



ADVCOMP 2011

The Fifth International Conference on Advanced Engineering Computing and
Applications in Sciences

ISBN: 978-1-61208-172-4

November 20-25, 2011

Lisbon, Portugal

ADVCOMP 2011 Editors

Sigeru Omatu, Osaka Institute of Technology, Japan

Simon G. Fabri, University of Malta - Msida, Malta

ADVCOMP 2011

Foreword

The Fifth International Conference on Advanced Engineering Computing and Applications in Sciences [ADVCOMP 2011], held between November 20 and 25, 2011 in Lisbon, Portugal, brought together researchers from the academia and practitioners from the industry in order to address fundamentals of advanced scientific computing and specific mechanisms and algorithms for particular sciences. The conference contributions presented novel research in all aspects of new scientific methods for computing and hybrid methods for computing optimization, as well as advanced algorithms and computational procedures, software and hardware solutions dealing with specific domains of science.

With the advent of high performance computing environments, virtualization, distributed and parallel computing, as well as the increasing memory, storage and computational power, processing particularly complex scientific applications and voluminous data is more affordable. With the current computing software, hardware and distributed platforms, effective use of advanced computing techniques is more achievable.

We take here the opportunity to warmly thank all the members of the ADVCOMP 2011 Technical Program Committee, as well as the numerous reviewers. The creation of such a broad and high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to ADVCOMP 2011. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the ADVCOMP 2011 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that ADVCOMP 2011 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the area of engineering computing and applications in science.

We are convinced that the participants found the event useful and communications very open. We also hope the attendees enjoyed the historic charm of Lisbon, Portugal.

ADVCOMP 2011 Chairs:

Jorge Ejarque Artigas
Chih-Cheng Hung
Sigeru Omatu
Helmut Reiser
Juha Röning
Erich Schweighofer

ADVCOMP 2011

Committee

ADVCOMP Advisory Chairs

Chih-Cheng Hung, Southern Polytechnic State University, USA
Juha Rönning, Oulu University, Finland
Sigeru Omatu, Osaka Institute of Technology, Japan
Erich Schweighofer, University of Vienna, Austria

ADVCOMP 2011 Research/Industry Chair

Jorge Ejarque Artigas, Barcelona Supercomputing Center (BSC-CNS), Spain
Helmut Reiser, Leibniz Supercomputing Centre (LRZ)-Garching, Germany

ADVCOMP 2011 Technical Program Committee

Witold Abramowicz, The Poznan University of Economics, Poland
Sónia Maria Almeida da Luz, Polytechnic Institute of Leiria, Portugal | University of Extremadura, Spain
Frederic Amblard, IRIT - Université Toulouse 1, France
Vincenzo Ambriola, Università di Pisa, Italy
Renato Amorim, University of London- Birkbeck, UK
Gabriel Amorós, Universitat de València, Spain
Regina B Araujo, Federal University of Sao Carlos, Brazil
Arne Bachmann, German Aerospace Center - Cologne, Germany
Sulieban Bani-Ahmad, Al-balqa Applied University - Salt, Jordan
Lotfi ben Othmane, Eindhoven Technical University, Netherlands
Simona Bernardi, Centro Universitario de la Defensa / Academia General Militar - Zaragoza, Spain
Ateet Bhalla, NRI Institute of Information Science and Technology - Bhopal, India
Pierre Borne, Ecole Centrale de Lille - Villeneuve d'Ascq, France
Kenneth P. Camilleri, University of Malta - Msida, Malta
Juan-Vicente Capella-Hernández, Universitat Politècnica de València, Spain
Antonio Casimiro Costa, University of Lisbon, Portugal
Marisa da Silva Maximiano, Polytechnic Institute of Leiria, Portugal | University of Extremadura - Cáceres, Spain
Vieri del Bianco, University College Dublin, Ireland
Javier Diaz, Indiana University, USA
Khalil El-Khatib, University of Ontario Institute of Technology, Canada
Sameh Elnikety, Microsoft Research, USA
Jorge Ejarque Artigas, Barcelona Supercomputing Center (BSC-CNS), Spain
Simon G. Fabri, University of Malta - Msida, Malta
Umar Farooq, Smart Technologies ULC - Calgary, Canada
Mehdi Farshbaf-Sahih-Sorkhabi, Azad University - Tehran / Fanavaran co., Tehran, Iran
Mohammad-Reza Feizi-Derakhshi, University of Tabriz, Iran
Leonardo Garrido, Tec de Monterrey, Mexico
Wolfgang Gentzsch, EU DEISA Project & Open Grid Forum, Germany
Luis Gomes, Universidade Nova de Lisboa, Portugal
Teofilo Gonzales, University of California, Santa Barbara, USA
Santiago Gonzalez de la Hoz, University of Valencia, Spain
Bernard Grabot, ENIT, France
Daniela Grigori, University of Versailles, France

Maki K. Habib, The American University in Cairo, Egypt
Jameleddine Hassine, King Fahd University of Petroleum & Mineral (KFUPM), Saudi Arabia
Eckhard M. S. Hitzer, University of Fukui, Japan
Wladyslaw Homenda, Warsaw University of Technology, Poland
Eduardo Huedo Cuesta, Universidad Complutense de Madrid, Spain
Chih-Cheng Hung, Southern Polytechnic State University, USA
Eshref Januzaj, Mali Information Technologies, Kosovo
Rajkumar Kannan, Bishop Heber College (Autonomous) - Trichy, India
Dimitrios A. Karras, Chalkis Institute of Technology, Hellas
Vasileios Karyotis, NTUA, Greece
Youngjae Kim, Oak Ridge National Laboratory / National Center for Computational, USA
William Knottenbelt, Imperial College London, UK
Evangelos Kranakis, Carleton University, Canada
Danny Krizanc, Wesleyan University, USA
Markus Kunde, German Aerospace Center - Koeln, Germany
Luigi Lavazza, Università dell'Insubria - Varese, Italy
Clement Leung, Hong Kong Baptist University, Hong Kong
Cheng-Xian (Charlie) Lin, University of Tennessee - Knoxville, USA
Juan Pablo López-Grao, University of Zaragoza, Spain
Emilio Luque, Universidad Autónoma de Barcelona (UAB), Spain
Lau Cheuk Lung, Federal University of Santa Catarina, Brazil
Anthony A. Maciejewski, Colorado State University - Fort Collins, USA
Shikharesh Majumdar, Carleton University - Ottawa, Canada
Sunilkumar S. Manvi, REVA Institute of Technology and Management - Bangalore, India
Atif Memon, University of Maryland - College Park, USA
Seyedeh Leili Mirtaheri, Iran University of Science & Technology, Iran
Nathalie Mitton, INRIA Lille - Nord Europe, France
V́ctor Méndez Muńoz, University of Valencia, Spain
Mohamed Mohandes, King Fahd University of Petroleum & Minerals (KFUPM) - Dhahran, Saudi Arabia
Henning Müller, University Hospitals of Geneva, Switzerland
Camelia Muńoz-Caro, Universidad de Castilla-La Mancha, Spain
Adrian Muscat, University of Malta, Malta
Toan Nguyen, INRIA, France
Alfonso Nińo Ramos, Universidad de Castilla-La Mancha, Spain
Sigeru Omatu, Osaka Institute of Technology, Japan
Sascha Opletal, University of Stuttgart, Germany
Flavio Oquendo, European University of Brittany - UBS/VALORIA, France
Juan Pablo López-Grao, University of Zaragoza, Spain
Santodsh Pandey, Cisco Systems, Inc., USA
Laura Papaleo, University of Genova, Italy
Kunal Patel, Ingenuity Systems, USA
Witold Pedrycz, University of Alberta, Canada
Meikel Poess, Oracle, USA
Radu-Emil Precup, Politehnica University of Timisoara, Romania
Helmut Reiser, Leibniz Supercomputing Centre (LRZ)-Garching, Germany
Dolores Rexachs, Universidad Aut3noma de Barcelona (UAB), Spain
Harald Richter, TU Clausthal, Germany
Ivan Rodero, University of New Jersey/NSF Center for Autonomic Computing - Piscataway, USA
Dieter Roller, Universit4t Stuttgart, Germany
Juha R3ning, Oulu University, Finland
Jos3 Francisco Salt Cairols, University of Valencia, Spain
M. Necip Sahinkaya, University of Bath, UK
Kenneth Scerri, University of Malta, Malta

Stefan Schmid, TU Berlin & T-Labs, Germany
Bruno Schulze, National Laboratory for Scientific Computing - LNCC - Petropolis, Brazil
Erich Schweighofer, University of Vienna, Austria
Kewei Sha, Oklahoma City University, USA
Stelios Sotiriadis, University of Bedfordshire, UK
Ryszard Tadeusiewicz, AGH University of Science and Technology - Krakow, Poland
Saïd Tazi, LAAS-CNRS. Université Toulouse 1, France
Daniel Thalmann, EPFL - Lausanne, Switzerland
Simon Tsang, Telcordia Technologies, Inc. - Piscataway, USA
José Valente de Oliveira, Universidade do Algarve, Portugal
Zhi Wang, North Carolina State University - Raleigh, USA
Michael Zapf, Universität Kassel, Germany
Marek Zaremba, Université du Québec en Outaouais, Canada
Nadia Zerida, Paris 8 University, France

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission to reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

Computer Simulation of Steady State Emission and Absorption Spectra for Molecular Ring <i>Pavel Herman, David Zapletal, and Milan Horak</i>	1
Hybrid Walking Point Location Algorithm <i>Roman Soukal, Martina Malkova, Tomas Vomacka, and Ivana Kolingerova</i>	7
Virtual Environment in Civil Engineering: Construction and Maintenance of Buildings <i>Alcinia Z. Sampaio, Ana Rita Gomes, and Joana Prata</i>	13
An Electrical Circuits e-Tutor based on Symbolic and Qualitative Analysis <i>Jason Debono and Adrian Muscat</i>	21
Adaptive Free-form Deformation for the Modification of CAD/CAM Data <i>Alexei Sacharow, Tobias Surmann, and Dirk Biermann</i>	27
On Root Classification in Kinetic Data Structures <i>Tomas Vomacka and Ivana Kolingerova</i>	32
Advanced Space Filtering for the Construction of 3D Additively Weighted Voronoi Diagram <i>Michal Zemek, Martin Manak, and Ivana Kolingerova</i>	37
A Novel Approach for Detection of Copy-Move Forgery <i>Mengyu Qiao, Andrew Sung, Qingzhong Liu, and Bernardete Ribeiro</i>	44
Concurrent Differential Evolution for Uncertain Optimization Problems <i>Kiyoharu Tagawa and Takashi Ishimizu</i>	48
Energy-aware MPSoC with Space-sharing for Real-time Applications <i>Stefan Aust and Harald Richter</i>	54
Image Restoration by Revised Bayesian-Based Iterative Method <i>Sigeru Omatu and Hideo Araki</i>	60
Automatic Error Detection in Gaussian Processes Regression Modeling for Production Scheduling <i>Bernd Scholz-Reiter and Jens Heger</i>	66
Scalable Resource Provisioning in the Cloud Using Business Metrics <i>Wlodzimierz Funika and Pawel Koperek</i>	72
e-Reverse Logistics for Remanufacture-to-Order: An Online Auction Based and Multi-Agent System Supported	78

Solution <i>Bo Xing, Wen-Jing Gao, Kimberly Battle, Fulufhelo Nelwamondo, and Tshilidzi Marwala</i>	
e-RL: The Internet of Things Supported Reverse Logistics for Remanufacture-to-Order <i>Bo Xing, Wen-Jing Gao, Kimberly Battle, Fulufhelo Nelwamondo, and Tshilidzi Marwala</i>	84
Particle Swarm Optimization for Nonlinear Model Predictive Control <i>Julian Mercieca and Simon Fabri</i>	88
Extending Microsoft Project for Real-World Job-Shop Scheduling <i>Peter Steininger</i>	94
An Analysis of MOSIX Load Balancing Capabilities <i>Siavash Ghiasvand, Ehsan Mousavi Khaneghah, Sina Mahmoodi Khorandi, Seyedeh Leili Mirtaheeri, Najmeh Osouli Nezhad, Meisam Mohammadkhani, and Mohsen Sharifi</i>	100
Characterizing Energy Efficiency in I/O System for Scientific Applications <i>Javier Panadero, Sandra Mendez, Dolores Rexachs, and Emilio Luque</i>	106
Supervised Hybrid SOM-NG Algorithm <i>Mario J. Crespo-Ramos, Ivan Machon-Gonzalez, Hilario Lopez-Garcia, and Jose Luis Calvo-Rolle</i>	113
Building Virtual Private Clouds with Network-aware Cloud <i>Joao Soares, Jorge Carapinha, Marcio Melo, Romeu Monteiro, and Susana Sargento</i>	119

Computer Simulation of Steady State Emission and Absorption Spectra for Molecular Ring

Pavel Heřman*, David Zapletal† and Milan Horák‡

*Department of Physics, Faculty of Science
University of Hradec Králové, Hradec Králové, Czech Republic
Email: pavel.herman@uhk.cz

†Institute of Mathematics, Faculty of Economics and Administration
University of Pardubice, Pardubice, Czech Republic
Email: david.zapletal@upce.cz

‡Faculty of Education
University of Hradec Králové, Hradec Králové, Czech Republic
Email: milan.horak@uhk.cz

Abstract—Knowledge of optical properties could shed more light on initial ultrafast phases of bacterial photosynthesis. Software package *Mathematica* is used for computer simulation of absorption spectra and steady state fluorescence spectra of ring molecular system, which can model cyclic antenna unit LH2 of the bacterial photosystem from purple bacterium *Rhodospseudomonas acidophila*. Three different models of uncorrelated static disorder are included in our simulations: Gaussian disorder in local excitation energies, Gaussian disorder in nearest neighbour transfer integrals and Gaussian disorder in radial positions of molecules in the ring. Dynamic disorder, interaction with a bath, is also included in Markovian approximation. The cumulant-expansion method of Mukamel et al. is used for the calculation of spectral responses of the system with exciton-phonon coupling. Calculated fluorescence spectra are compared with measured one. Values of the interpigment interaction energy and unperturbed transition energy from the ground state for above mentioned static disorder types are given.

Keywords—LH2 ring; exciton; fluorescence spectrum; static disorder; dynamic disorder; *Mathematica*.

I. INTRODUCTION

Ultrafast initial phases of photosynthesis in purple bacteria have been thoroughly studied over the last years. The bacterial light-harvesting complexes contain pigments circularly arranged in a protein scaffold. The crystal structure of peripheral light-harvesting complex LH2 from *Rhodospseudomonas acidophila* has been determined with high resolution [1], [2]. LH2 is a highly symmetric ring of nine pigment-protein subunits, each containing two transmembrane polypeptide helices and three bacteriochlorophylls (BChl).

In what follows, we are dealing with ring-shaped unit with nonameric symmetry resembling LH2 ring from *Rhodospseudomonas acidophila* with a strong interaction J between BChl molecules. Therefore, in our theoretical approach, an extended Frenkel exciton states model is considered. In spite of extensive investigation, the role of the protein moiety in

governing the dynamics of the excited states has not been totally clear yet. At room temperature, the solvent and protein environment fluctuate with characteristic time scales ranging from femtoseconds to nanoseconds. The simplest approach is to substitute fast fluctuations by dynamic disorder and slow fluctuations by static disorder.

In several steps, we have extended the former investigations of static disorder effect on the anisotropy of fluorescence made by Kumble and Hochstrasser [3] and Nagarajan et al. [4]–[6] for LH2 rings. After studying the influence of dynamic disorder for simple systems (dimer, trimer) [7]–[9] we added the dynamic disorder effect to our model of LH2 ring by using a quantum master equation in Markovian [10] and non-Markovian limits [11]. We also studied influence of four types of uncorrelated static disorder [12], [13] (Gaussian disorder in local excitation energies, transfer integrals, radial positions of BChls and angular positions of BChls on the ring) and influence of correlated static disorder (elliptical deformation) [11]. Models of rings with different arrangement of optical dipole moments (radial arrangement) were also investigated [14]–[16].

Recently, we have focused on the modeling of the steady state fluorescence spectra for molecular rings with 18 molecules and tangentially arranged optical transition dipole moments [17]. Main goal of the present paper is the investigation of absorption and steady state fluorescence spectra for different types of uncorrelated static disorder. The results for above mentioned three types of uncorrelated static disorder are compared with experimental fluorescence profile [18]. The dynamic disorder - interaction with the phonon bath in Markovian approximation for low and room temperature is taken into account simultaneously.

The remainder of this article is organized as follows. In Section II, we give the model of LH2 ring and review the theory we have used. In Section III, we mention computational point of view and used software. In Section

IV, we give the results of our absorption and steady state fluorescence spectra simulations, and in Section V, we draw some conclusions.

II. PHYSICAL MODEL

The Hamiltonian of an exciton in the ideal ring coupled to a bath of harmonic oscillators reads

$$H^0 = H_{\text{ex}}^0 + H_{\text{ph}} + H_{\text{ex-ph}}. \quad (1)$$

Here

$$H_{\text{ex}}^0 = \sum_{m,n(m \neq n)} J_{mn} a_m^\dagger a_n \quad (2)$$

corresponds to an exciton, e.g., the system without any disorder. The operator a_m^\dagger (a_m) creates (annihilates) an exciton at site m , J_{mn} (for $m \neq n$) is the so-called transfer integral between sites m and n .

The second term in (1),

$$H_{\text{ph}} = \sum_q \hbar \omega_q b_q^\dagger b_q, \quad (3)$$

represents phonon bath in the harmonic approximation (the phonon creation and annihilation operators are denoted by b_q^\dagger and b_{-q} , respectively).

Last term in (1),

$$H_{\text{ex-ph}} = \frac{1}{\sqrt{N}} \sum_m \sum_q G_q^m \hbar \omega_q a_m^\dagger a_m (b_q^\dagger + b_{-q}), \quad (4)$$

describes exciton-phonon interaction, which is assumed to be site-diagonal and linear in the bath coordinates (the term G_q^m denotes the exciton-phonon coupling constant).

Inside one ring, the pure exciton Hamiltonian can be diagonalized using the wave vector representation with corresponding delocalized "Bloch" states α and energies E_α . Considering homogeneous case with only nearest neighbour transfer matrix elements

$$J_{mn} = J_0 (\delta_{m,n+1} + \delta_{m,n-1}) \quad (5)$$

and using Fourier transformed excitonic operators (Bloch representation)

$$a_\alpha = \sum_n e^{i\alpha kn}, \quad \alpha = (2\pi/N)l, \quad l = 0, \pm 1, \dots, \pm N/2, \quad (6)$$

the simplest exciton Hamiltonian in α - representation reads

$$H_{\text{ex}}^0 = \sum_\alpha E_\alpha a_\alpha^\dagger a_\alpha, \quad E_\alpha = -2J_0 \cos \alpha. \quad (7)$$

Influence of uncorrelated static disorder is modeled by:

- (I) the local excitation energy fluctuations $\delta \varepsilon_n$ (uncorrelated, Gaussian distribution, standard deviation Δ , $H_s^I = \sum_n \delta \varepsilon_n a_n^\dagger a_n$),
- (II) the transfer integral fluctuations δJ_{mn} (uncorrelated, nearest neighbour approximation, Gaussian distribution, standard deviation Δ_J , $H_s^{II} = \sum_{mn(m \neq n)} \delta J_{mn} a_m^\dagger a_n$),

- (III) fluctuations of the radial positions of molecules on the ring δr_n (uncorrelated, nearest neighbour approximation, Gaussian distribution, standard deviation Δ_r).

Hamiltonian of the uncorrelated static disorder H_s^x adds to the Hamiltonian of the ideal ring H_{ex}^0 .

