaise Pascal,
93162 Noisy le Grand CEDEX, France

FRANCOSIM Officers

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

Last data update April 2006

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 with affiliated societies.

→ www.eurosim.info

✉ a.w.heemink@its.tudelft.nl

✉ DBSS / A. W. Heemink
Delft University of Technology, ITS - twi,
Mekelweg 4, 2628 CD Delft, The Netherlands

DBSS Officers

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

Last data update April 2006

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

✉ javor@eik.bme.hu

✉ HSS / András Jávör,
Budapest Univ. of Technology and Economics,
Sztoczek u. 4, 1111 Budapest, Hungary



HSS Officers

President	András Jávör, javor@eik.bme.hu
Vice president	Gábor Szűcs, szucs@itm.bme.hu
Secretary	Ágnes Vigh, vigh@itm.bme.hu
Repr. EUROSIM	András Jávör, javor@eik.bme.hu
Deputy	Gábor Szűcs, szucs@itm.bme.hu
Edit. Board SNE	András Jávör, javor@eik.bme.hu
Web EUROSIM	Gábor Szűcs, szucs@itm.bme.hu

Last data update March 2008

PSCS – Polish Society for Computer Simulation - update

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 257 members.

→ www.ptsk.man.bialystok.pl

✉ leon@ibib.waw.pl

✉ PSCS / Leon Bobrowski, c/o IBIB PAN,
ul. Trojdena 4 (p.416), 02-109 Warszawa, Poland

PSCS Officers

President	Leon Bobrowski, leon@ibib.waw.pl
Vice president	Andrzej Grzyb, Tadeusz Nowicki
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
PSCS Board Members	R. Bogacz, Z. Strzyzakowski Andrzej Tylikowski

Last data update March 2009

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.eurosims.info

✉ Mario.savastano@uniina.it

✉ ISCS / Mario Savastano,
c/o CNR - IRSIP,
Via Claudio 21, 80125 Napoli, Italy

ISCS Officers

President	M. Savastano, mario.savastano@uniina.it
Vice president	F. Maceri, Franco.Maceri@uniroma2.it
Repr. EUROSIM	F. Maceri, Franco.Maceri@uniroma2.it
Edit. Board SNE	M. Savastano, mario.savastano@uniina.it

Last data update April 2005

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

✉ esko.juuso@oulu.fi

✉ SIMS / SIMS/Esko Juuso, Department of Process and Environmental Engineering, 90014 Univ.Oulu, Finland

SIMS Officers

President	Esko Juuso, esko.juuso@oulu.fi
Treasurer	Vadim Engelson, vaden@ida.liu.se
Repr. EUROSIM	Esko Juuso, esko.juuso@oulu.fi Erik Dahlquist erik.dahlquist@mdh.se
Edit. Board SNE	Esko Juuso, esko.juuso@oulu.fi
Web EUROSIM	Vadim Engelson, vaden@ida.liu.se

Last data update December 2009

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.

→ www.slosim.si

✉ slosim@fe.uni-lj.si

✉ SLOSIM / Rihard Karba, Faculty of Electrical Engineering, University of Ljubljana,
Tržaška 25, 1000 Ljubljana, Slovenia

**SLOSIM Officers**

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	B. Zupančič, borut.zupancic@fe.uni-lj.si
Edit. Board SNE	Rihard Karba, rihard.karba@fe.uni-lj.si
Web EUROSIM	Milan Simcic, milan.simcic@fe.uni-lj.si

Last data update December 2009

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✉ david.al-dabass@ntu.ac.uk

- ✉ UKSIM / Prof. David Al-Dabass
Computing & Informatics,
Nottingham Trent University
Clifton lane, Nottingham, NG11 8NS
United Kingdom

UKSIM Officers

President	David Al-Dabass, david.al-dabass@ntu.ac.uk
Secretary	A. 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, rhc@maths.soton.ac.uk
Repr. EUROSIM	Richard Zobel, r.zobel@ntlworld.com
Edit. Board SNE	Richard Zobel, r.zobel@ntlworld.com

Last data update March 2009 (partially)