The cumulant-expansion method of Mukamel et al. [19], [20] is used for the calculation of spectral responses of the system with exciton-phonon coupling. Absorption $OD(\omega)$ and steady-state fluorescence $FL(\omega)$ spectra can be expressed as

$$OD(\omega) = \omega \sum_\alpha d_\alpha^2 \times \times \text{Re} \int_0^\infty dt e^{i(\omega - \omega_\alpha)t - g_{\alpha\alpha\alpha\alpha}(t) - R_{\alpha\alpha\alpha\alpha}t}, \quad (8)$$

$$FL(\omega) = \omega \sum_\alpha P_\alpha d_\alpha^2 \times \times \text{Re} \int_0^\infty dt e^{i(\omega - \omega_\alpha)t + i\lambda_{\alpha\alpha\alpha\alpha}t - g_{\alpha\alpha\alpha\alpha}^*(t) - R_{\alpha\alpha\alpha\alpha}t}. \quad (9)$$

Here $\vec{d}_\alpha = \sum_n c_n^\alpha \vec{d}_n$ is the dipole strength of eigenstate α , c_n^α are the expansion coefficients of the eigenstate α in site representation and P_α is steady state population of the eigenstate α . The inverse lifetime of exciton state $R_{\alpha\alpha\alpha\alpha}$ [18] is given by the elements of Redfield tensor [21] (a sum of the relaxation rates between exciton states)

$$R_{\alpha\alpha\alpha\alpha} = - \sum_{\beta \neq \alpha} R_{\beta\beta\alpha\alpha}. \quad (10)$$

The g-function and λ -values in (8) and (9) are given by

$$g_{\alpha\beta\gamma\delta} = - \int_{-\infty}^\infty \frac{d\omega}{2\pi\omega^2} C_{\alpha\beta\gamma\delta}(\omega) \times \times \left[\coth \frac{\omega}{2k_B T} (\cos \omega t - 1) - i(\sin \omega t - \omega t) \right], \quad (11)$$

$$\begin{aligned} \lambda_{\alpha\beta\gamma\delta} &= - \lim_{t \rightarrow \infty} \frac{d}{dt} \text{Im} \{ g_{\alpha\beta\gamma\delta}(t) \} = \\ &= \int_{-\infty}^\infty \frac{d\omega}{2\pi\omega} C_{\alpha\beta\gamma\delta}(\omega). \end{aligned} \quad (12)$$

The matrix of the spectral densities $C_{\alpha\beta\gamma\delta}(\omega)$ in the eigenstate (exciton) representation reflects one-exciton states coupling to the manifold of nuclear modes. In what follows only diagonal exciton phonon interaction in site representation is used (see (4)), i.e., only fluctuations of pigment site energies are assumed and the restriction to completely uncorrelated dynamic disorder is applied. In such case each site (i.e., each chromophore) has its own bath completely uncoupled from the baths of the other sites. Furthermore it is assumed that these baths have identical properties [12], [22], [23]

$$C_{mm'n'}(\omega) = \delta_{mn} \delta_{mm'} \delta_{nn'} C(\omega). \quad (13)$$

After transformation to exciton representation we have

$$C_{\alpha\beta\gamma\delta}(\omega) = \sum_n c_n^\alpha c_n^\beta c_n^\gamma c_n^\delta C(\omega). \quad (14)$$

Several models of spectral density of the bath are used in literature [18], [24], [25]. In our present investigation we

have used the model of Kühn and May [24]

$$C(\omega) = \Theta(\omega) j_0 \frac{\omega^2}{2\omega_c^3} e^{-\omega/\omega_c} \quad (15)$$

which has its maximum at $2\omega_c$.

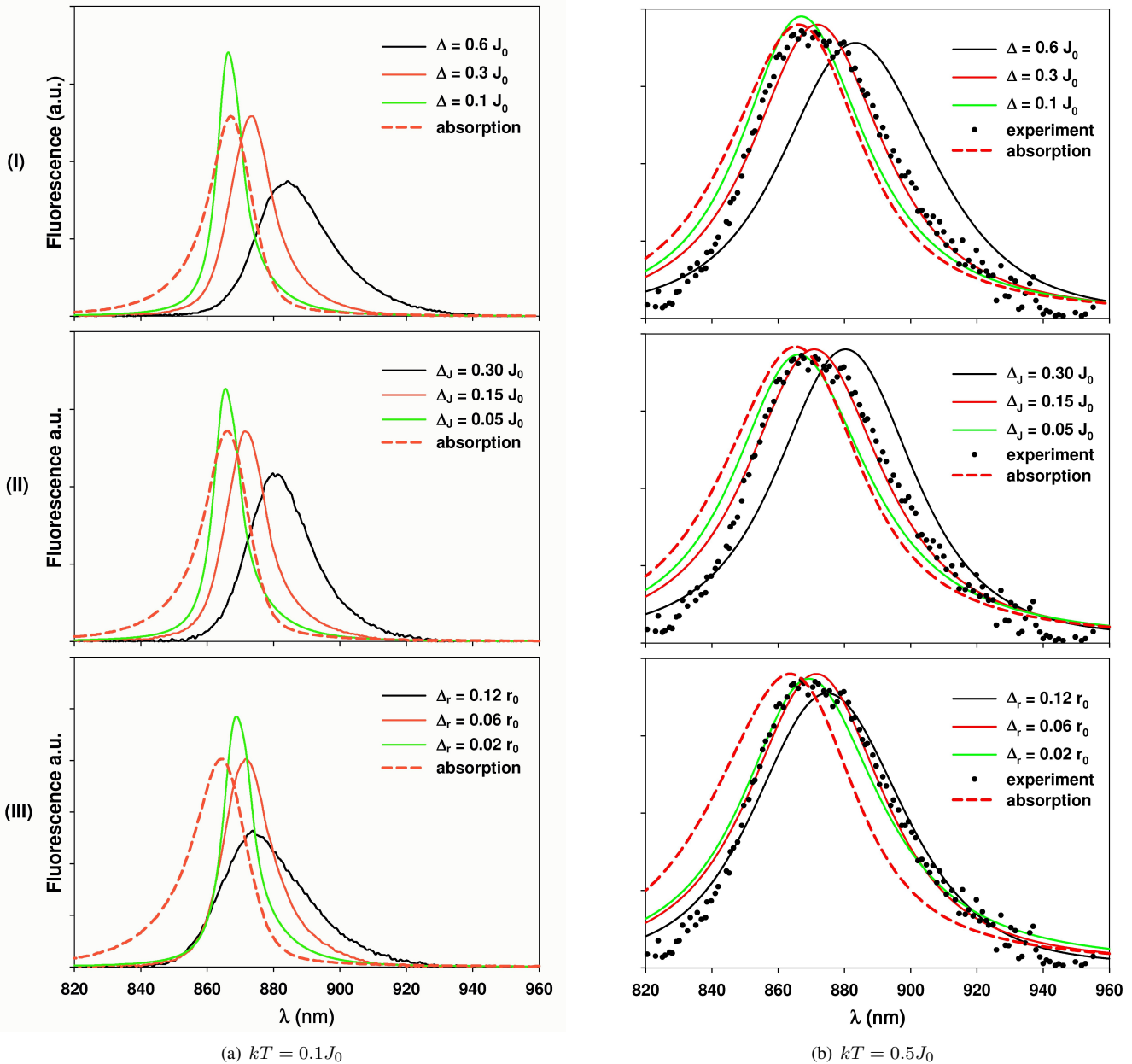


Figure 1. Resulting absorption spectra $OD(\omega)$ and steady-state fluorescence $FL(\omega)$ spectra of LH2 averaged over 2000 realizations of static disorder for three different types of static disorder and for two temperatures. Rows: (I) - Gaussian static disorder in local excitation energies $\delta\varepsilon$, (II) - Gaussian static disorder in nearest neighbour transfer integrals δJ , (III) - Gaussian static disorder in radial positions of molecules δr . Columns: (a) - low temperature $kT = 0.1J_0$, (b) - room temperature $kT = 0.5J_0$. Experimental fluorescence profile averaged in time and over the particles for room temperature [18] (points) is also displayed.

III. COMPUTATIONAL POINT OF VIEW

In the present paper, we are dealing with simulations of absorption spectra and steady state fluorescence spectra for molecular ring, which can model B850 ring from bacterial LH2 complex. Fluorescence $FL(\omega)$ and absorption $OD(\omega)$ spectra have to be simulated for large number of different static disorder realizations. Each realization of static disorder is created by random number generator. Then, so as to have

single ring fluorescence spectrum $FL(\omega)$ and absorption spectrum $OD(\omega)$, it is necessary to put through numerical integrations (see (8), (9)) for each realization of static disorder and finally average these results over all realizations of static disorder. For all calculation, the software package *Mathematica* [26] was used. This package is very convenient not only for symbolic calculations [27], that are needed for expression of all required quantities, but it can be used also for numerical ones [28]. That is why *Mathematica* was

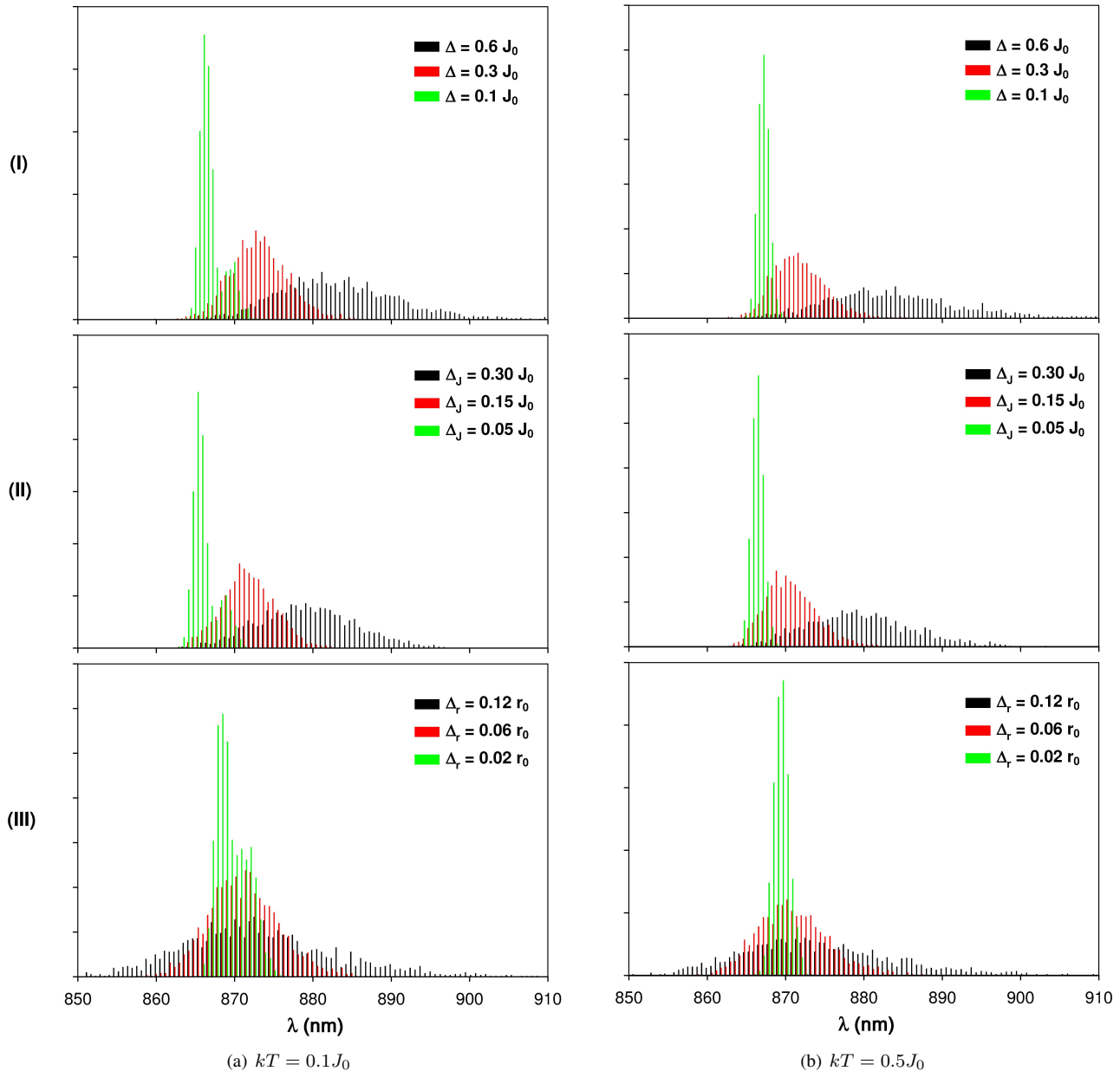


Figure 2. Peak position distributions of calculated steady-state single ring fluorescence spectra $FL(\omega)$ of LH2 for 2000 realizations of static disorder for three different types and three different strengths of static disorder and for two temperatures. Rows: (I) - Gaussian static disorder in local excitation energies $\delta\epsilon$, (II) - Gaussian static disorder in nearest neighbour transfer integrals δJ , (III) - Gaussian static disorder in radial positions of molecules δr . Columns: (a) - low temperature $kT = 0.5J_0$, (b) - room temperature $kT = 0.1J_0$.

used by us as for symbolic calculations as for numerical integrations and also for final averaging of results over all realizations of static disorder.

IV. RESULTS

Three above mentioned types of uncorrelated static disorder have been taken into account in our simulations simultaneously with dynamic disorder in Markovian approximation. Dimensionless energies normalized to the transfer integral $J_{12} = J_0$ have been used. Estimation of J_0 varies in literature between 250 cm^{-1} and 400 cm^{-1} . Contrary to Novoderezhkin et al. [18], different model of spectral density (the model of Kühn and May [12]) has been used. In agreement with our previous results [29] we have used $j_0 = 0.4 J_0$ and $\omega_c = 0.212 J_0$ (see (15)). The strengths of individual types of uncorrelated static disorder have been taken in agreement with [30]. For each type of static disorder, the simulations have been done for three values of the static disorder strength:

- (I) fluctuations in local excitation energies $\delta\varepsilon_n$:
 $\Delta = 0.1, 0.3, 0.6 J_0$,
- (II) fluctuations in transfer integrals δJ_{mn} :
 $\Delta_J = 0.05, 0.15, 0.30 J_0$,
- (III) fluctuations in radial positions of molecules on the ring δr_n :
 $\Delta_r = 0.02, 0.06, 0.12 r_0$, where r_0 is the radius of unperturbed ring.

Resulting steady state fluorescence spectra $FL(\omega)$ and absorption spectra $OD(\omega)$ averaged over 2000 realizations of each static disorder type and strength can be seen in Fig. 1 for low temperature ($kT = 0.1 J_0$, Fig. 1(a)) and for room one ($kT = 0.5 J_0$, Fig. 1(b)).

Comparison of calculated fluorescence spectra with experimental fluorescence profile averaged in time and over the particles at room temperature [18] is also displayed in Fig. 1(b). Simulated fluorescence spectra fitting has been done for middle values of static disorder strength. From this fitting we can determine values of the interpigment interaction energy J_0 and unperturbed transition energy from the ground state ΔE_0 for which the experimental data are best reproduced (the values are given in Section V).

Peak position of single ring fluorescence spectrum depend on the realization of static disorder and also on the temperature. To investigate this effect we calculate peak position distributions of simulated fluorescence spectra for all three above mentioned types of static disorder and for low ($kT = 0.1 J_0$) and room ($kT = 0.5 J_0$) temperature. For each static disorder type we consider three different static disorder strengths (see above). Calculated peak position distributions are shown in Fig. 2.

V. CONCLUSION

Software package *Mathematica* has been found by us very useful for the simulations of molecular ring spectra. For each

above mentioned type of static disorder the experimental fluorescence profile averaged in time and over the particles at room temperature [18] is best reproduced with:

- (I) local excitation energy fluctuations $\delta\varepsilon_n$ (for the standard deviation $\Delta = 0.3 J_0$):
 the interpigment interaction energy $J_0 = 370 \text{ cm}^{-1}$ and unperturbed transition energy from the ground state $\Delta E_0 = 12280 \text{ cm}^{-1}$,
- (II) transfer integral fluctuations δJ_{mn} (for the standard deviation $\Delta_J = 0.15 J_0$):
 $J_0 = 400 \text{ cm}^{-1}$ and $\Delta E_0 = 12350 \text{ cm}^{-1}$,
- (III) fluctuations of radial positions of molecules on the ring δr_n ($\Delta_r = 0.06 r_0$):
 $J_0 = 400 \text{ cm}^{-1}$ and $\Delta E_0 = 12300 \text{ cm}^{-1}$.

The fluorescence spectra shift to higher wavelengths (to lower energies) for increasing static disorder (it can be seen from Fig. 1). This shift is the smallest in case of static disorder in radial positions of molecules (III).

In Fig. 1 the shift of fluorescence spectra peak position to the higher wavelength in comparison with absorption spectra peak position is also visible. This shift is highest (8 nm) for the static disorder type (III).

If the temperature is changed from $kT = 0.5 J_0$ to $0.1 J_0$, the peak position distributions of single ring fluorescence spectra do not alter very much (Fig. 2). At low temperature we can see enlargement of the distribution to the higher wavelengths for all types of static disorder (mainly for small strength of static disorder). At room temperature, this effect is covered by larger widening of single ring spectra caused by dynamic disorder.

ACKNOWLEDGMENT

Support from the Faculty of Science, University of Hradec Králové (project of specific research No. 2116/2011 - P. Heřman and M. Horák) is gratefully acknowledged.

REFERENCES

- [1] G. McDermott, S. M. Prince, A. A. Freer, A. M. Hawthornthwaite, M. Z. Papiz, R. J. Cogdell, and N. W. Isaacs, "Crystal-structure of an integral membrane light-harvesting complex from photosynthetic bacteria," *Nature*, vol. 374, 1995, pp. 517–521.
- [2] M. Z. Papiz, S. M. Prince, T. Howard, R. J. Cogdell, and N. W. Isaacs, "The structure and thermal motion of the B800-850 LH2 complex from *Rps. acidophila* at 2.0 Å over-circle resolution and 100 K: New structural features and functionally relevant motions," *J. Mol. Biol.*, vol. 326, Mar. 2003, pp. 1523–1538, doi:10.1016/S0022-2836(03)00024-X.
- [3] R. Kumble and R. Hochstrasser, "Disorder-induced exciton scattering in the light-harvesting systems of purple bacteria: Influence on the anisotropy of emission and band \rightarrow band transitions," *J. Chem. Phys.*, vol. 109, 1998, pp. 855–865.

- [4] V. Nagarajan, R. G. Alden, J. C. Williams, and W. W. Parson, "Ultrafast exciton relaxation in the B850 antenna complex of *Rhodobacter sphaeroides*," *Proc. Natl. Acad. Sci. USA*, vol. 93, 1996, pp. 13774–13779.
- [5] V. Nagarajan, E. T. Johnson, J. C. Williams, and W. W. Parson, "Femtosecond pump-probe spectroscopy of the B850 antenna complex of *Rhodobacter sphaeroides* at room temperature," *J. Phys. Chem. B*, vol. 103, 1999, pp. 2297–2309.
- [6] V. Nagarajan and W. W. Parson, "Femtosecond fluorescence depletion anisotropy: Application to the B850 antenna complex of *Rhodobacter sphaeroides*," *J. Phys. Chem. B*, vol. 104, 2000, pp. 4010–4013.
- [7] V. Čápek, I. Barvík, and P. Heřman, "Towards proper parametrization in the exciton transfer and relaxation problem: dimer," *Chem. Phys.*, vol. 270, 2001, pp. 141–156.
- [8] P. Heřman and I. Barvík, "Towards proper parametrization in the exciton transfer and relaxation problem. II. Trimer," *Chem. Phys.*, vol. 274, 2001, pp. 199–217.
- [9] P. Heřman, I. Barvík, and M. Urbanec, "Energy relaxation and transfer in excitonic trimer," *J. Lumin.*, vol. 108, 2004, pp. 85–89.
- [10] P. Heřman, U. Kleinekathöfer, I. Barvík, and M. Schreiber, "Exciton scattering in light-harvesting systems of purple bacteria," *J. Lumin.*, vol. 94–95, 2001, pp. 447–450.
- [11] P. Heřman and I. Barvík, "Non-Markovian effects in the anisotropy of emission in the ring antenna subunits of purple bacteria photosynthetic systems," *Czech. J. Phys.*, vol. 53, 2003, pp. 579–605.
- [12] P. Heřman, U. Kleinekathöfer, I. Barvík, and M. Schreiber, "Influence of static and dynamic disorder on the anisotropy of emission in the ring antenna subunits of purple bacteria photosynthetic systems," *Chem. Phys.*, vol. 275, 2002, pp. 1–13.
- [13] P. Heřman and I. Barvík, "Temperature dependence of the anisotropy of fluorescence in ring molecular systems," *J. Lumin.*, vol. 122–123, 2007, pp. 558–561, doi:10.1016/j.jlumin.2006.01.224.
- [14] P. Heřman and I. Barvík, "Coherence effects in ring molecular systems," *Phys. Stat. Sol. C*, vol. 3, No. 10, 2006, 3408–3413, doi:10.1002/pssc.200672119.
- [15] P. Heřman, D. Zapletal, and I. Barvík, "The anisotropy of fluorescence in ring units III: Tangential versus radial dipole arrangement," *J. Lumin.*, vol. 128, 2008, pp. 768–770, doi:10.1016/j.jlumin.2007.12.023.
- [16] P. Heřman, I. Barvík, and D. Zapletal, "Computer simulation of the anisotropy of fluorescence in ring molecular systems: Tangential vs. radial dipole arrangement," *Lecture Notes in Computer Science*, vol. 5101, 2008, pp. 661–670, doi:10.1007/978-3-540-69384-0_71.
- [17] P. Heřman, D. Zapletal, and J. Šlégr, "Comparison of emission spectra of single LH2 complex for different types of disorder," *Physics Procedia*, vol. 13, 2011, pp. 14–17, doi:10.1016/j.phpro.2011.02.004.
- [18] V. I. Novoderezhkin, D. Rutkauskas, and R. van Grondelle, "Dynamics of the emission spectrum of a single LH2 complex: Interplay of slow and fast nuclear motions," *Biophysical Journal*, vol. 90, 2006, pp. 2890–2902, doi:10.1529/biophysj.105.072652.
- [19] W. M. Zhang, T. Meier, V. Chernyak, and S. Mukamel, "Exciton-migration and three-pulse femtosecond optical spectroscopies of photosynthetic antenna complexes," *J. Chem. Phys.*, vol. 108, 1998, pp. 7763–7774.
- [20] S. Mukamel, *Principles of nonlinear optical spectroscopy*. New York: Oxford University Press, 1995.
- [21] A. G. Redfield, "The Theory of Relaxation Processes," *Adv. Magn. Reson.*, vol. 1, 1965, pp. 1–32.
- [22] D. Rutkauskas, V. Novoderezhkin, R. J. Cogdell, and R. van Grondelle, "Fluorescence spectroscopy of conformational changes of single LH2 complexes," *Biophysical Journal*, vol. 88, 2005, pp. 422–435, doi:10.1529/biophysj.104.048629.
- [23] D. Rutkauskas, V. Novoderezhkin, R. J. Cogdell, and R. van Grondelle, "Fluorescence spectral fluctuations of single LH2 complexes from *Rhodospseudomonas acidophila* strain 10050," *Biochemistry*, vol. 43, 2004, pp. 4431–4438, doi:10.1021/bi0497648.
- [24] V. May and O. Kühn, *Charge and Energy Transfer in Molecular Systems*. Berlin: Wiley-WCH, 2000.
- [25] O. Zerlauskienė, G. Trinkunas, A. Gall, B. Robert, V. Urbonienė, and L. Valkunas, "Static and Dynamic Protein Impact on Electronic Properties of Light-Harvesting Complex LH2," *J. Phys. Chem. B*, vol. 112, 2008, pp. 15883–15892, doi:10.1021/jp803439w.
- [26] S. Wolfram, *The Mathematica Book*, 5th ed., Wolfram Media, 2003.
- [27] M. Trott, *The Mathematica GuideBook for Symbolics*. New York: Springer Science+Business Media, Inc., 2006.
- [28] M. Trott, *The Mathematica GuideBook for Numerics*. New York: Springer Science+Business Media, Inc., 2006.
- [29] P. Heřman, D. Zapletal, and I. Barvík, "Computer simulation of the anisotropy of fluorescence in ring molecular systems: Influence of disorder and ellipticity," *Proc. IEEE 12th International Conference on Computational Science and Engineering (CSE '09)*, IEEE Press, vol. 1, 2009, pp. 437–442, doi:10.1109/CSE.2009.88.
- [30] P. Heřman, I. Barvík, and D. Zapletal, "Energetic disorder and exciton states of individual molecular rings," *J. Lumin.*, vol. 119–120, 2006, pp. 496–503, doi:10.1016/j.jlumin.2006.01.042.

Hybrid Walking Point Location Algorithm

Roman Soukal
Martina Málková
Tomáš Vomáčka
Ivana Kolingerová

*Department of Computer Science and Engineering
University of West Bohemia
Plzen, Czech Republic*

soukal@kiv.zcu.cz, mmalkov@kiv.zcu.cz, tvomacka@kiv.zcu.cz, kolinger@kiv.zcu.cz

Abstract—Finding which triangle in a planar triangular mesh contains a query point (so-called point location problem) is one of the most frequent tasks in computational geometry. Therefore, using an algorithm with the lowest possible complexity is appropriate. However, such complexity may be achieved only by using additional data structures, leading into algorithms that are more difficult to implement and have additional memory demands. A possible solution is to use walking algorithms, which are easier to implement. They either do not require any additional memory, or require only a small portion of it in order to achieve the lowest possible complexity as well. In this paper, we propose a new walking algorithm combining two existing approaches to provide speed, robustness and easy implementation, and compare it with the fastest representatives of walking algorithms. Experiments proved that our algorithm is faster than the fastest existing visibility and straight walk algorithms, and depending on the character of input data, either as fast as the orthogonal walk algorithms or faster.

Keywords—Algorithm design and analysis; Computational geometry; Computer graphics.

I. INTRODUCTION

Point location problem is a very frequent task in computational geometry problems, such as triangulation construction, morphing and terrain editing. In this text, we focus on point location algorithms for triangular meshes, since triangular meshes are the most common way of data representation and its manipulation. Other representations, such as convex or non-convex polygonal meshes, can be triangulated first to use these algorithms. The algorithms can also be used for terrain models represented by triangular meshes without any preprocessing only by not using the height information during the location.

The point location problem is defined as follows. For a given planar triangular mesh and a query point, the task is to find which triangle from the mesh geometrically contains the query point. Algorithms solving this problem can be divided into two groups: algorithms with and without additional data structures. The former concentrate on having the lowest time complexity possible, in this case $O(\log n)$ per query point (n is the number of vertices in the mesh). Despite their low complexity, these algorithms have some disadvantages: they

have additional memory demands, they are more difficult to implement, and they are often problematically modified to cover adding or deleting vertices. The latter group tries to avoid these disadvantages, but has a slightly higher, but still sublinear, complexity.

The name of walking algorithms has arisen from their operating principle. They use the triangle neighborhood relations to go via the triangles between the starting triangle and the one containing the query point. Such point location process is called a *walk*. The starting triangle may be arbitrary, however, its clever selection may radically shorten the length of the walk, therefore we will cover this topic as well. Walking algorithms do not need any additional data structures, they use only the neighborhood relations in the mesh, thus often they are more often chosen than the optimal time complexity solutions.

There exist several walking algorithms solving the location process, some are robust, others faster. In this paper, we propose a new walking algorithm combining two existing solutions in order to gain speed from the faster and still remain robust. The main idea of our approach is to compute such a transformation that the line connecting a selected vertex of the starting triangle and the query point is parallel with x -axis. This transformation is then used for the tested points throughout the walk to enable a cheap comparison of their position with respect to this line. Surprisingly, despite the use of transformations, the walk is still fast, because only the query point and the tested points (one per visited triangle) are transformed. Since the walk goes straight between the starting triangle and the query point, it cannot cross the border of a convex triangular mesh, which contributes to its robustness.

Section II presents the existing walking algorithms and a sophisticated selection of the starting triangle for the walk. Section III describes our new proposed algorithm, Section IV shows experiments comparing our solution with the existing algorithms. Section V summarizes the characteristics of our algorithm.

II. OVERVIEW

Point location by walking algorithms usually works in two steps: (1) selection of the initial triangle for the walk, and (2) using the neighborhood relationships between the triangles (*walking*) to find the target triangle, containing the query point.

A sophisticated selection of the initial triangle may radically improve the speed of the process. One way is to use some additional knowledge about the data, i.e., if we know the range of the mesh vertices coordinates, we can start all the locations in a triangle containing the point lying in the middle of the triangular mesh, used for instance by [1]. Or, we may know that the next query point will be close to the last one, in which case the best solution is to use the target triangle from the last location as the initial one for the next [11], [17].

Without any knowledge about the data, we may select the initial triangle as the nearest triangle from a set A of randomly chosen triangles from T , where $\|A\| \ll \|T\|$ [9]. Devroye et al. in [4] showed that such an improved straight walk achieves $O(\sqrt[3]{n})$ time complexity per one search for uniformly distributed points, Zhu in [18] came to the same complexity for the remembering stochastic walk.

Other solutions use some additional memory: [10] simplifies the triangular mesh and locate the points in the simplified version first, [13] introduces a bucketing method, which uses a uniform grid to quickly find a proper initial triangle. Some algorithms [14], [16] try to avoid the sensitivity of the original bucketing method on data uniformity by using adaptive structures instead of a uniform grid.

When we know the initial triangle, the walk may proceed. There are several algorithms solving this step. They can be divided into three groups: visibility, straight and orthogonal walks, according to the style how they determine the way of the walk.

Visibility walks use local “visibility” tests to determine the way of their walk. These tests look for such an edge that defines a line separating the query point and the third vertex of the triangle. The walk then moves across this edge to the neighborhood triangle.

The first visibility walk algorithm is called Lawson’s oriented walk [7]. The algorithm starts in the initial triangle and uses the 2D orientation test to move to its neighbors until it reaches the query point:

$$\textit{orientation2D}(\mathbf{t}, \mathbf{u}, \mathbf{v}) = \begin{vmatrix} u_x - t_x & v_x - t_x \\ u_y - t_y & v_y - t_y \end{vmatrix} \quad (1)$$

where points \mathbf{t} , \mathbf{u} define an oriented line and \mathbf{v} is the tested point. In each triangle, the algorithm tests the triangle edges until it finds an edge, where the third vertex of the triangle lies on the opposite side of the edge than the query point. Then, it crosses such an edge to the next triangle. If such an edge does not exist, the triangle containing the query point has been found.

The Lawson’s oriented walk algorithm tests edges of the current triangle in a deterministic order, depending on the arrangement of edges in triangles, generated during the construction of the triangulation. This leads to the fact that the walk may loop for non-Delaunay triangulations [3], [15]. [3] proposed an algorithm avoiding the loop by choosing the edges of the current triangle in a random order. This modification is called *stochastic*. Furthermore, since it is not necessary to test the edge incident to the previous triangle, the process was sped up by remembering this edge and skipping the test. This modification is called *remembering* and brings a significant speedup, since only one or two orientation tests are needed instead of up to three (except of course the first triangle, where all the three edges may be tested). As the second test is done only to find out whether we are in the target triangle, [6] suggested to speed up the process even more by testing only one edge for the first k steps. If the triangle is found within this k steps, we circle around it, so it is necessary to determine a proper k based on the input.

Straight walk algorithms do not use only the local comparisons to determine the way of the walk, but they use an oriented line $\overrightarrow{\mathbf{pq}}$, connecting one point \mathbf{p} (its choice depends on the particular solution) of the starting triangle with the query point \mathbf{q} and then pass all triangles intersected by this line.

The standard straight walk algorithm [3], [8] works in two steps: an initialization step and a straight walk step. In the initialization step, a point \mathbf{p} is chosen as any of the starting triangle vertices and a triangle intersected by the line segment $\overrightarrow{\mathbf{pq}}$ is found. The walk starts from this triangle, and in each step, it uses such a vertex of the current triangle that is opposite to the edge used to enter this triangle and finds out its position with respect to $\overrightarrow{\mathbf{pq}}$ (using the 2D orientation test). Based on its position, it selects which edge it should cross to the next triangle. Before crossing, it computes the orientation test for the point \mathbf{q} with respect to this edge. If the point \mathbf{q} is on the inner side of the edge, the final triangle has been found. Otherwise, the walk crosses the edge to the next triangle and continues.

[12] proposed a modification of this method simplifying the initialization step and speeding up the algorithm. Instead of the 2D orientation tests, an implicit line equation of $\overrightarrow{\mathbf{pq}}$ is used. The equation is precomputed in the initialization step along with an implicit equation of a line normal to $\overrightarrow{\mathbf{pq}}$ in \mathbf{q} , which is used to determine if there is a possibility that the target triangle has been found. However, the location of the target triangle is not precise, so the remembering stochastic walk algorithm is used for the short, final location (usually about 2 triangles, for more detail see Section IV).

Orthogonal walks first navigate along one coordinate axis and then along the other, which makes the local tests cheaper, since only components of the coordinates are compared during the walk. The walk is usually longer than

other walks, but its tests are much cheaper, which results in a faster location.

The original orthogonal walk [3] consists of three steps: an initialization step, a walk along the x -axis, and a walk along the y -axis. During the initialization step, any vertex \mathbf{p} of the starting triangle is chosen and two lines are defined: a horizontal line, containing this vertex and parallel to the x -axis, and a vertical line, containing the query point and parallel to the y -axis. A triangle intersected by the horizontal line and containing \mathbf{p} is found. The walk then follows this line in a similar manner as the straight walk, but with component comparisons only: y -values are used to determine the next triangle, x -values are used to determine if the triangle intersected by the vertical line is found. When it is done, the walk continues by following the vertical line, where y -values are used to determine the next triangles and x -values to determine the target triangle containing the query point. For a few final triangles in each walk direction, it uses 2D orientation tests for a precise location.

The original orthogonal walk has a significant drawback: it does not solve the case, when the walk crosses the border of the triangular mesh. If the horizontal walk crosses the border, the vertical walk starts from the last triangle and usually does not find the correct triangle.

[1] proposes a modification, where the initialization step is simplified, and the walk is sped up by using fewer comparisons. Instead of two comparisons determining when the target triangle may be found, in which case the original walk uses the 2D orientation test to be sure, it uses only one comparison. This way the walk may stop too early, but since the previous tests were not precise anyway, slightly less precision is not so important, and the Remembering stochastic walk algorithm (RSW - details see in [3]) is used for the last few steps. The use of RSW also solves the problem with the possibility of crossing borders of the triangular mesh, because in such a case, the algorithm does not end at the correct triangle, but the final location with RSW does. However, the final walk is then longer and slows down the whole location.

Figure 1 shows such a situation: the horizontal walk reaches the border at the triangle γ . Here, the algorithm switches to the vertical walk, where the vertical line is moved to the last tested point. The vertical walk stops at the triangle δ , because the y component of \mathbf{s}_j is higher then the one of \mathbf{q} . This should mean that the triangle contains \mathbf{q} or is close to the target triangle, but as the horizontal walk had to stop early, the triangle is still quite far. The original algorithm would stop here and would not locate the right triangle. Its modification uses the RSW algorithm at this step and therefore locates the right triangle, but for a higher time cost, because the RSW algorithm has more expensive tests.

The complexity of the presented algorithms has been proved only for their basic representatives, and also either

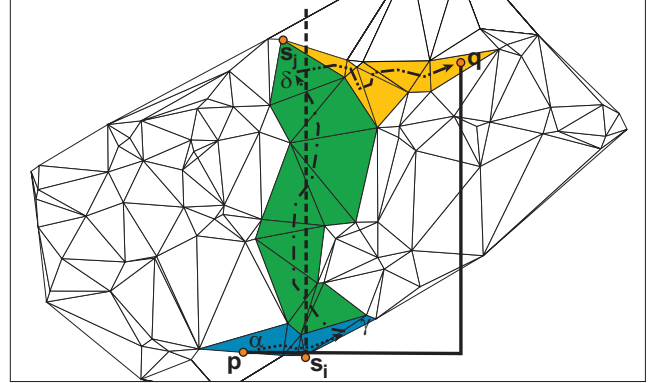


Figure 1. A case when the orthogonal walk crosses the border of the triangular mesh (a dotted line denotes the horizontal walk, a chain line denotes the vertical walk, a chain line with double dots denotes the final location by the RSW algorithm; the dashed and solid line denote the lines controlling the walk).

the time complexity or the number of visited triangles has been derived (note that these two values do not necessarily correspond). The stochastic walk has been shown to need $O(\sqrt{n \cdot \log n})$ expected time for uniform data [18]. The straight walk has been proved to visit $O(\sqrt{n})$ triangles in the expected case and uniform distribution [5], [10], a bound based on [2] shows that the orthogonal walk has similar complexity as the straight walk [4], [10].

III. THE PROPOSED ALGORITHM

The algorithm described in this paper is called a *Hybrid walk*, because it combines the basic idea of two walking strategies: straight and orthogonal walk, to keep the advantages of both. The algorithm works in four steps. In the initialization step, a point \mathbf{p} is chosen as any of the vertices of the starting triangle, and a transformation matrix M is set to transform the tested points in a way as if the line $\overrightarrow{\mathbf{p}\mathbf{q}}$ was parallel with the x -axis and $p'_x < q'_x$ (prime symbol denotes the transformed vertices, i.e., $\mathbf{p}' = \mathbf{p} \cdot M$). From this point, each tested vertex is first transformed, and then only its coordinate components are compared in the tests. In the next step, a triangle intersected by $\overrightarrow{\mathbf{p}\mathbf{q}}$ and containing \mathbf{p} is found. The third step is the walk itself, following the line $\overrightarrow{\mathbf{p}\mathbf{q}}$. The final, short location (about 2 triangles) is done by the RSW algorithm.

There exist many transformations meeting the previous requirements, to achieve the fastest computation possible, we chose the transformation matrix combining rotation by angle φ and scaling by k . The variables φ and k are determined by the mutual position of the points \mathbf{p} , \mathbf{q} . The equation used for transforming a vertex v is as follows:

$$\mathbf{v}' = (v_x, v_y) \cdot \begin{pmatrix} k \cdot \cos \varphi & k \cdot \sin \varphi \\ -k \cdot \sin \varphi & k \cdot \cos \varphi \end{pmatrix} \quad (2)$$

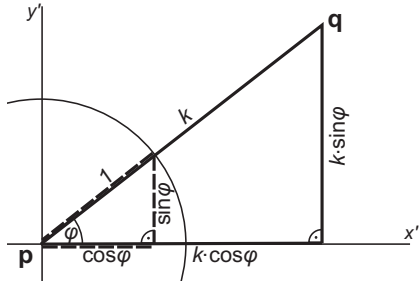


Figure 2. Components of the transformation matrix (x' is parallel with the x -axis, y' with the y -axis).

Computing sine and cosine of φ would be slow, but it can be avoided by using triangle similarity (see Figure 2), so $q_y - p_y$ and $q_x - p_x$ are computed instead of $k \cdot \sin \varphi$ and $k \cdot \cos \varphi$. Note that this fast computation of sine and cosine is the reason why k was introduced, otherwise $k = 1$ would be used.

Now, let us describe the algorithm in detail (for pseudocode, see Algorithm 1). In further text, we assume that the vertices of the triangles are in a CCW order. During the initialization step, the point \mathbf{p} needs to be selected as simply and fast as possible, because the tests done after the transformation are cheaper. Therefore, we choose any of the vertices of the starting triangle as \mathbf{p} (as is done in [3]) and compute the transformation matrix.

When the transformation matrix is set, the triangle δ intersected by $\overrightarrow{\mathbf{p}\mathbf{q}}$ and containing \mathbf{p} needs to be located. This is done in a similar manner as in the straight walk, but with cheaper tests thanks to the use of transformations. We select a different vertex than \mathbf{p} from the starting triangle, let us denote it \mathbf{r} , and compute the y -coordinate of its transformed version \mathbf{r}' . Then we turn around \mathbf{p} until we find the desired triangle. In each triangle, we determine which edge we should cross to the neighborhood triangle by comparing r'_y with q'_y . Therefore, during this step, only one transformed coordinate has to be computed and one coordinate component comparison per triangle is performed.

The walk starts from the triangle δ and follows the line $\overrightarrow{\mathbf{p}\mathbf{q}}$. In each triangle τ_i with vertices $\mathbf{l}_i, \mathbf{r}_i, \mathbf{s}_i$, the edge $\epsilon_{\mathbf{l}_i\mathbf{r}_i}$ is used to cross to this triangle, \mathbf{l}_i is to the left of $\overrightarrow{\mathbf{p}\mathbf{q}}$ and \mathbf{r}_i to the right. The edge to cross is determined by comparing the y components of \mathbf{s}'_i and \mathbf{q}' . If \mathbf{s}'_i is above $\overrightarrow{\mathbf{p}\mathbf{q}}$, we cross the edge $\epsilon_{\mathbf{r}_i\mathbf{s}_i}$, otherwise, we cross the edge $\epsilon_{\mathbf{l}_i\mathbf{s}_i}$. Note that if the line leaves the triangle through its vertex, the walk may continue by both $\epsilon_{\mathbf{r}_i\mathbf{s}_i}$ and $\epsilon_{\mathbf{l}_i\mathbf{s}_i}$, in the pseudocode we choose the latter one. Also the x -components of \mathbf{s}'_i and \mathbf{q}' are compared to end the walk if there is the possibility that the target triangle has been found. Therefore, during this step, both transformed components of \mathbf{s}_i have to be computed and two component comparisons per triangle are performed.