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→ simulacion@cea-ifac.es

- ✉ CEA-SMSG / María Jesús de la Fuente,
System Engineering and Automatic Control department,
University of Valladolid,
Real de Burgos s/n., 47011 Valladolid, SPAIN

CAE - SMSG Officers

President	María J. la Fuente, maria@autom.uva.es
Repr. EUROSIM	Emilio Jimenez, emilio.jimenez@unirioja.es
Edit. Board SNE	Emilio Jimenez, emilio.jimenez@unirioja.es

Last data update March 2009

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/✉ merkur@itl.rtu.lv

- ✉ LSS / Yuri Merkuryev, Dept. of Modelling and Simulation Riga Technical University
Kalku street 1, Riga, LV-1658, LATVIA

LSS Officers

President	Yuri Merkuryev, merkur@itl.rtu.lv
Repr. EUROSIM	Yuri Merkuryev, merkur@itl.rtu.lv
Edit. Board SNE	Yuri Merkuryev, merkur@itl.rtu.lv

Last data update December 2008

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.

→ www.ici.ro/romsim/✉ sflorin@ici.ro

- ✉ ROMSIM / Florin Stanculescu,
National Institute for Research in Informatics, Averescu
Av. 8 – 10, 71316 Bucharest, Romania

ROMSIM Officers

President	Florin Stanculescu, sflorin@ici.ro
Vice president	Florin Hartescu, flory@ici.ro Marius Radulescu, mradulescu@ici.ro
Secretary	Zoe Radulescu, radulescu@ici.ro
Repr. EUROSIM	Florin Stanculescu, sflorin@ici.ro
Deputy	Florin Hartescu, flory@ici.ro
Edit. Board SNE	Florin Stanculescu, sflorin@ici.ro

Last data update March 2009



EUROSIM 2013

8th EUROSIM Congress on Modelling and Simulation

The City Hall, Cardiff, Wales, United Kingdom 10-13 September 2013



EUROSIM Congresses are the most important modelling and simulation events in Europe. For EUROSIM2013, we are soliciting original submissions describing novel research and developments in the following (and related) areas of interest: Continuous, discrete (event) and hybrid modelling, simulation, identification and optimization approaches. Two basic contribution motivations are expected: M&S Methods and Technologies and M&S Applications. Contributions from both technical and non-technical areas are welcome.

Congress Topics

The EUROSIM 2013 Congress will include invited talks, parallel, special and the poster sessions. The Congress topics of interest include, but are not limited to:

Intelligent Systems and Applications
Hybrid and Soft Computing
Communication Systems and Networks
Case Studies, Emergent Technologies
Workflow Modelling and Simulation
Web-based Simulation
Security Modelling and Simulation
Computer Games and Simulation
Neural Networks, Fuzzy Systems &
Evolutionary Computation
Autonomous Mental Development
Bioinformatics and Bioengineering
Circuits, Sensors and Devices

e-Science and e-Systems
Image, Speech & Signal Processing
Human Factors and Social Issues
Industry, Business, Management
Virtual Reality, Visualization and
Computer Games
Internet Modelling, Semantic Web
and Ontologies
Computational Finance & Economics
Systems Intelligence and
Intelligence Systems
Adaptive Dynamic Programming and
Reinforcement Learning

Methodologies, Tools and
Operations Research
Discrete Event /RT Systems
Mobile/Ad hoc wireless
networks, mobicast, sensor
placement, target tracking
Control of Intelligent Systems
and Control Intelligence
Robotics, Cybernetics, Control
Engineering, & Manufacturing
Energy, Power, Transport,
Logistics, Harbour, Shipping
and Marine Simulation
Semantic & Data Mining

Congress Venue / Social Events

The Congress will be held in the historic and magnificent City Hall in the heart of Cardiff, the capital city of Wales. The Gala Dinner will be held in the main hall of the National Museum of Wales. Social activities include visits to Cardiff Castle and Caerphilly Castle.

Congress Team: K. Al-Begain, A. Orsoni, R. Zobel, R. Cant, D. Al-Dabass; kbegain@glam.ac.uk

Info: www.eurosim2013.info



*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[®]**