The triangle in which the walk ends does not necessarily

Input: the query point \mathbf{q} , the chosen starting triangle $\alpha \in T$
Output: the triangle ω which contains \mathbf{q}

```

// initialization step
triangle  $\tau = \alpha = \mathbf{lrs}$ ;
point  $\mathbf{p} = \mathbf{s}$ ;
if  $\mathbf{p} = \mathbf{q}$  then return  $\tau$ ;
vector  $\mathbf{a} = \mathbf{q} - \mathbf{p}$ ;
//  $k = \|\mathbf{a}\|$ 
double  $k\cos = a_x$ ;
double  $k\sin = a_y$ ;
 $q'_x = q_x \cdot k\cos - q_y \cdot k\sin$ ;
 $q'_y = q_x \cdot k\sin + q_y \cdot k\cos$ ;
double  $r'_y = r_x \cdot k\sin + r_y \cdot k\cos$ ;
if  $r'_y > q'_y$  then
    //  $\mathbf{r}$  is above  $\overrightarrow{\mathbf{p}\mathbf{q}}$ 
    double  $l'_y = l_x \cdot k\sin + l_y \cdot k\cos$ ;
    while  $l'_y > q'_y$  do
         $\tau =$  neighbor of  $\tau$  trough  $\epsilon_{pl}$ ;
         $\mathbf{r} = \mathbf{l}$ ;
         $\mathbf{l} =$  vertex of  $\tau$ , where  $\mathbf{l} \neq \mathbf{p}, \mathbf{l} \neq \mathbf{r}$ ;
         $l'_y = l_x \cdot k\sin + l_y \cdot k\cos$ ;
    end
     $\mathbf{s} = \mathbf{l}; \mathbf{l} = \mathbf{r}; \mathbf{r} = \mathbf{p}$ ;
else
    //  $\mathbf{r}$  is below  $\overrightarrow{\mathbf{p}\mathbf{q}}$ 
    repeat
         $\tau =$  neighbor of  $\tau$  trough  $\epsilon_{pr}$ ;
         $\mathbf{l} = \mathbf{r}$ ;
         $\mathbf{r} =$  vertex of  $\tau$ , where  $\mathbf{r} \neq \mathbf{p}, \mathbf{r} \neq \mathbf{l}$ ;
         $r'_y = r_x \cdot k\sin + r_y \cdot k\cos$ ;
    until  $r'_y > q'_y$ ;
     $\mathbf{s} = \mathbf{r}; \mathbf{r} = \mathbf{l}; \mathbf{l} = \mathbf{p}$ ;
end
// now  $\overrightarrow{\mathbf{p}\mathbf{q}}$  intersects  $\tau$ 
// walk step - following the line segment  $\overrightarrow{\mathbf{p}\mathbf{q}}$ 
 $s'_x = s_x \cdot k\cos - s_y \cdot k\sin$ ;
while  $s'_x < q'_x$  do
     $s'_y = s_x \cdot k\sin + s_y \cdot k\cos$ ;
    if  $s'_y < q'_y$  then  $\mathbf{r} = \mathbf{s}$ ; else  $\mathbf{l} = \mathbf{s}$ ;
     $\tau =$  neighbor of  $\tau$  trough  $\epsilon_{lr}$ ;
     $\mathbf{s} =$  vertex of  $\tau$  where  $\mathbf{s} \neq \mathbf{r}, \mathbf{s} \neq \mathbf{l}$ ;
     $s'_x = s_x \cdot k\cos - s_y \cdot k\sin$ ;
end
return remembering_stochastic_walk( $\mathbf{q}, \tau$ );
    
```

Algorithm 1: Hybrid Walk

contain the query point, but it is usually very close to the one that does. The final location is performed by the RSW algorithm (as can be seen in Section IV, it visits about 2 triangles in average). For its implementation, we used the pseudocode from [3].

IV. EXPERIMENTAL RESULTS

For the testing purposes, we implemented the proposed algorithm and the previous algorithms in Java with double precision floating point arithmetic. The algorithms were tested on Intel Q6600 2,40GHz. Based on the tests performed on different types of triangulations, we chose Delaunay triangulation as a sufficient representative. The tests were performed on triangulations of many different datasets, which were of three different types: randomly distributed points in the unit square, the real geodetic data from land registers and LIDAR data.

We selected the fastest of the existing algorithms and compared them with our proposed solution. The selected algorithms were: Remembering stochastic walk (RSW),

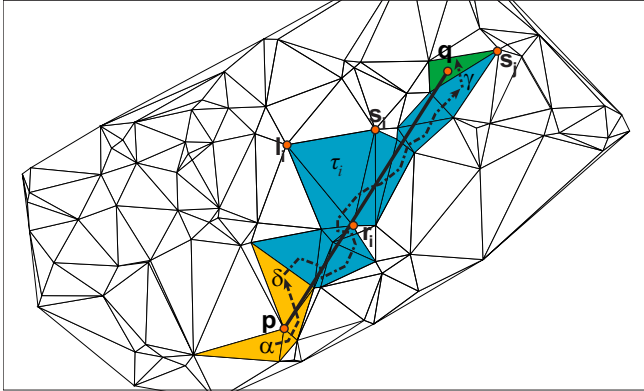


Figure 3. A hybrid walk example (a dashed line denotes the initialization step, a chain line denotes the walk and the final location by RSW algorithm is marked by a dotted line).

Normal Straight Walk (NSW), Improved Orthogonal Walk (IOW) and our Hybrid Walk (HW). Remembering walk is faster than its modification RSW, however, it may loop for other than Delaunay triangulations, thus to be objective, we did not include it in our test results.

In each test, 10^7 pairs of the initial triangle and the target point were generated randomly, and the average number of the tested properties were computed. Such properties were: the length of the walk ($\#\Delta$), the number of the tests ($\#\text{tests}$) and the time per one location ($t[\mu s]$). The properties $\#\text{tests}$ and $\#\Delta$ consist of two values for some algorithms. The former value concerns the walk, the latter concerns the final location performed by RSW.

Table I contains selected results of the tests for a triangular mesh enclosed in a rectangle preventing the orthogonal walk from crossing the border of the triangular mesh. Note that the number of triangles visited by our algorithm is about the same as in NSW, because both algorithms visit triangles intersected by the line connecting a point from the starting triangle with the query point, the only difference is in the particular selected point. However, tests done in each triangle by NSW are slower than by our algorithm, therefore the time per one location is higher. RSW algorithm is the slowest because of the 2D orientation tests and randomization done in each step. We included it in the tests as a representative of visibility walk group, but especially because it is used for the final location in all NSW, IOW and HW algorithms.

Table II compares our algorithm with IOW for randomly distributed points in a rectangle 2:1, rotated by $\pi/6$, which aims to resemble a more realistic situation, where some particular walks cross the border of the triangular mesh. It can be seen that our algorithm is the most suitable for such data that are not enclosed in a shape preventing the walk from crossing the border of the triangular mesh. Even a small percentage of walks leaving the triangular mesh slows down the whole location process in a way that our

algorithm is faster. With a growing percentage of such walks, our algorithm becomes significantly faster. Recall Figure 1, providing an explanation for this behavior. Each walk leaving a triangular mesh in OW leads into a longer final walk done by RSW algorithm, which is slower (see Table I).

Even for the cases, when the walk does not cross the triangular mesh border (Table I), our algorithm has comparable results to OW, particularly for a uniform distribution its speed is similar or better. Moreover, its implementation is simpler than the one of OW, because our algorithm does not have four different cases which need to be solved separately.

V. CONCLUSION

We presented a new walking algorithm, combining the basic idea of two walking strategies (straight and orthogonal walk). Experiments proved that our algorithm is faster than the fastest existing visibility and straight walk algorithms, and comparable with orthogonal walk algorithms. If there is even a small percentage of walks that cross the boundary of the triangular mesh, our algorithm becomes faster than the orthogonal walk. Furthermore, its implementation is simpler, because the problem does not split into cases which has to be solved separately, as it is for the orthogonal walk.

ACKNOWLEDGMENT

The authors would like to thank their colleague Martin Maňák for his feedback and inspiring discussions. This work has been supported by the Grant Agency of the Czech Republic under the research project 201/09/0097 and by the project SGS-2010-028.

REFERENCES

- [1] Star-shaped polyhedron point location with orthogonal walk algorithm. *Procedia Computer Science*, 1(1):219–228, 2010.
- [2] Jean-Daniel Boissonnat and Monique Teillaud. On the randomized construction of the Delaunay tree. *Theoretical Computer Science*, 112(2):339 – 354, 1993.
- [3] O. Devillers, S. Pion, and M. Teillaud. Walking in a triangulation. In *Proceedings of the 17th Annual Symposium on Computational Geometry*, pages 106–114, 2001.
- [4] L. Devroye, E. P. Mucke, and Binhai Zhu. A note on point location in Delaunay triangulations of random points, 1998.
- [5] P. J. Green and R. Sibson. Computing Dirichlet tessellations in the plane. *The Computer Journal*, 21:168–173, 1978.
- [6] I. Kolingerová. A small improvement in the walking algorithm for point location in a triangulation. In *Proceedings of the 22nd European Workshop on Computational Geometry*, pages 221–224, 2006.
- [7] C. L. Lawson. *Mathematical Software III; Software for C1 Surface Interpolation*, pages 161–194. Academic Press, New York, 1977.

Algorithm	# Δ	#test	t[μ s]	# Δ	#test	t[μ s]	# Δ	#test	t[μ s]
	ϕ per located point			ϕ per located point			ϕ per located point		
Real geodetic data from land registers									
	4897 vertices (9774 Δ)			15824 vertices (31642 Δ)			70437 vertices (140868 Δ)		
RSW	92.09	122.18	6.24	158.1	208.45	15.19	321.27	417.88	64.29
NSW	87.46 + 2.29	174.02 + 4.28	3.92	149.16 + 2.43	297.31 + 4.42	10.75	293.4 + 2.93	585.8 + 4.9	49.32
IOW	94.16 + 4.33	188.32 + 7.03	3.30	176.52 + 2.69	353.04 + 5.19	8.40	319.04 + 3.48	638.08 + 6.35	28.72
HW	87.46 + 2.29	174.02 + 4.28	3.52	149.21 + 2.44	297.83 + 4.81	9.31	293.6 + 2.94	594.6 + 5.57	39.69
LIDAR									
	34932 vertices (69858 Δ)			313348 vertices (626690 Δ)			3722068 vertices (7444130 Δ)		
RSW	205.3	266.18	25.59	647.23	830.46	137.78	2563.63	3277.01	880.98
NSW	189.99 + 1.76	378.98 + 3.76	15.98	598.62 + 1.92	1196.24 + 3.92	105.57	2385.51 + 2.03	4770.03 + 4.02	659.48
IOW	246.61 + 1.75	493.22 + 3.82	13.74	801.88 + 1.75	1603.76 + 3.83	82.29	2886.95 + 2.0	5773.9 + 4.21	448.46
HW	187.61 + 1.76	374.7 + 3.83	14.09	596.38 + 1.92	1192.16 + 4.07	89.42	2385.69 + 2.03	4770.78 + 4.23	530.01
Randomly distributed points in the unit square									
	10^4 vertices (19994 Δ)			10^5 vertices (199994 Δ)			10^6 vertices (1999994 Δ)		
RSW	115.19	150.58	7.44	362.82	468.62	71.31	1144.86	1472.05	287.63
NSW	107.23 + 1.76	213.45 + 3.76	4.87	339.75 + 1.76	678.51 + 3.76	53.74	1073.59 + 1.76	2146.17 + 3.76	220.92
IOW	135.35 + 1.76	270.71 + 3.84	4.61	432.08 + 1.76	864.15 + 3.84	49.11	1371.06 + 1.76	2742.12 + 3.84	188.51
HW	108.5 + 1.76	216.4 + 3.84	4.29	342.9 + 1.76	685.23 + 3.84	49.09	1065.25 + 1.76	2129.88 + 3.85	191.13

Table I
COMPARISON OF THE SELECTED ALGORITHMS WITH RANDOMLY CHOSEN α

Algorithm	# Δ	#test	t[μ s]	# Δ	#test	t[μ s]	# Δ	#test	t[μ s]
	ϕ per located point			ϕ per located point			ϕ per located point		
Randomly distributed points in rotated rectangle 2x1									
	10^4 vertices (19994 Δ)			10^5 vertices (199994 Δ)			10^6 vertices (1999994 Δ)		
IOW	142.17 + 17.45	284.34 + 24.49	5.93	451.16 + 52.23	902.33 + 69.96	61.35	1419.15 + 150.07	2838.3 + 197.61	239.87
HW	123.07 + 1.77	245.46 + 3.85	4.88	385.05 + 1.76	769.42 + 3.84	55.13	1218.09 + 1.76	2435.55 + 3.84	217.44

Table II
COMPARISON OF THE SELECTED ALGORITHMS ON A RECTANGLE ROTATED BY $\pi/6$ WITH RANDOMLY CHOSEN α

[8] Kurt Mehlhorn and Stefan Näher. Leda: A platform for combinatorial and geometric computing. *Communications of the ACM*, 38(1):96–102, 1995.

[9] E. P. Mücke, I. Saias, and B. Zhu. Fast randomized point location without preprocessing in two and three-dimensional Delaunay triangulations. In *Proceedings of the 12th Annual Symposium on Computational Geometry*, volume 26, pages 274–283, 1996.

[10] K. Mulmuley. Randomized multidimensional search trees: Dynamic sampling. In *Proceedings of the 7th Annual Symposium on Computational Geometry*, pages 121–131, 1991.

[11] S. W. Sloan. A fast algorithm for constructing Delaunay triangulations in the plane. *Advances in Engineering Software*, 9(1):34–55, 1987.

[12] Roman Soukal and I. Kolingerová. Straight walk algorithm modification for point location in a triangulation. In *EuroCG'09: Proceedings of the 25th European Workshop on Computational Geometry*, pages 219–222, Brussels, Belgium, 2009.

[13] P. Su and R. L. S. Drysdale. A comparison of sequential Delaunay triangulation algorithms. In *Proceedings of the 11th Annual Symposium on Computational Geometry*, pages 61–70, 1995.

[14] B. Žalik and I. Kolingerová. An incremental construction algorithm for Delaunay triangulation using the nearest-point paradigm. *International Journal of Geographical Information Science*, 17(2):119–138, 2003.

[15] Frank Weller. On the total correctness of Lawson’s oriented walk. In *Proceedings of the 10th International Canadian Conference on Computational Geometry*, pages 10–12, 1998.

[16] M. Zadavec and B. Žalik. An almost distribution independent incremental Delaunay triangulation algorithm. *The Visual Computer*, 21(6):384–396, 2005.

[17] Sheng Zhou and Christopher B. Jones. HCPO: an efficient insertion order for incremental Delaunay triangulation. *Information Processing Letters*, 93(1):37–42, 2005.

[18] Binhai Zhu. On lawsons oriented walk in random delaunay triangulations. In Andrzej Lingas and Bengt Nilsson, editors, *Fundamentals of Computation Theory*, volume 2751 of *Lecture Notes in Computer Science*, pages 222–233. Springer Berlin / Heidelberg, 2003.

Virtual Environment in Civil Engineering

Construction and Maintenance of Buildings

Alcnia Z. Sampaio, Ana Rita Gomes, Joana Prata

Dep. Civil Engineering and Architecture

Technical University of Lisbon

Lisbon, Portugal

e-mail: zita@civil.ist.utl.pt, ritagomes05@hotmail.com, jo.p@sapo.pt

Abstract— This paper describes two prototype applications based on Virtual Reality (VR) technology for use in construction and maintenance planning of buildings. The first, applied to construction, is an interactive virtual model designed to present plans three-dimensionally (3D), connecting them to construction planning schedules, resulting in a valuable asset to the monitoring of the development of construction activity. The 4D application considers the time factor showing the 3D geometry of the different steps of the construction activity, according to the plan established for the construction. The 4D model offers a detailed analysis of the construction project. It allows the visualization of different stages of the construction and the interaction between all stakeholders during the actual construction activity. A second VR model was created in order to help in the maintenance of exterior closures of walls in a building. It allows the visual and interactive transmission of information related to the physical behavior of the elements. To this end, the basic knowledge of material most often used in faades, anomaly surveillance, techniques of rehabilitation, and inspection planning were studied. This information was included in a database that supports the periodic inspection needed in a program of preventive maintenance. This work brings an innovative contribution to the field of construction and maintenance supported by emergent technology.

Keywords- *Construction; Maintenance; Virtual reality; Interactive model.*

I. INTRODUCTION

The main aim of a research project, now in progress at the Department of Civil Engineering of the Technical University of Lisbon, is to develop virtual models as tools to support decision-making in the planning of construction management and maintenance, PTDC/ ECM/67748/ 2006, "Virtual Reality technology applied as a support tool to the planning of construction maintenance". A first prototype for the lighting system had already been completed [1]. A second prototype concerning construction planning is now complete [2] and the VR model concerning maintenance of the closure of exterior walls is also finished [3]. This paper describes these two later models created as part of the overall research project.

These interactive models integrate Virtual Reality (VR) technology and applications implemented in Visual Basic (VB) language. The models allow interaction with the 3D geometric model of a building, visualizing components for each construction. They are linked to databases of the

corresponding technical information concerning construction planning and the maintenance of the materials used as exterior closures. The principal objective of the interactive VR prototypes is to support decision-making in the area of planning.

Information technology, namely 4D modeling (3D+time) and VR techniques is currently in use both in the construction activity and in education [4]. At the Department of Civil Engineering, some didactic models have already been generated. The research project presented in this paper follows on from that previous educational work: two 3D geometric models which support activity in the rehabilitation of buildings [5]; and three VR models developed to support classes in Civil Engineering (wall, bridge and roof construction) in Technical Drawing, Construction and Bridge disciplines [6]. The didactic VR models are in common use in both face-to-face classes and on an e-learning platform.

Virtual Reality technology can support the management of data that is normally generated and transformed or replaced throughout the lifecycle of a building. This technology constitutes an important support in the management of buildings allowing interaction and data visualization. At present, the management of building planning can be presented in 3D form and various materials can be assigned to the fixtures and furnishing enabling the user to be placed in the virtual building and view it from inside as well as outside. This study contemplates the incorporation of the 4th dimension, that is, time, into the concept of visualization. The focus of the work is on travelling through time, or the ability to view a product or its components at different points in time throughout their life. In maintenance, the time variable is related to the progressive deterioration of the materials throughout the building's lifecycle. It is implicit that the incorporation of the time dimension into 3D visualization will enable the designer/user to make more objective decisions about the choice of the constituent components of the building.

In construction management, over the years, technical drawings have played a crucial role in communication between the numerous partners in a project. Generally, drawings represent formal solutions, and often incompatibility mistakes are only detected at advanced stages, on site, accruing additional costs. In this field 4D models promote the interaction between the geometric model and construction activity planning, allowing immediate perception of the evolution of the work. In planning, in correct evaluation and the meeting of needs as they arise, 4D

models constitute a positive contribution to decision-making when establishing planning strategies [7].

Section II describes the maintenance model, highlighting the constitution of the database supporting the model, and the organization of a user-friendly interface designed to be used by an inspection worker. During the construction of this model, the basic knowledge of the topics involved, such as aspects related to the materials, the techniques of rehabilitation and conservation and the planning of maintenance is outlined and discussed. In addition, methods of interconnecting this knowledge with the virtual model are explored. This prototype was trialled in an actual project.

These aspects of the construction activity are in constant evolution, so require the study of preventive maintenance, through, for example, the planning of periodic local inspections and corrective maintenance with repair activity analysis. For this reason, the model facilitates the visual and interactive access to results, supporting the drawing-up of inspection reports.

The construction model, presented in Section III, brings an innovative aspect to 4D modelling as usually applied to construction planning, through the incorporation of pictures into the interface of the VR model, an important support element in the comparison between what is planned and what is in progress *in situ* after each construction task.

II. THE WALL MAINTENANCE MODEL

Façade coatings play an important role in the durability of buildings, since they constitute the exterior layer that ensures the protection of the wall against the aggressive actions of a physical, chemical or biological nature. Naturally, they should also give the façade the required decorative effect. Since this building component is exposed to adverse atmospheric conditions it frequently shows an evident degree of deterioration, requiring maintenance work. In order to arrive at the best solution for eventual maintenance and repair work, a survey of defects and deterioration must be conducted.

In order to better understand the operation of façade coating, bibliographic research of materials usually applied to this type of material was carried out and a table of characteristics of these was drawn up. Subsequently, a survey was made of anomalies, probable causes, solutions and methods of repair for each of the coatings studied. The visualization of the maintenance data of a building and the impact of time on the performance of these exterior closure materials require an understanding of their characteristics [8] (Figure 1):

- Types of material: painted surfaces, natural stone panels and ceramic wall tiles;
- Application processes: stones (panel, support devices, adherent products, etc.); ceramic tiles (fixed mechanism, procedures, ...); painted surfaces (types of paint products, prime and paint scheme surface, exterior emulsion paints, application processes);
- Anomalies: dust and dirt, lasting lotus leaf effect, covering power, insufficient resistance to air permeability or weather-proof isolation, damaged

stones or ceramic tiles, alkali and smear effect, efflorescence, fractures and fissures and so on;

- Repair work: surface cleaning, wire truss reinforcing, cleaning and pointing of stonework joints, removing and replacement of ceramic wall tiles, removing damaged paint and paint surface, preparing and refinishing stone panels, etc..

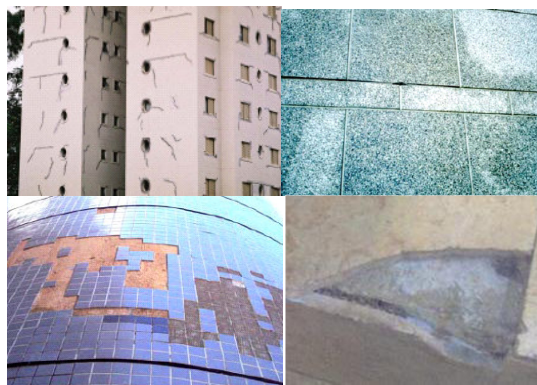


Figure 1: Different types of materials applied as façade coatings.

Depending on the role that the façade coatings play on the wall as a whole they can be classified as finishing, sealing or thermal insulation. The most frequent materials used as coating finishes are painting, tiling and, as sealing coating of the natural stone:



- Paint coating contributes to the aesthetic quality of the building and its environment and also protects the surface of the exterior wall against corrosion, deterioration and penetration of aggressive agents [9];
- The ceramic coating consists essentially of tiling panels, cement and adhesive and the joints between the slabs. The application of ceramic tiling to building façades has considerable advantages particularly as some degree of waterproofing is afforded by the glazed surface along with a great resistance to acids, alkalis and vapour [10];
- The use of natural stone in the coating of façade surfaces is a good solution both technically and aesthetically. The principal characteristics of the stones are: reduced water absorption, sufficient mechanical resistance to bending and impact, abrasion and shearing parallel to the face of the slabs.

A. The database

The most frequent anomalies that occur in the coated façades were analysed in order to create a database linked to the virtual model that could support the planning of inspections and maintenance strategies in buildings. This database contains the identification of anomalies that can be found in each type of material used in façades and the corresponding probable cause. For each type of anomaly the most adequate repair solutions were also selected and included in the database. The following example, concerning

deficiencies in tiles, illustrates the methodology implemented in this virtual application (Table 1).

TABLE I. EXAMPLE OF ANOMALIES AND THE ASSOCIATED REPAIR SOLUTION

Anomaly	<i>Detachment</i>	<i>Cracking / Fracturing</i>
		
Specification of the anomaly	Fall in areas with deterioration of support	Failure of the support (wide cracks with well defined orientation)
Repair solution	Replacement of the coat (with use of a repair stand as necessary)	Replacement of the coat (with repair of cracks in the support)
Repair methodology	<ol style="list-style-type: none"> 1. Removal of the tiles by cutting grinder with the aid of a hammer and chisel; 2. Timely repair of the support in areas where the detachment includes material constituent with it; 3. Digitizing layer of settlement; 4. Re-settlement of layer and tiles. 	<ol style="list-style-type: none"> 1. Removal of the tiles by cutting grinder; 2. Removal of material adjustment in the environment and along the joint; 3. Repair of cracks, clogging with adhesive material (mastic); 4. Settlement layer made with cement in two layers interspersed with glass fibre; 5. Re-settlement of layer and tiles

B. The interface

The implementation of the prototype system makes use of graphical software programming, *Microsoft Visual Basic 6.0*, software to establish a suitable database, *Microsoft office access*, graphical drawing system, *AutoCAD Autodesk* and VR technology based software, *EON Studio* [11].

Many potential users are not computer experts. Human perceptual and cognitive capabilities, therefore, were taken into account when designing this visualization tool with the result that the model is easy to use and does not require sophisticated computer skills. It uses an interactive 3D visualization system based on the selection of elements directly within the virtual 3D world. Furthermore, associated with each component, there are integrated databases, allowing the consultation of the required data at any point in time.

The interface is composed of a display window allowing users to interact with the virtual model, and a set of buttons for inputting data and displaying results (Figure 2). For each new building to be monitored, the characteristics of the environment (exposure to rain and sea) and the identification of each element of the façades must be defined. The data associated to each element are the building orientation, the type of exterior wall (double or single), and the area and type of coating.

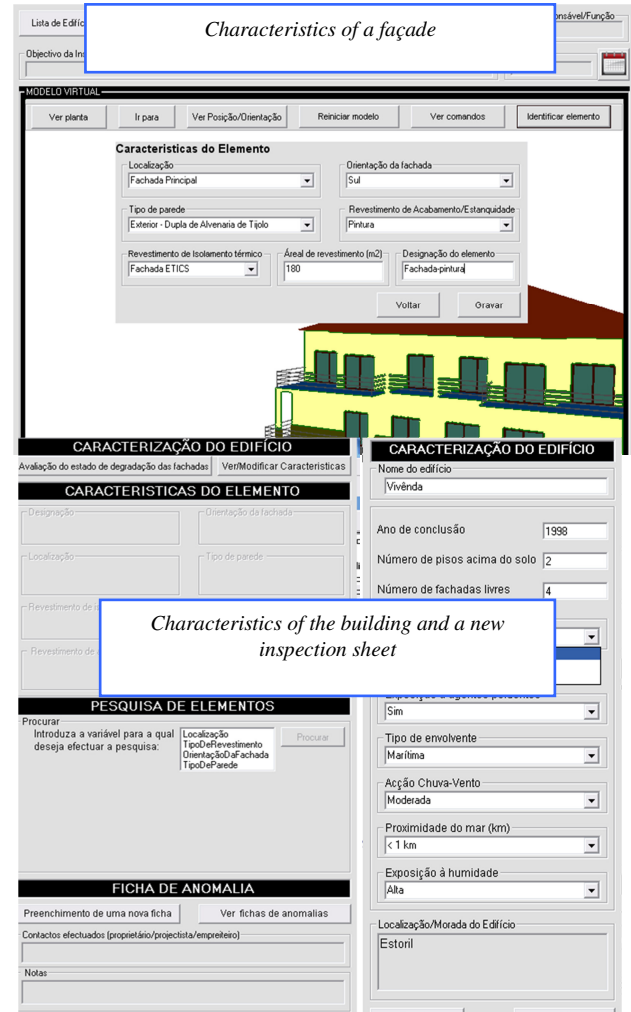


Figure 2: The main interface of the interactive application.

Once each monitored element has been characterized, various inspection reports can be defined and recorded and thereafter consulted when needed. An inspection sheet is accessed from the main interface (Figure 3).

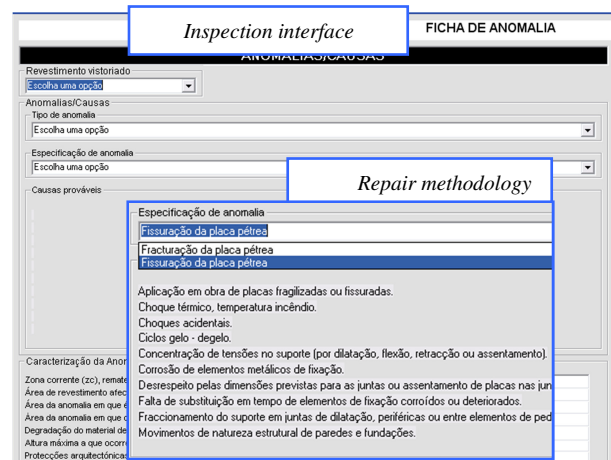


Figure 3: Inspection sheet interface

The inspection sheet includes the type of covering (natural stone, Figure 4), the anomalies (*Cracking /Fracturing*) and a list of possible causes to be selected and associated to the element. Several photos can be added.

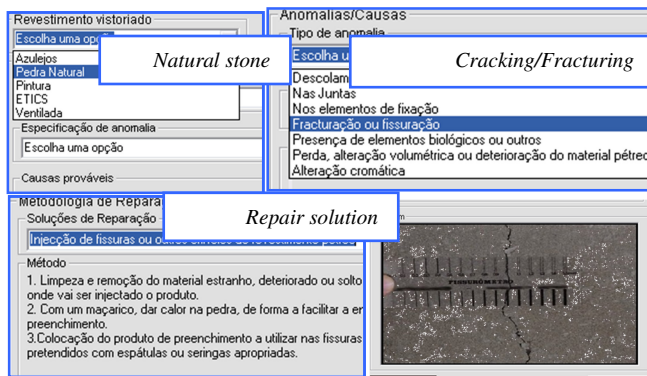


Figure 4: Selecting data in the inspection interface

The repair methodology associated to the selected type of anomaly is also presented. These data are then linked to the element. A report history of inspections is permanently associated to the building so that during any later interaction with the model, the inspection report that was defined using the prototype can be accessed.

To sum up, by using the drop-down menus allowed by the interface, the user can associate the characteristics of the observed anomaly to: a façade element; the type of anomaly, the specification, details and the probable cause of the anomaly, an adequate repair solution and pictures taken in the building. After completing all fields relating to an anomaly, the user can present the report as a pdf file.

C. The case study

First, the 3D geometric model of a building was created (Figure 5). In this case, the building consists of a ground-floor, a 1st floor and an attic with dwelling space. The coating elements of the walls were then modeled as independent geometric objects. In this way, each element can then support characterization data of the applied material and different kinds of information related to maintenance.

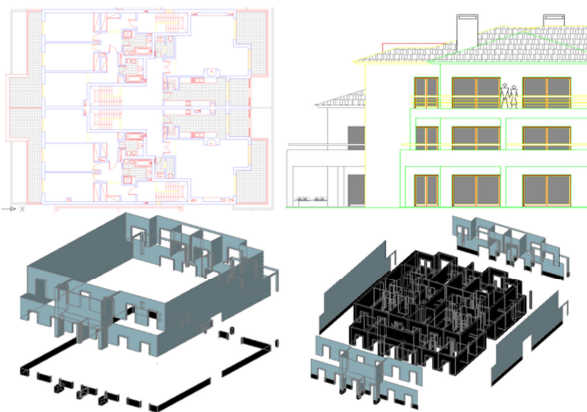


Figure 5: Steps in the geometric modeling process.

All coatings studied were considered in this case-study. Thus the main façade was assumed to be tiled and the remaining façades painted while hall façades are of natural stone. Figure 6 shows how to identify a façade in the virtual model of the building. Figure 7 shows the inspection report of an anomaly.



Figure 6: Identification of a façade element.



Figure 7: An inspection sheet report.

III. THE CONSTRUCTION PLANNING MODEL

Construction management can be defined as the planning, co-ordination and control of a project from conception to completion (including commissioning) on behalf of a client [12]. This requires the identification of the client's objectives in terms of usage, function, quality, time and cost, and the establishment of relationships between the people involved, integrating, monitoring and controlling the contributors to the project and their output, and evaluating and selecting alternative solutions in pursuit of the client's satisfaction with the outcome of the project. It is essential, therefore, that the project designer has the depth of knowledge to be able to correctly identify the different stages of the construction planning, as well as to take into consideration the logistics and resources involved in the project. The construction planning used in the implemented prototype is realistic and considers the graphic and written documentation, measurements and quantities map, specifications and regulations relevant to the project [13].

A prototype based on VR technology with application to these demands of construction planning, was created. This interactive virtual model presents the project in 3D, integrated with the construction planning schedule, resulting in a valuable asset in monitoring the development of the construction activity, compared to the construction planning already drawn up. The 4D application allows the time factor to be considered in conjunction with the 3D geometry of the different steps of the construction activity, according to the schedule established for the construction, thus offering a detailed analysis of the construction project. Additionally, VR technology allows the visualization of different stages of the construction and interaction with the real-time construction activity. This application clearly shows the constructive process, avoiding inaccuracies and building errors, thereby facilitating better communication between partners in the construction process.

This application was developed in three stages: planning, modeling, and the integration of the first two stages.

- Planning takes into consideration the final purpose of the presentation, and the definition of tasks; the details, therefore have to be in line with this idea. Using *Microsoft Project 2007*, the tasks are introduced and the relations between them defined;
- Geometric modeling needs to relate correctly to the tasks as defined at the planning stage. Using *AutoCAD 2010* as a modeling tool, the layers make the distinction between the different tasks and elements are created in enough detail to support correct comprehension. The application also presents a real-time illustration of the evolution of the construction through photographs of the site, taken at specific points in time;
- The third stage, integration, makes use of two programs: *EON Studio 5.0* and *Microsoft Visual C# 2008 Express Edition*, where the first takes the 3D model created with *AutoCAD* and introduces it in the application developed using the second.

A. The interface

The application, developed in C#, integrates all the components described with the interface as shown in Figure 8.

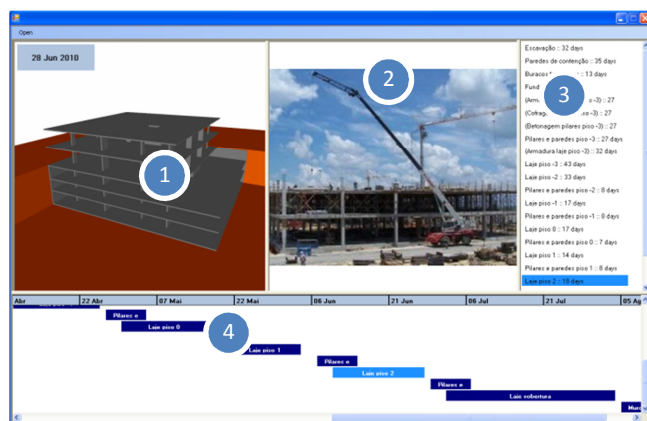


Figure 8: Application interface.

The application his organized as outlined below: Virtual model (1); Pictures of construction site (2); Planning task list (3); Gantt map (4). The interaction with the application is made through 3 and 4. Both the task list items and Gantt map bars are buttons which, when pressed, send the information to the *EON* for the task selected, and in return *EON* presents the model in the current state, that is, it shows and hides specific elements depending on the specific stage of the construction.

EON can interact with the model in a number of different ways. In this prototype only the state of the elements and camera position is changed. The state of an element is presented by its Hidden property, whether it is selected or not, whilst the camera position is determined by translation and rotation coordinates. *EON Studio* also offers the possibility of changing the material associated with each element, creating a more realistic model.

Any new objects can be introduced into the application, just by modeling the new elements considering their positions relative to the ones already in the simulation and programming the associated action in *EON Studio*.

Likewise, the application accepts any kind of construction project, as long as its implementation imperatives are met. Additionally, with the appropriate models, it can also be used in construction site management.

The weakness of this prototype lies in the time needed to carry out the preparation for the actual interaction with the application. Modeling a building may not be very extensive. The programming of the actions in *EON Studio*, however, can be time-consuming.

B. The case study

As a method of testing the application, a construction project was undertaken, more particularly, the structure of a building, using both its graphic documentation, that is, the architectural and structural blueprints, and the project description and construction planning (Figure 9).

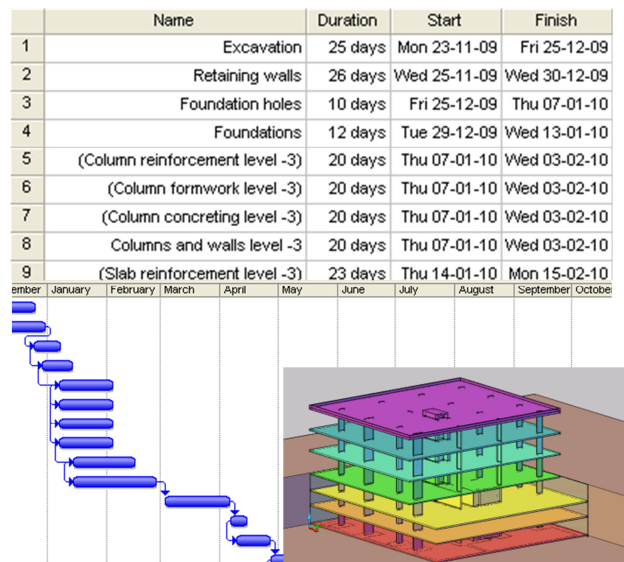


Figure 9: Construction planning (list and Gantt map) and the 3D model of the building structure.

The whole project was simplified to serve this paper's academic purposes: the list of tasks was defined based on the more characteristic stages of a construction process, and a few tasks focused on the construction details of certain elements. As a result, *AutoCAD* layers were created for each task defined and the 3D model constructed. When finished, the 3D model was exported to *EON Studio*, where a diagram of events was created, after which the application was ready to be used.

As explained above, the task list and the virtual model are connected: when selecting a task, the relevant construction stage is presented (Figure 10). The first scenario is the landscape and then the foundation work is shown (Figure 11).

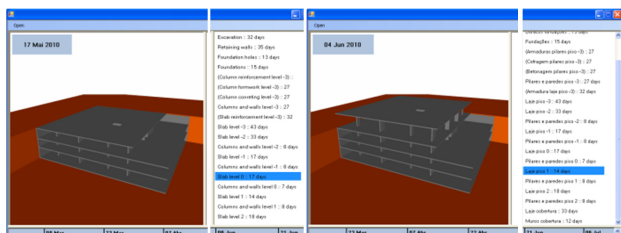


Figure 10: Application's virtual model and task list.

In Figure 11, some construction details have been modeled. Progress across three different stages, of one of the columns, is shown in Figure 12. There being no picture associated, the camera symbol becomes visible instead. A detail of the reinforcement and concrete of a slab is shown in Figure 13.

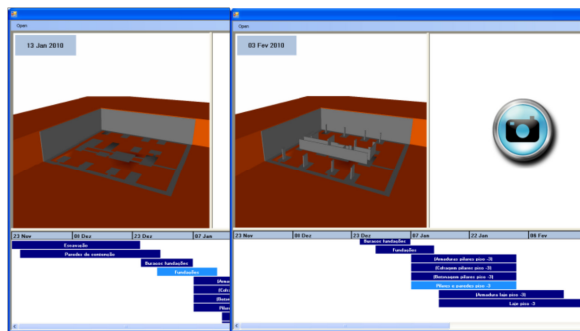


Figure 11: Visualization of the foundation work.

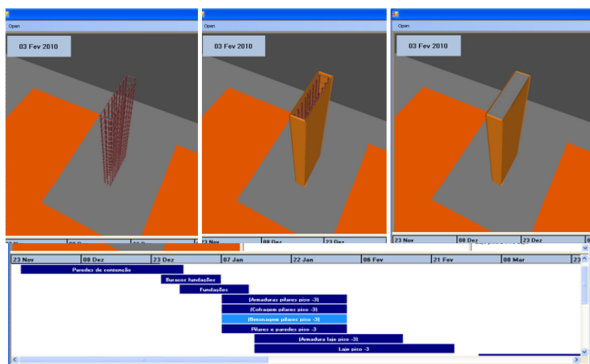


Figure 12: Column construction: reinforcement, formwork and concreting.

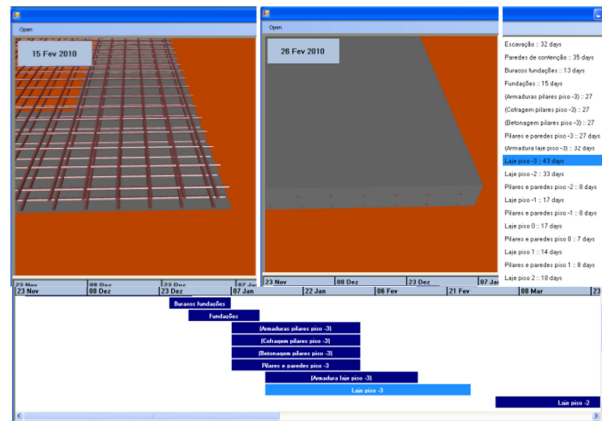


Figure 13: Construction of a slab.

When constructing a building, the planning sometimes needs to be changed due to unexpected occurrences. Implementing these changes in the prototype is actually very simple, as the user has only attribute new start and finish dates to the task in *MS Project* and load the new file into the application.

When a task is selected in the construction planning chart the static position of the model is presented. A first view is always linked to a task. This was established to provide easier interaction with the 3D model, and to focus the attention of the user on the important sections of each task, guiding them through the proper course of development of the construction.

Next, the user can manipulate the virtual model, in order to choose the identical perspective as that shown in the photo. So, with the visualization of what is planned and what has been done in the real building, the construction work can be better compared and analyzed (Figure 14).

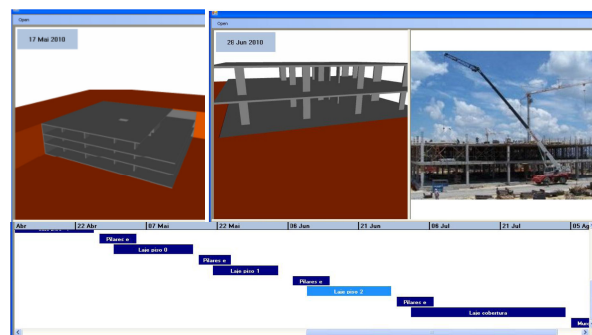


Figure 14: Rotation applied to the virtual model.

In addition by manipulating the model the user can walk through the virtual building observing any construction detail he wants to compare.

All steps have been modelled and linked to the planning chart. Figure 15 shows the details of the construction work. The date for each visualized task is shown in the upper left corner of the virtual model window.

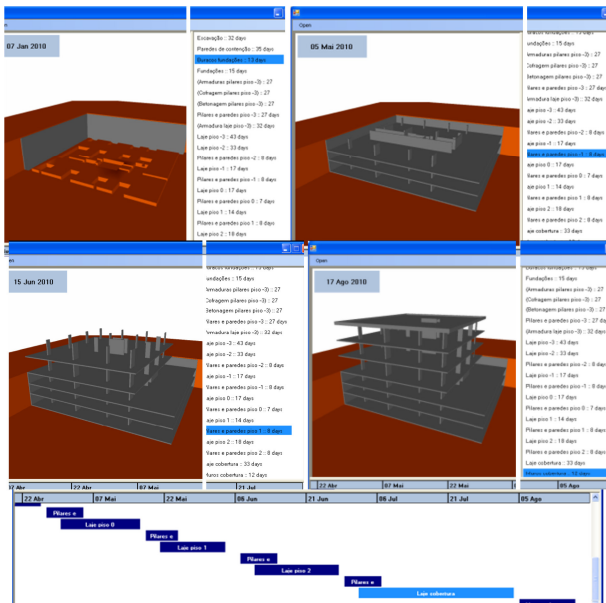


Figure 15: Sequence of the construction process.

IV. CONCLUSIONS

Virtual Reality technology with its capability of interaction and connectivity between elements was employed in the developed prototypes within a research project, offering several benefits both in presenting and developing projects and in supporting decision-making in the maintenance domain.

A VR model to support the maintenance of walls enables the visual and interactive transmission of information related to the physical behaviour of the elements. The model shows the characteristics of each element of the building in the model and the information related to inspection, anomalies and repair works. As the 3D model is linked to a database in an interactive environment and has a user-friendly interface with easily manipulation of the data, it engenders a collaborative system. With this application, the user may fully interact with the program referring to the virtual model at any stage of the maintenance process and can analyse the best solution for repair work. It can also support the planning of maintenance strategies. The developed software is easy to handle and transport for on-site inspections and comprises information of the causes, solutions and methods for repairing anomalies.

Technical drawings and explanatory texts often have little detail and are frequently insufficient in fully comprehending the object. Using VR models means that mistakes can more easily be caught before construction starts, which translates into time and cost reduction. The construction planning model can be used with any kind of construction project and, being a flexible application, accepts new data when necessary, allowing for a comparison between the planned and the constructed. The prototype can also be expanded to include other aspects of construction management, such as resource administration, or to have real-time access to the construction, through the use of

cameras installed on site. The use of new mobile technologies could move the application to the construction site, clarifying any doubts about location or position of each component.

Both models support construction activity. The VR construction model allows the presentation of each step comparing what is planned with the real situation observed, the pictures taken *in situ*. The model, therefore, helps the designer and owner to redefine the early plan introducing changes to the work in progress. Thus, economic benefits of updating the planning schedule are achieved along with better, error-free construction with no unnecessary delays.

The maintenance model supports the global analysis of the need for repair tasks in a building, helping the designer to define an adequate plan of rehabilitation work. The plan must incorporate the repair of all anomalies detected during an inspection visit, which are reported, with the help of the VR model, it, too, bringing economic benefits.

ACKNOWLEDGMENT

The authors gratefully acknowledge the financial support of the Foundation for Science and Technology, a Governmental Organization for the research project PTDC/ECM/67748/ 2006, "Virtual Reality technology applied as a support tool to the planning of construction maintenance", now in progress.

REFERENCES

- [1] Sampaio, A.Z., Ferreira, M.M., and Rosário, D.P., 2009. Interactive virtual application on building maintenance: The lighting component, In Proc. IRF2009, 3rd International Conference on Integrity, Reliability and Failure: Challenges and Opportunities, Symposium Visualization and human-Computer Interaction, Porto, Portugal, July 20-24, abstract pp. 221-222, paper 11 pgs.
- [2] Santos, J.P. 2010 'Construction Planning using 4D Virtual Models', Integrated Master Degree Thesis in Construction, Technical University of Lisbon, Portugal.
- [3] Gomes, A.R., 2010. Virtual Reality technology applied to the maintenance of façades, Integrated Master Degree Thesis in Construction, Technical University of Lisbon, Portugal.
- [4] Mohammed, E.H., 2007. n-D Virtual Environment in Construction Education, the 2nd International Conference on Virtual Learning, ICVL 2007, pp. 1-6.
- [5] Sampaio, A.Z., Henriques, P.G., and Ferreira, P.S., 2006. Virtual Reality Models Used in Civil Engineering, IMSA'06 Proceedings of the 24th IASTED international conference on Internet and multimedia systems and applications Education, ACTA Press Anaheim, CA, USA, USA. <http://portal.acm.org/citation.cfm?id=1169188> <retrieved: March, 2011>
- [6] Sampaio, A.Z., Ferreira, M.M. Rosário, D.P., and Martins, O.P., 2010. 3D and VR models in Civil Engineering education: Construction, rehabilitation and maintenance, Automation in Construction 19 (2010) 819–828.
- [7] Webb, R.M. and Haupt, T.C. 2003. The Potential of 4D CAD as a Tool for Construction Management, 7th Int. Conf. on Construction Application of Virtual Reality, USA.
- [8] Gomes, A.M. and Pinto, A.P., 2009. Didactic text of construction materials, Technical University of Lisbon, IST, Lisbon, Portugal.
- [9] Ferreira, L., Coroado, J., Freitas, V., and Maguregui, I., 2009. Causes of the fall of tiles applied to exteriors of buildings.

- Patterned tiling in buildings from 1850-1920. In Conf.Patorreb, 3rd Meeting on Pathology and rehabilitation of Buildings, FEUP, Porto, March 18 -20
- [10] Veiga, M. and Malanho, S., 2009. Natural stone coating: methodology of diagnosis and repair of anomalies. In Conf.Patorreb 3rd Meeting on Pathology and Rehabilitation of Buildings, FEUP, Oporto, Portugal, March 18-20
- [11] EON Studio, <http://www.eonreality.com/> <retrieved: Septembre, 2010>
- [12] Walker, A. 2002. Project Management in Construction, Fourth edition, Oxford, Blachweel Publishing.
- [13] Casimiro, J., 2006. _Integrated Planning of Deadlines and Costs in PME's Small and Medium Bubsinneses, Civil Engineering, Final Report, Technical University of Lisbon, Portugal.

An Electrical Circuits e-Tutor based on Symbolic and Qualitative Analysis

Jason Debono

Institute for Electronics
Malta College for Science and Technology
Corradino, Malta
jason.debono@mcast.edu.mt

Adrian Muscat

Dept of Communications and Computer Eng.
University of Malta
Msida, Malta
adrian.muscat@um.edu.mt

Abstract—Numerical Time Domain electrical circuit simulators are the de facto standard in industry and education. Nevertheless circuits simulators based on symbolic or qualitative techniques have been equally studied. These latter techniques aim at simulating the mental models that an experienced engineer taps when manually designing and analysing a circuit. This paper describes the development of two software tools based on symbolic and qualitative analysis and their use as an electrical circuits eTutor is studied and discussed. As a comparison the paper also investigates the limitations and drawbacks of time domain electrical circuit simulators when used as a pedagogical tool. The field tests are carried out with the engagement of polytechnic teachers and students at the higher national diploma level.

Keywords—Electrical; Circuits; eTutor; Symbolic; Qualitative, Analysis.

I. INTRODUCTION

Most college, polytechnic and university electrical circuit theory courses include theoretical as well as practical sessions. The practical sessions are important for two reasons; (a) students learn how to link theoretical models to the real-life circuits, and (b) students learn how to carry out the appropriate measurements using the right instrument. These practical skills are indispensable for professional engineers during the installation, testing and maintenance, of electrical and electronics systems. However instrumentation is generally expensive and its use is restricted to labs. In this respect circuit simulators, such as SPICE, augmented with a graphical schematic capture front and back ends are very useful. With such elearning tools students connect virtual components together using virtual wires, choose and add virtual instruments to the circuit, and finally, carry out a computer analyses or. The software outputs the variables chosen or measurements as displayed on the virtual instruments. Such measurements include numerical values, like for example electrical current on a virtual meter, and voltage waveforms on a virtual oscilloscope. This type of eLearning software, widely distributed among colleges and polytechnics helps students in the acquisition of practical skills including the selection of instrumentation. It also speeds up the process and reduces the cost since there is no need for building the circuit in real life. However it does not help the student in understanding

how the circuit works or how to design the circuit. On the contrary, it encourages the student not to carry out a manual or mental analysis.

Apart from practical skills, electrical engineering students learn how to analyse and design electrical circuits. Traditionally students have been taught how to analyse electrical circuits using pen and paper through the application of the relevant theories, including Ohm's Law, Kirchhoff's Current Law and Kirchhoff's Voltage Law. As explained above SPICE simulators were not specifically designed to help students learn how circuits work, consequently SPICE simulators have some serious limitations when used as a pedagogical tool.

The electrical theories taught to students are an essential part of the mental models that the students must develop. Using these theories students can write down symbolic (algebraic) equations that describe how the circuit being analysed behaves. In contrast numerical simulators calculate values iteratively, and this approach limits the understanding and insight that the simulator can impart to its user about how the circuit being analysed functions. In the last few years symbolic simulators have been developed that build the symbolic equations that describe the circuit being analysed and display these equations explicitly. Examples of recently developed symbolic simulators are SAPWIN [1] and SNAP [2].

Additionally, accomplished engineers apply mental models in what-if analysis to understand how a change in a parameter at a point of the circuit affects the other parameters of the circuit. A change in a parameter, like for example the input voltage, is thought of as first influencing the parameters of its neighbouring components and nodes. In turn these varying parameters affect their own neighbours and hence the changes propagate throughout the circuit. This method of analysing a circuit was successfully implemented in a software program in 1977 by Sussman and Stallman, who termed this technique as the 'Propagation of Constraints' [3]. The algorithm implemented by Sussman and Stallman calculated the numerical values of voltages and currents. In fact the mental models used by engineers are usually more simplistic than this because the engineers only consider the direction of change, that is, an increase, a decrease or no change at all in the parameter's value. In other words the quality of the change is considered and not

the quantity of the change. This kind of reasoning has been studied and formally applied to electric circuits and other physical systems like thermodynamic systems in the field of study entitled “Qualitative Physics”. In 1984 Johan De Kleer implemented the Qualitative Analysis of electrical circuits in a program he called EQUAL [4]. This program is able to explain how a circuit works using qualitative arguments and even categorize the circuit as being a power-supply, logic-gate, amplifier or multivibrator.

In this paper two programs that target smaller scopes are discussed. The first program accepts input circuits that are made up of an arbitrary number of resistors and only one battery. This program processes serial and parallel connections of resistors. The second program accepts input circuits that are made up of an arbitrary number of resistors, voltage sources and current sources. The software tool then tutors the user on how to select valid spanning trees and the corresponding fundamental cutsets for the input circuit. The symbolic Kirchhoff’s Current Law (KCL) equation for each fundamental cutset is then generated by the program, which the user or student can compare to his/her workings. Both tools analyse the topology of the input circuit to accomplish their respective type of analysis. The first program is targeted to MQF level 4 students while the other program is to be used by MQF level 5 students.

The rest of the paper is organised as follows: Section II reviews circuit analysis techniques and section III reviews numerical models in education. The tools developed are described in section IV and the section V discusses results and conclusions.

II. CIRCUIT ANALYSIS PARADIGMS

In this section five types of circuit analysis paradigms are discussed, the *nodal and loop analysis*, *graph theory* in electrical circuits, *qualitative analysis*, *symbolic analysis* and the *propagation of constraints*.

A. Nodal and Loop Analysis

A circuit can be described by using either the mesh or the nodal formulation. The mesh equations are based on Kirchhoff’s Voltage Law (KVL), which states that the sum of voltage drops along any closed loop is zero. On the other hand, the nodal equations are based on KCL, which states that the algebraic sum of currents leaving any node is zero. A more general definition of KCL is that in any fundamental cutset that separates the network into two parts, the sum of the currents in the cutset edges is zero. If the number of branches in the network is denoted by the letter b and number of ungrounded nodes is denoted by the letter n , then to solve a circuit; (a) the number of mesh equations required is equal to $b - n$, and (b) the number of nodal equations required is equal to n .

In general nodal analysis yields less equations than mesh (loop) analysis and hence nodal analysis is usually easier to carry out [5].

B. Graph Theory

Graph Theory is used in various ways to aid circuit analysis, for example *Signal Flow Graphs*. This paper focuses on the use of graph theory to analyze the topology of electrical circuits, which is the study of inter-connected objects represented by ‘edges’ in a graph. The points where the end-points of edges touch together are formally called ‘vertices’ or ‘nodes’.

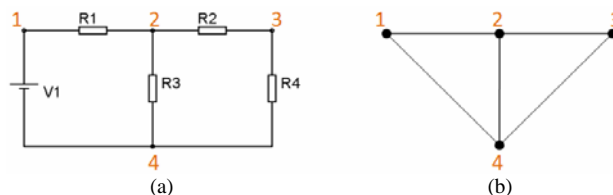


Figure 1: (a) Example Circuit1, and (b) Graph of Example Circuit.

A graph is extracted from the schematic diagram of a circuit by replacing the components with edges. For example the graph shown in fig.1(b) is extracted from the circuit shown in fig.1(a). A graph of an electrical circuit contains more than one spanning tree, and from each spanning tree a set of fundamental loops and fundamental cutsets can be extracted.

1) Spanning Tree

A *spanning tree* of a graph is defined as any set of connecting branches that connects every node to every other node without forming any closed paths or loops [5].

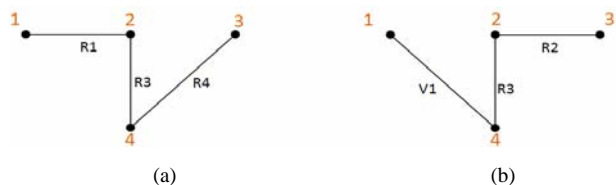


Figure 2: (a) Spanning Tree I, and (b) Spanning Tree II.

Fig.2(a) and fig.2(b) show two different spanning trees for the graph shown in fig.1(b). Once a spanning tree has been defined, the edges making part of the spanning tree are referred to as *branches*. The remaining branches are referred to as *links* or chords.

2) Fundamental Loops

A *fundamental loop* is a loop that contains one (and only one) link in its set of edges [5].

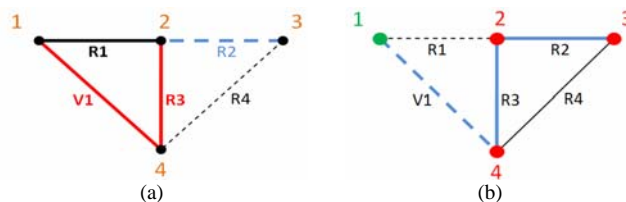


Figure 3. (a) Fundamental Loop for R1, (b) Fundamental Cutset for V1

Fig.3(a) shows the fundamental loop for link R1 when considering the *Spanning Tree* shown in fig.2(b). To construct a loop that includes only the link R1 (which is shown as a thick black line) and no other links, the tree branches shown in red must be used. Therefore the fundamental loop of R1 is made up of the edges: R1, R3, and V1.

3) Fundamental Cut Sets

A cut set is a minimal set of edges that when cut, divides the graph into two groups of nodes. A *fundamental cutset* is a cutset that contains one (and only one) tree branch in its set of edges [5]. Fig.3(b) shows the fundamental cutset for tree branch V1 when considering the *Spanning Tree* shown in fig.2(b). By cutting V1 node 1, shown in green becomes isolated from the group of remaining nodes, that is nodes 2, 3 and 4, which are shown in red. Together with branch V1, the link R1 has to be cut to keep the two groups of nodes separated, hence the complete fundamental cutset is: V1, and R1.

C. Qualitative Electrical Circuits Analysis

De Kleer [4] divides qualitative analysis of electrical circuits into two independent types of analysis, which are: (a) causal analysis, and (b) teleological analysis.

The way that the components are connected together in a circuit gives a specific structure to the circuit. The schematic diagram of a circuit describes this structure. Each component in the circuit causes some effects on the other components that are connected to it, and these in turn affect the components that are connected to them, and so on. The aim of *causal analysis* is to combine the behaviour of the individual components to explain the behaviour of the overall composite system. That said, a composite system is built so that it serves a purpose. The purpose of a circuit is also referred to as the function of the circuit. *Teleological analysis* describes how by knowing the behaviour of a circuit one can deduce its function.

Causal analysis relates *structure* to *behaviour* and *teleological analysis* relates *behaviour* to *function*. These two types of analysis were also investigated by Marc Fosséprez in 1988 [6]. Marc Fosséprez states that it is relatively easy to deduce how a circuit behaves once its function is known, but it is much harder to deduce how a circuit behaves if only the circuit's structure (its schematic diagram) is given. His work focuses on this latter task and he gives definitions about the different structures that circuits can possibly have and mathematical proofs that employ topology and graph theory.

D. Symbolic Simulators

Symbolic simulators are able to generate the transfer function of circuits input to them. The transfer function is a commonly used symbolic expression that describes how a circuit behaves. Using the transfer function the output signal that the circuit generates for a given input signal can be calculated. The advantage of using a symbolic transfer

function is that the circuit is analysed symbolically only once to obtain the transfer function and then as many numerical answers as needed can be obtained from the transfer function by substituting the symbols with the numerical values being considered.

Considerable research has been carried out on the symbolic analysis of electrical circuits in the late 1960's and a number of software Symbolic Simulators were developed in the 1980's,[7]. For example De Kleer developed a symbolic simulator called SYN together with Sussman in 1979 [8]. De Kleer states that SYN has several limitations that are were overcome in EQUAL, the Qualitative Analysis Simulator that he developed [4]. These limitations include the lack of the ability to use approximations that drastically simplify the algebra without sacrificing accuracy. Some of these problems have been addressed in modern symbolic simulators [9].

Good examples of modern symbolic simulators that are equipped with a Graphical User Interface (GUI), including a schematic capture front end are SAPWIN [1] and SNAP [2].

E. Propagation of Constraints

The propagation of Constraints has proved to be a powerful algorithm in circuit analysis. Fosséprez recommends its use when searching for a pair of compatible *current* and *voltage* (i , v) orientations, while analysing circuits qualitatively [6]. In 2006 Peter Robinson et al developed a type of software authoring tool for an 'Intelligent Book' [10] in which this algorithm is used to find the currents, voltages and component values inside different circuits. The values generated by the Propagation of Constraint algorithm are used to verify the values input by the students that make use of the 'Intelligent Book'.

III. NUMERICAL MODELS IN EDUCATION

The main author has carried out a study based on a questionnaire regarding the effectiveness of using SPICE simulators as a pedagogical tool and using handouts which explain step by step the symbolic calculations involved in electric circuit analysis. The questionnaire involved both open ended questions and Likert scale questions. The questionnaire was handed out to the students of the two first year classes of the National Diploma in Electrical and Electronics Engineering (MQF level 4) at MCAST, Malta and a total of thirty one filled in questionnaires were collected. The full report on this study is published in [11]. The report outlines two conclusions that are relevant in this paper; (a) in general although students find the SPICE simulator as motivating very few agree that it helps in understanding how circuits work, and (b) The larger proportion of students acknowledge that it would be much more useful if the simulator explains how the results are obtained. These results confirms what the other researchers that advocate the use of symbolic and qualitative have stated in their papers, which is that software that give explanations, and not just results, aids the students better.

IV. SOFTWARE TOOLS DEVELOPED

Two separate software tools were developed, the first one using MS C++ and the second one using MS C#. The first tool is text based while the second one is provided with a GUI, which makes it more adequate to be used as an eTutor.

A. Input Text Files

Both software tools require the user to input the circuit to be analysed as a text file, but the format is different in the two cases. The first tool requires the circuit to be specified in a matrix in which the rows represent nodes and columns represent components. The first row of this matrix is reserved for the components' values, that is the voltage of the battery and the resistance of each resistor. In the cells of the other rows a '1' means that terminal one of the corresponding component is connected to the node corresponding to the cell's row, '2' means that terminal two of the corresponding component is connected to the node corresponding to the cell's row and '0' means that the corresponding component is not connected to the particular node considered. The format of the input text file for the second program is more compact since in it a text line is dedicated to each component and the numbers of the two nodes to which the component is connected are stated in the corresponding line. This does away with the '0s' that were used for the first program. The other information included in each line of this text file is the X and Y coordinates of where the component is to be drawn in the GUI, the name of the component and its value.

B. Series and Parallel Reductions Program

The first software tool accepts input circuits that are made up of an arbitrary number of resistors and only one battery. This program then analyses the connections of all the resistors and identifies resistors that are connected in parallel. Each group of resistors connected in parallel is replaced by one equivalent resistor. The program then identifies serially connected resistors and replaces each group by one equivalent resistor. This process is depicted in fig.4. At each step the program outputs a matrix in text format which specifies the connections in the resultant simplified circuit.

If the resultant circuit contains other groups of resistors that are connected in parallel or in series further reductions are done. This process is repeated until no further reductions are possible.

C. Graphical Circuits Analysis Program

The aim of the program is to eTutor students that are learning how to identify a fundamental tree and the corresponding fundamental cutsets in a given circuit and how to generate the KCL current equations for each fundamental cutset. This topic is covered in a unit called 'Further Electrical Principles' that higher national diploma (MQF level 5) students follow in the second year of their course at MCAST.

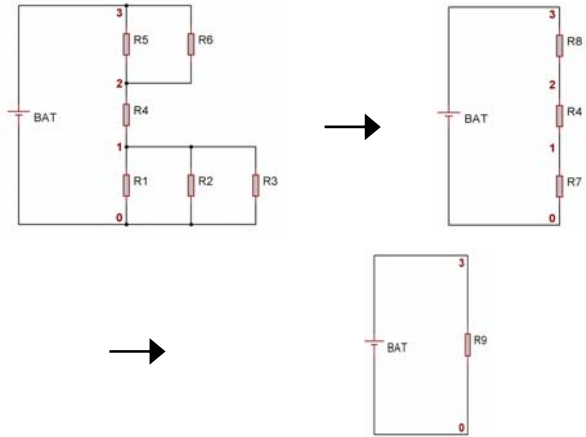


Figure 4. Example of the parallel and series resistors reduction processes carried out by the first program.

Once a circuit is specified correctly in the input text file it can be loaded in the program. Fig 5. shows an example of a loaded circuit. The user is asked to chose a spanning tree, by clicking on the components in the circuit. Once the user selects a group of components that s/he think makes up a valid spanning tree, s/he must press the 'Check Spanning Tree' button so that the program verifies if the selected group of components makes up a valid spanning tree. If it does not the program informs the user and gives relevant feedback to the user of why the selection does not make up a valid spanning tree. The program informs the user whether s/he selected the right amount of components and whether s/he captured all the nodes in the circuit with the group of components selected. The program also informs the user if there are loops present in the selection made.

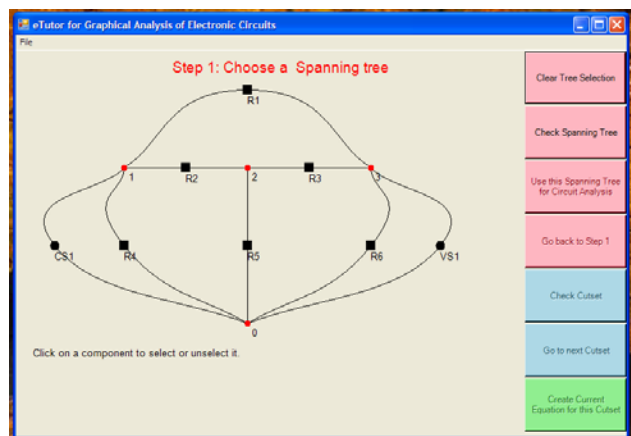


Figure 5. Example of a loaded circuit in the program's GUI

On the other hand, if the selection makes up a valid tree, if the selection makes up a valid tree the program informs the user and allows the user to select this spanning tree to continue with the circuit analysis. To do this the user has to press the 'Use this Spanning Tree for Circuit Analysis' button. Once this button is pressed the program goes into Step 2, in which the user has to select the

correct fundamental cutset for each of the tree branches inside the selected spanning tree. The tree branch for which the user has to select the links that make up the fundamental cutset is highlighted in red, as shown in fig.6.

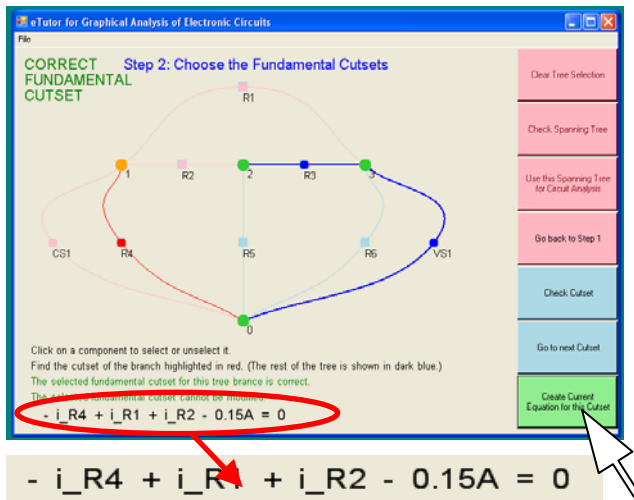


Figure 6. Example of a fundamental cutset KCL equation generated when the correct fundamental cutset is selected and the appropriate button is pressed

The fundamental cutset must separate one of the group of nodes from the remaining group of nodes. To help the user the program highlights all the nodes in one of these groups in orange and the nodes in the other group in green. After that the user selects the components that s/he thinks make up the fundamental cutset, s/he must press the ‘Check Fundamental Cutset’ button. Once this button is pressed the program checks if the selected components make up a valid fundamental cutset. If this is not the case the program gives relevant feedback to the user. The program states whether one or more components that should be included in the selection are not selected and it also states if one or more components that should not be included in the selection are in fact selected. In the case when the selected components make up a valid fundamental cutset, then the user is informed accordingly and is allowed to press the button labelled “Create Current Equation for the Cutset”. When this button is pressed the KCL equation for the fundamental cutset is generated by the program and displayed at the bottom of the screen as shown in fig.6. The user can then press the ‘Go to next Cutset’ button to find the fundamental Cutset of the next branch in the spanning tree. This process has to be repeated until the fundamental cutsets of all the branches in the spanning tree are found.

At this point it is desirable that the program tutors the user on how to find the fundamental loop for each link present in the graph, but this feature has not been implemented yet. The author plans to have this feature functional in the future so that it can be used by the higher national diploma students.

D. Algorithms in the Graphical Circuits Analysis Program

In the computer program developed, the nodes are implemented in a list. The branches or components at a given node are defined in another list. The algorithms then operate on these lists. From graph theory it is known that a valid spanning tree must be made up of $n-1$ edges, where n is the number of nodes. Hence the first check made to verify the input tentative spanning tree is to count the number of selected components and check if it equal to $n-1$. If this is not the case it means that the selected components do not make up a valid spanning tree.

The next step to carry out is to check that the selected components capture all the nodes inside the circuit (graph). The algorithm just has to go through all the selected components and mark the two nodes to which each component is connected as *captured*. After that the algorithm has to go through the nodes and check that none of them is *non-captured*. If one or more nodes are *non-captured* then the selected components do not make up a valid spanning tree. There exist cases in which the two checks explained above are satisfied but the selected components still do not make up a valid spanning tree. In this case the selected branches will not be continuously connected and at least one loop will be present in the selection. To check for such cases the spanning tree algorithm starts off with one of the selected tree branches. It checks to which nodes this branch is connected and proceeds to discover which other branches one of these nodes is connected to. If there are more than one branch connected to this node the algorithm starts considering the first branch and it takes note of which branch this is so that once it finishes checking it and returns to the last node considered, it continues looking for the correct branch. This process is repeated for each node. When at least one branch is found connected to a node the algorithm jumps to the other node to which this branch is connected and hence travels further away from the first node that it considered at the start. Naturally the larger the selected tentative spanning tree is, the more searching the algorithm has to do. But in the case of invalid spanning tree selections there are two possible ways in which the algorithm completes. One way is that the algorithm steps forward (not backwards) into a node that it already checked, and hence a loop is discovered. The other way in which the algorithm can complete in the case of an invalid spanning tree selection is that it finds out that it exhausted all the branches and nodes that are connected to the first branch considered, but it did not find all the nodes present inside the graph. In this case it means that the algorithm has found one continuous length of connected branches, which is not connected to the remaining branches of the selected tentative spanning trees. Since spanning trees should not contain any discontinuities in their branches’ connection, this means that the selected components do not make up a valid spanning tree.

Another algorithm used in the graphical analysis program is the one that highlights in different colours the

two groups of nodes that are to be separated by a fundamental cutset. The searching that this algorithm does is very similar to that done by the algorithm that verifies spanning trees. However in this case, the fundamental cutsets algorithm does not check for loops because it is used after that a valid spanning tree is already selected, so it is already guaranteed that no loops are present. The important feature that this algorithm possesses, similarly to the previous algorithm, is that it always remembers which branch it checked last when jumping from one node to another, so that when it returns back to the node from where it jumped, it continues checking from the correct branch.

V. CONCLUSIONS

An important conclusion from the questionnaire delivered to the national diploma (MQF level 4) students is that they are keen to experience software tools that help them understand how to analyse electrical circuits, by explaining the results that these programs generate. This type of software simulates the full-time availability of a tutor.

Two software tools that give explanations of the analysis carried out computationally on circuits were developed. Both these programs contain elements of Symbolic and Qualitative analysis.

One of these programs is aimed at first year national diploma (MQF level 4) students. It identifies resistors that are connected in parallel and in series and replaces them by equivalent resistors. This program was tested and verified to function correctly, but since it was developed as a text based program it is not easy for the students to use it. Hence an improvement that is needed for this program to become useful is the provision of a GUI.

The other tool is aimed at second year higher national diploma (MQF level 5) students. Its aim is to tutor these students on how to find correct spanning trees and fundamental cutsets in graphs of electrical circuits. This program was tested and verified to function correctly. It interacts with its users through a GUI. It lets the user input the circuit of interest and try to select a valid spanning tree. When a valid spanning tree is selected the program lets the user work out all the valid fundamental cutsets corresponding to this spanning tree and then generates the corresponding KCL equations. Whenever the user does an incorrect choice during the selection process, the tool explains why the choice is incorrect, and hence acts like a Tutor.

There are many possible improvements that can be done to this tool, the most important of which is the inclusion of fundamental loops selections. The feedback that this program gives to its user can also be more informative, or even better, be in increasing steps of information, according to how much help the user desires to get. In any case, even at the current stage of development this program can aid significantly the students that are learning this topic. Besides being used by the author, this program was

demonstrated to two lecturers that teach this topic and both confirmed that this program will help them deliver the concerned topic more efficiently, leading to higher success rates among students.

The tool was first tested by fifteen students that undertook courses that included the topic under consideration in the previous academic year and all these students stated that this tool would have been of great help to them. The program was then tested with a class of 18 novel HND students during the academic year (2010-2011). These students were able to choose their own personal set of branches that make up valid spanning trees and fundamental cutsets in class. This reduced the amount of time that the students needed to learn and understand these two concepts, as well as the success rate among students.

REFERENCES

- [1] A. Luchetta, S. Manetti and A. Reatti, "SAPWIN - A Symbolic Simulator as a Support in Electrical Engineering Education", *IEEE Transactions on Education*, Vol. 44, pp. 9, May 2001.
- [2] D. BIOLEK, "SNAP – program with symbolic core for educational purposes", *Proceedings of 4th World Multi-Conference on: Circuits, Systems, Communications and Computers*, ISBN 960-8052-19-X, pp. 1711-1714, July 2000.
- [3] G. Sussman and R. Stallman "Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Circuit Analysis", *Artificial Intelligence*, Vol. 9, pp. 135-196, October 1977.
- [4] J. de Kleer, "How circuits work", *Artificial Intelligence, Special volume on qualitative reasoning about physical systems*, Vol. 24, pp. 205-280, December 1984.
- [5] J. W. Nilsson and S. A. Riedel, *Electric Circuits 5th Edition*, Addison Wesley, 1996.
- [6] M. Fosséprez, *Qualitative Analysis of Non-linear, Non-reciprocal Circuits*, John Wiley & sons, 1992.
- [7] G. Gielen, P. Wambacq and W.M. Sansen, "Symbolic Analysis Methods and Applications for Analog Circuits: A Tutorial Overview", *Proceedings of the IEEE*, vol. 82, pp.287-304, 1994.
- [8] J. de Kleer and G. Sussman, "Propagation of constraints applied to circuit synthesis", *International Journal of Circuit Theory and Applications*, Vol. 8, pp. 127–144, April 1980.
- [9] H. Floberg, *Symbolic Analysis in Analog Integrated Circuit Design*, Kluwer Academic Publishers, 1997.
- [10] K. Rehman, W. Billingsley, and P. Robinson, "Writing Questions for an Intelligent Book Using External AI", *Proceedings of the Sixth IEEE International Conference on Advanced Learning Technologies*, ISBN 0-7695-2632-2, pp. 1089 - 1091, 2006.
- [11] J. Debono, "Effectiveness of using Circuit Analysis Software in Vocational Electronics Engineering Courses", *Malta College of Arts, Science and Technology (MCAST) library*, September 2010. Also available at: <https://sites.google.com/site/jasondebonoportfolio/o>

Adaptive Free-form Deformation for the Modification of CAD/CAM Data

Alexei Sacharow, Tobias Surmann, Dirk Biermann
 Institute of Machining Technology
 Technische Universitaet Dortmund
 Dortmund, Germany
 sacharow@isf.de, surmann@isf.de, biermann@isf.de

Abstract—In production engineering, the process of reverse engineering often requires modifications of CAD/CAM data. In general, CAD surfaces are modified according to a set of discrete displacement vectors. For this purpose, smoothing space-deformation techniques like free-form deformation (FFD) can be used. We present a B-spline-based adaptive FFD, which is able to ensure a user-defined shape accuracy. In an iterative process, the control-point lattice of the B-spline volume is automatically refined so that the approximation errors resulting from the direct free-form deformation decrease. Therefore, the areas inside the volume with the highest deformation are identified and subsequently refined by inserting new knots into the B-spline volume. Numerical studies have shown that the presented method improves the common B-spline-based FFD technique with respect to accuracy and efficiency.

Index Terms—CAD; free-form deformation; reverse engineering

I. INTRODUCTION

There are several applications in production engineering, in which the geometry of a designed part has to be modified. First of all, during product development, modifications of the product are often required in order to optimize or change the product properties. This process is generally called reverse engineering [1], or reengineering; it begins with a prototype being manufactured with respect to the designed CAD model (CAD - Computer-Aided Design), followed by manual modifications by the engineer. To incorporate these modifications, first, the manufactured workpiece is measured by an optical or tactile scanning device. Then, the digitized data is compared to the designed CAD model by a registration process [2]. Normally, the outcome of the registration is a discrete displacement field, which is used for the following modification of CAD/CAM data (CAM - Computer-Aided Manufacturing). Another important application is the compensation of springback in sheet metal forming. Here, the geometry of forming tools has to be modified with respect to a modification field [3], which is obtained by a finite element simulation of the forming process or by a registration of the digitized data.

Free-Form Deformation (FFD) is a technique for space deformation and is used for the modification of 3D objects [4] in computer graphics and geometric modeling. Due to its continuity and flexibility, this method was already applied for industrial applications, e.g., rapid manufacturing [5] or

the compensation of form errors by modifying CAD/CAM data [6][7]. However, approximation errors always occur, thus, the desired shape accuracy cannot be ensured. In this paper, we present an adaptive FFD method, which allows the modification of 3D shapes with regard to a specified shape accuracy.

This paper is organized as follows. In Section II, a short overview of free-form deformation techniques is provided. Section III presents the new approach for the adaptive free-form deformation using B-splines. The presented method is validated in Section IV using two examples from sheet metal forming. Finally, a conclusion is given in Section V.

II. RELATED WORK

A first approach for free-form deformation was introduced by Barr [4]. He used differential affine transformations for regular global deformations, like scaling, tapering, bending, or twisting. Additionally, rules for the transformation of tangent and normal vectors were developed. Sederberg and Parry [8] presented a more general approach for spatial deformation. They defined the deformation function as a trivariate Bernstein polynomial tensor product (Bézier volume). The FFD volume is represented by a parallelepiped lattice of control points, and the space deformation is realized by moving the control points. This FFD technique proceeds as follows:

- 1) Define a lattice of control points, which encloses the object to deform
- 2) Calculate the local parameters of every point describing the embedded object
- 3) Deform the FFD volume by moving the control points
- 4) Displace the embedded object points.

Based on the work of Sederberg and Parry, Coquillart [9] developed an Extended Free-Form Deformation (EFFD) using arbitrarily shaped, non-parallelepipedical lattices. This method uses the Bézier technique, in which the lattice size depends on the degree of Bernstein polynomials, thus, only global deformations are possible. Extensions to B-splines [10] or NURBS (Non Uniform Rational B-spline) [11] increase the flexibility of FFD and provide the possibility for local deformations. Hsu et al. [12] developed a method for the direct manipulation of FFD, which allows to control the free-form deformation by displacing object points directly. They compute the repositioning of the control points in a sense of

least squares using the pseudo-inverse matrix. Hu et al. [13] introduced an explicit and more efficient solution for the direct manipulation problem. Sarraga [6] adopted the FFD-method for modifying CAD/CAM surfaces according to displacements prescribed at a finite set of points.

Biermann et al. [7] presented an approach for manufacturing modified workpieces, in which a B-spline volume is used for the direct deformation of original NC programs (NC - Numerical Control). The requirements for the FFD were accuracy and efficiency by deforming CAD/CAM data, which consist of several thousand points. Here, the accuracy increases with the number of control points, but it is still hard to ensure that the approximation error is within the required tolerance.

III. ADAPTIVE FFD

In this section, an adaptive FFD method is presented that is able to ensure a user-defined shape accuracy by deforming an object in the described manner. The data at hand is a shape (e.g., mesh or CAD model of a workpiece) and a finite set of displacement vectors according to the desired shape modification. For this purpose, Biermann et al. [7] used a B-spline volume with a regular lattice of a fixed size. The innovation of the presented FFD method, which also uses a B-spline volume, is the iterative refinement of the lattice and recomputation of the control point positions so that the approximation error decreases in each iteration.

The adaptive FFD proceeds as follows:

- 1) Initialize a B-spline volume with a regular, paraxial lattice of minimal size
- 2) Compute the deformation according to the displacement vectors
- 3) Deform the object and update the displacement vectors
- 4) If the shape deviation (e.g., least squares distance between source and target shape) is not within the tolerance, refine the lattice and go to step 2.

In the following, the individual steps are explained in more detail.

A. Definition and initialization

The B-spline volume $F : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ of degree p , q , and r is defined by:

$$F(u, v, w) = \sum_{i,j,k=1}^{l,m,n} B_{i,p}(u)B_{j,q}(v)B_{k,r}(w)\mathbf{P}_{i,j,k}, \quad (1)$$

where $\mathbf{P}_{i,j,k}$ are the control points on a $(l \times m \times n)$ -lattice, (u, v, w) are the local parameters of a point inside the B-spline volume, and $B_{i,p}(u)$, $B_{j,q}(v)$, and $B_{k,r}(w)$ are the nonrational B-spline basis functions defined on the knot vectors

$$\begin{aligned} U &= \{\underbrace{u_{\min}, \dots, u_{\min}}_{p+1}, u_{p+1}, \dots, u_l, \underbrace{u_{\max}, \dots, u_{\max}}_{p+1}\}, \\ V &= \{\underbrace{v_{\min}, \dots, v_{\min}}_{q+1}, v_{q+1}, \dots, v_m, \underbrace{v_{\max}, \dots, v_{\max}}_{q+1}\}, \\ W &= \{\underbrace{w_{\min}, \dots, w_{\min}}_{r+1}, w_{r+1}, \dots, w_n, \underbrace{w_{\max}, \dots, w_{\max}}_{r+1}\}. \end{aligned}$$

In general, the parameter space and, therefore, the knot vectors are normalized to $[0, 1]$. This requires the embedding of the objects, where for every point $\mathbf{p}_i = (x_i, y_i, z_i)$ the local parameters (u_i, v_i, w_i) have to be found. In case of a Bézier volume, this can be easily done by solving linear equations [8]. In case of a B-spline or NURBS, due to the multiplicity of outer knots, the local parameters have to be found by numerical search [14]. Regarding the deformation of complex and large objects, this operation may be very costly. To overcome this problem, we define the initial B-spline volume as the identity so that the embedding operation vanishes. The initial B-spline volume has a regular, paraxial lattice of minimal size. This means $l = p + 1$, $m = q + 1$, and $n = r + 1$. For the normalized parameter space, the B-spline volume is equivalent to a Bézier volume, and for the embedding function $E : \mathbb{R}^3 \rightarrow [0, 1] \times [0, 1] \times [0, 1]$ yields:

$$E(x, y, z) = \left(\frac{x - x_1}{x_l - x_1}, \frac{y - y_1}{y_m - y_1}, \frac{z - z_1}{z_n - z_1} \right), \quad (2)$$

where $\mathbf{P}_{1,1,1} = (x_1, y_1, z_1)$ and $\mathbf{P}_{l,m,n} = (x_l, y_m, z_n)$. By reparameterizing the parameter space to $[x_1, x_l] \times [y_1, y_m] \times [z_1, z_n]$, the embedding function becomes identity: $E(x, y, z) = (x, y, z)$. This means that the local parameters of a point are equal to its coordinates.

B. Computing the deformation

The direct manipulation of an FFD volume requires the computation of control point displacements so that the distance between the deformed source point \mathbf{p}_i and corresponding target point \mathbf{q}_i is minimized for every pair $(\mathbf{p}_i, \mathbf{q}_i)$, $0 < i \leq d$, where d is the number of displacement vectors. This can be formulated as a least-squares problem:

$$R = \sum_{i=1}^d \|\mathbf{q}_i - F(\mathbf{p}_i)\|_2^2 \rightarrow \min. \quad (3)$$

The deformation of a point $\mathbf{p} = (x, y, z)$ is given by:

$$\begin{aligned} F(\mathbf{p}) &= F(x, y, z) \\ &= \sum_{i,j,k=1}^{l,m,n} B_{i,j,k}(x, y, z)(\mathbf{P}_{i,j,k} + \boldsymbol{\delta}_{i,j,k}) \\ &= \mathbf{p} + \sum_{i,j,k=1}^{l,m,n} B_{i,j,k}(x, y, z)\boldsymbol{\delta}_{i,j,k}, \end{aligned} \quad (4)$$

where $\boldsymbol{\delta}_{i,j,k}$ are the displacement vectors of the control points and $B_{i,j,k}(x, y, z) = B_{i,p}(x)B_{j,q}(y)B_{k,r}(z)$. The minimization problem (3) can be formulated as a set of linear equations and solved in a sense of least squares using the Singular Value Decomposition (SVD) [7].

In order to allow an iterative recalculation of control point displacements, equation (4) can be defined recursively:

$$F_s(\mathbf{p}) = F_{s-1}(\mathbf{p}) + \sum_{i,j,k=1}^{l,m,n} B_{i,j,k}(x, y, z)\boldsymbol{\delta}_{i,j,k}^s, \quad (5)$$

where $F_0(\mathbf{p}) = \mathbf{p}$ and $\delta_{i,j,k}^s$ are the displacements of the control points in the iteration s , which updates the positions of the control points from the previous iteration. By considering only the control points which were involved in the refinement of the lattice in the previous iteration, the complexity of the least-squares problem (3) decreases and, thus, the computing time is reduced significantly. In doing so, the FFD volume is optimized locally in each iteration and the shape deviations decrease continuously.

C. Refinement of the lattice

In order to reduce the approximation errors, the flexibility of the FFD volume has to be increased. One well-known technique for increasing the degrees of freedom and thereby the flexibility of B-splines is knot insertion [14]. In case of a B-spline curves, adding a new knot does not change the shape of the curve. Therefore, inserting a new knot cause addition of a new control point and the displacing of some existing control points. In case of a B-spline volume with an $(l \times m \times n)$ -lattice, e.g. inserting of a knot into the knot vector U increases the lattice size to $((l+1) \times m \times n)$ control points. By inserting knots successively into the knot vectors U , V , and W , the lattice can be expanded to an arbitrary size.

The challenge is to decide where the knots are to be inserted, so that the approximation error decreases as much as possible. Here, it is important to keep the lattice size small in order to not increase the computing time unnecessarily. Therefore, we subdivide the FFD volume into cells or subspaces and analyze the discrete deformation field inside each cell. Cells with strong deformation will be refined by knot insertion.

An intuitive subdivision of the FFD volume is given by the knot vectors U , V , and W . With respect to the initialization of the minimal B-spline volume, the parameter space is equal to the model space and the unequal knots inside a knot vector indicate the partitioning of the volume into the corresponding direction. For instance, the knot vector $U = \{\underbrace{u_{min}, \dots, u_{min}}_{p+1}, \underbrace{u_{max}, \dots, u_{max}}_{p+1}\}$ yields only

one partition of the initial B-spline volume into x -direction, in particular, $[u_{min}, u_{max}]$ with respect to parameter space or, equivalently, $[x_{min}, x_{max}]$ with respect to model space. Furthermore, inserting a knot u_1 would subdivide the model space into cells $[x_{min}, u_1] \times [y_{min}, y_{max}] \times [z_{min}, z_{max}]$ and $[u_1, x_{max}] \times [y_{min}, y_{max}] \times [z_{min}, z_{max}]$. In this process, we consider three partitionings of the FFD volume, one in each direction.

In the next step, the deformation field inside each cell is analyzed in order to decide which cell has to be subdivided. For the comparison of the cells, an appropriate measure D is required, which describes the strength of space deformation inside a cell. The simplest one is the maximum length M of all deformation vectors. This will force the refinement of at least three cells, one for each direction, which is not always necessary. In fact, the disturbance of the vector field or, rather, its gradients into the corresponding direction, e.g., gradient into x -direction for u -cells, should be taken into account by



Fig. 1. Source and target shape of the hat profile.

the measure D . Additionally, large cells should be preferred for the refinement. These requirements are combined into one measure, e.g., for the u -cells, by:

$$D_x = M \cdot L_x \cdot \frac{1}{d} \sum_{i=1}^d \delta_x(\vec{v}_i), \quad (6)$$

where L_x is the length of the cell into x -direction and $\delta_x(\vec{v}_i)$ is the partial derivative of the vector field at vector $\vec{v}_i = (\mathbf{p}_i, \mathbf{q}_i)$. We approximate $\delta_x(\vec{v}_i)$ for the k -neighborhood by:

$$\delta_x(\vec{v}_i) = \frac{1}{k} \sum_{j=1}^k \frac{\|F_s(\vec{v}_j) - F_s(\vec{v}_i)\|_2 \cdot |(\mathbf{p}_j - \mathbf{p}_i) \cdot \mathbf{b}_x|}{\|\mathbf{p}_j - \mathbf{p}_i\|_2^2}, \quad (7)$$

where $\mathbf{b}_x = (1, 0, 0)^T$ is one of the canonical basis vector in \mathbb{R}^3 and $F_s(\vec{v}_i) = (\mathbf{q}_i - F_s(\mathbf{p}_i))$ is a residual deviation vector after deformation with F_s . Note, $F_0(\mathbf{p}_i) = \mathbf{p}_i$. The deformation measures D_y and D_z for v - and w -cells are calculated analogously. Among all cells we first determine the maximum value D_{max} and then refine the cells with $D \geq \alpha D_{max}$, $0 \leq \alpha \leq 1$, by inserting a knot into the middle of the cell. By varying the scalar α , it is possible to control the number of subdivided cells: for $\alpha = 1$, only one cell is refined and, for $\alpha = 0$, all cells are refined. Due to the cubic time complexity of SVD, inserting only one knot per iteration results in the shortest total computing time, although the number of iterations and the lattice size are not necessarily minimal. Thus, we set $\alpha = 1$ in the following numerical studies.

IV. RESULTS

The presented method is validated on the basis of two examples from sheet metal forming. The first example is a simple hat profile, see Fig. 1. It is a 2.5D object, i.e., the object has no variation of the shape along the x -axis, and the corresponding mesh has 5,859 vertices. The target shape is the reference shape of the workpiece and the source shape was obtained from the target shape by bending. Thus, the displacements of the mesh vertices are known.

We use FFD for the adjustment of the source shape to the target shape with respect to the displacement vectors. Initially, the B-spline volume has a lattice with $(2 \times 4 \times 4)$ control points. The adjustment is proceeded by the adaptive FFD and the process terminates if the normalized residual $R_N = \frac{1}{d} R$ is below a user-defined threshold ϵ , which we set to 0.0001. For this simple example from Fig. 1, ten iterations were required to achieve the desired approximation quality. In this process the lattice has been refined to the size of $(2 \times 13 \times 4)$ control points in approx. 7 seconds. The calculated B-spline volume

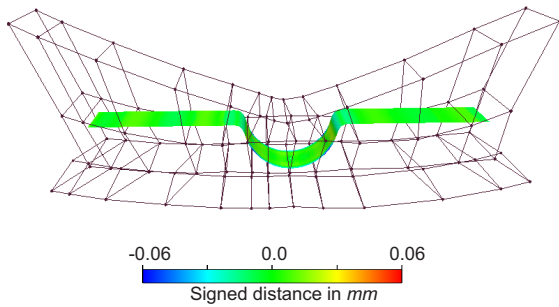


Fig. 2. Adjustment of the source shape to the target shape by adaptive FFD for the hat profile.

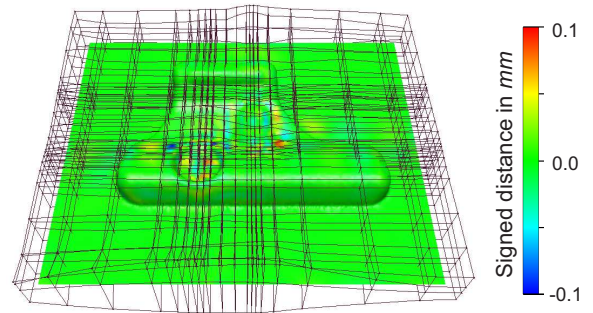


Fig. 4. Shape deviation of the tank after the adjustment by FFD.

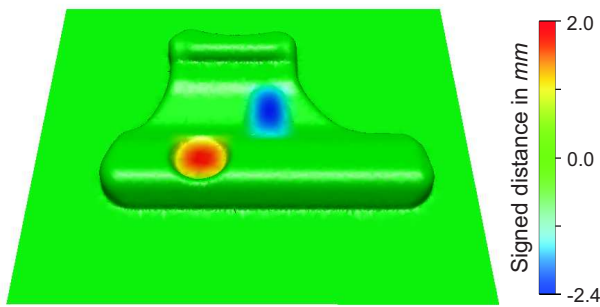


Fig. 3. Initial shape deviation of the tank.

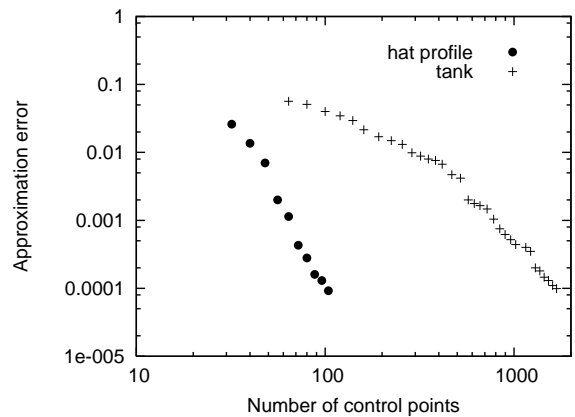


Fig. 5. Development of the approximation error R_N with regard to the number of control points.

and the deformed shape are presented in Fig. 2. It visualizes the Euclidean distance between the source and the target shape with respect to the displacement vectors after the adjustment with FFD. The maximum and average distances between the deformed source shape and the target shape were lowered from 9.0 mm and 4.958 mm to 0.06 mm and 0.007 mm , respectively. As expected, the lattice was refined into y -direction, which has the most space disturbance. For comparison, using a common, uniform B-spline volume with $(2 \times 13 \times 4)$ control points for this adjustment, the maximum and average distance were reduced to 0.08 mm and 0.014 mm , respectively, in approx. 4 seconds. For this example, the presented FFD approach outperforms the common method with respect to accuracy.

The second example is the geometry of the die for a tank wall, which we simply call the tank. The die was designed in a CAD environment and exported to a mesh with 5,577 vertices. The target shape was generated from the source shape by displacing some of the vertices, see Fig. 3. Here, the deviations of the target shape are local in nature and vary from -2.4 mm to 2 mm . This is a more challenging and practically relevant shape. Again, the displacements of vertices are known and we fit the source shape into the target shape by using the adaptive FFD.

The initial B-spline volume has a lattice with $(4 \times 4 \times 4)$ control points, which encloses both shapes. The termination threshold was set to 0.0001. Since the deformation of the tank is complex in shape, the adjustment process needs 34 iterations and a computing time of approx. 8 minutes. The

lattice was refined to the size of $(20 \times 21 \times 4)$ control points. The deviation of the source shape from the target shape after adjustment is shown in Fig. 4. The maximum and average distances have been reduced to 0.11 mm and 0.005 mm , respectively. Furthermore, it can be observed that the FFD lattice has a high density of control points in the deformed areas. The calculation of the uniform B-spline volume with the same size for 5,577 displacement vectors takes approx. 23 minutes. Thereby the maximum and average distance were lowered to 0.24 mm and 0.012 mm , respectively.

Additionally, we analyze the convergence behavior of our algorithm. Fig. 5 presents the development of the approximation error R_N with regard to number of control points. Considering the logarithmic scales, it can be seen that, in this case, the convergence behavior of the presented method is superlinear.

V. CONCLUSION

In this paper, an adaptive free-form deformation technique was presented, which is used for modifying a workpiece geometry according to a set of displacement vectors. The novelty of this method is the iterative approach, in which the lattice of the FFD volume is automatically refined with regard to the current shape deviations. Due to the locality of B-splines, only a small number of control points is recomputed

in each iteration, thus, the computing time was decreased significantly. Another advantage is the possibility to control the approximation quality of the FFD. This increases the accuracy and the efficiency of the manufacturing process, in which the FFD volume is used for shape modification of the workpiece.

The numerical studies have shown that the desired shape modifications can be obtained by deforming the object using the adaptive FFD. The presented method is more efficient than the common FFD method using a B-spline volume with a uniform control point lattice. The rate of convergence appears to be superlinear with respect to the lattice size. Further improvements can be achieved by a nonuniform subdivision of cells. For practical use, the initial lattice of FFD can be roughly adapted to the object shape by the user in order to reduce the number of iterations and speed up the calculation.

ACKNOWLEDGMENT

This work has been funded as subproject C4 of the Collaborative Research Center "3D-Surface Engineering" (SFB 708) by the German Research Foundation (DFG).

REFERENCES

- [1] T. Várady, R. R. Martin, and J. Cox, "Reverse engineering of geometric models – an introduction," *Computer-Aided Design*, vol. 29, no. 4, pp. 255–268, 1997.
- [2] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, 1992.
- [3] W. Gan and R. H. Wagoner, "Die design method for sheet springback," *International Journal of Mechanical Sciences*, vol. 46, no. 7, pp. 1097–1113, 2004.
- [4] A. H. Barr, "Global and local deformations of solid primitives," *SIGGRAPH Comput. Graph.*, vol. 18, no. 3, pp. 21–30, 1984.
- [5] L. Orazi, "Constrained free form deformation as a tool for rapid manufacturing," *Computers in Industry*, vol. 58, pp. 12–20, 2007.
- [6] R. F. Sarraga, "Modifying cad/cam surfaces according to displacements prescribed at a finite set of points," *Computer-Aided Design*, vol. 36, no. 4, pp. 343–349, 2004.
- [7] D. Biermann, A. Sacharow, T. Surmann, and T. Wagner, "Direct free-form deformation of nc programs for surface reconstruction and form-error compensation," *Production Engineering – Research and Development*, vol. 4, pp. 501–507, 2010.
- [8] T. W. Sederberg and S. R. Parry, "Free-form deformation of solid geometric models," *SIGGRAPH Comput. Graph.*, vol. 20, no. 4, pp. 151–160, 1986.
- [9] S. Coquillart, "Extended free-form deformation: a sculpturing tool for 3d geometric modeling," *SIGGRAPH Comput. Graph.*, vol. 24, no. 4, pp. 187–196, 1990.
- [10] J. Griessmair and W. Purgathofer, "Deformation of solids with trivariate b-splines," in *Proceedings of Eurographics*, 1989, pp. 137–148.
- [11] H. J. Lamousin and W. N. J. Waggenpack, "Nurbs-based free-form deformations," *IEEE Computer Graphics and Applications*, vol. 14, pp. 59–65, 1994.
- [12] W. M. Hsu, J. F. Hughes, and H. Kaufman, "Direct manipulation of free-form deformations," in *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1992, pp. 177–184.
- [13] S.-M. Hu, H. Zhang, C.-L. Tai, and J.-G. Sun, "Direct manipulation of ffd: efficient explicit solutions and decomposable multiple point constraints," *The Visual Computer*, vol. 17, no. 6, pp. 370–379, 2001.
- [14] L. Piegl and W. Tiller, *The NURBS book (2nd ed.)*. New York, NY, USA: Springer-Verlag, 1997.

On Root Classification in Kinetic Data Structures

Tomáš Vomáčka, Ivana Kolingerová
 Faculty of Applied Sciences
 Univerzity of West Bohemia
 Pilsen, Czech Republic
 tvomacka@kiv.zcu.cz; kolinger@kiv.zcu.cz

Abstract—In this paper we discuss the mathematical properties of kinetic events computation for kinetic data structures with polynomial-type certificate functions. We show that it is neither theoretically possible nor numerically safe to ignore the multiplicities of roots of these equations. The multiplicities of the roots are sometimes ignored in order to speed up the process of estimating their location, however, they must be taken into account during the management of the kinetic data structures. Some of the roots obtained by the computations of these equations do not necessarily carry the expected information (i.e., the times of future kinetic events) and they may be therefore avoided entirely during the computation. This text shows how to distinguish these roots before their exact location is computed and thus to avoid their computation.

Keywords—Computational geometry, Polynomials, Data structures.

I. INTRODUCTION

Since many modern applications use a set of moving primitives (points, triangles) or even more complex objects as the input data, the kinetic data structures (KDS). Note that this term is used in the meaning of a set of rules that defines the mutual relationships for a given set of primitives – the eventual implementation is not considered. KDS represent a valid tool in applications such as collision detection, physical and mathematical simulations, simulations of crowds, etc. We may also encounter them in less obvious applications such as kinetic-based management of function envelopes or motion interpolation.

The mathematical properties of KDS are rarely discussed in literature. In this paper, we are going to explore one specific mathematical property of kinetic data structures. Since the general topology of each kinetic data structure has to change as a result of the movement of the input data, we need to compute the time instants of these changes. Experiments have shown that this particular task is the most time-consuming part of KDS management. Therefore it would be very convenient to be able to either speed up the computation or omit some of the calculations. We show that it is possible to ignore a nontrivial part of the computation based solely on the algebraic properties of the given equations.

This text will be organized in the following fashion: in Section II, the previous work in the field of kinetic data structures will be discussed. Section III will focus

on the basic principles and properties of the kinetic data structures. Section IV will discuss the types of roots one may encounter during the management of kinetic data structures. Section V will summarize various aspects of our research and Section VI will provide the reader with the results of our work and will conclude the paper.

II. STATE OF THE ART

A. Kinetic Data Structures Use

Kinetic data structures were first introduced by Basch et al. in [1] as a means of maintaining several different data structures (such as 2D convex hull or an envelope of convex functions) over a set of moving points with the aim to create an apparatus for managing a kinetic Voronoi diagram. Since then, the kinetic data structures have been undergoing an extensive research (with a special focus on the spatial division kinetic data structures such as the aforementioned Voronoi diagram and Delaunay triangulation) – see [2]–[6] and others. There are many different fields of application of KDS: kinetic Delaunay triangulation was proposed as a means of detecting collisions in [4]. Examples of this approach include the collision detection between convex polygons by Erickson et al. who showed in [7] that the kinetic approach may be used to detect collisions between convex polygons in E^2 . Their work was further extended by Guibas et al. in [8] by employing a kinetic regular triangulation for managing the bounding spheres of the moving polyhedra. Practical use may include such applications as the one proposed by Goralski and Gold in [9] which uses the kinetic Voronoi diagram for the purpose of managing a spatial relationships between marine vessels to aid the human navigators (see Fig. 1), or the work of Ferrez – [3] which uses a kinetic regular triangulation to simulate the behavior of a granular material within a container. This particular KDS was used in order to detect the collisions between grains of various sizes (the different radii of the grains were reflected by altering the weights of the points in the triangulation).

Another field of application of KDS is the area of crowd simulation. If the crowd is simulated in the agent-based fashion, the spatial relationship among pedestrians may be managed by a kinetic data structure at the local level (i.e., to prevent collisions between pedestrians) – see [10].

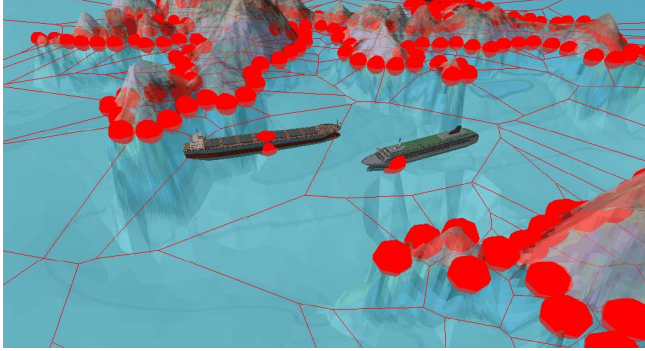


Figure 1. An example of the collision detection application of kinetic Voronoi diagram in the marine environment, [9].

The area of mathematical and physical simulations may also benefit from using KDS. Beni addressed the problem of solving partial differential equations in [6] and suggested that KDS may be used for the purposes of fluid dynamics simulations in 3D.

Although there are various types of kinetic data structures and the mathematical properties explained in this text are common to all of them, we will often explain them on the specific case of kinetic Delaunay triangulation (KDT) which is a typical and the most widely used KDS.

B. Mathematical Properties of KDS

Several different papers and theses have discussed the problem of mathematical properties of the kinetic data structures, namely the problem of computational complexity of KDS – see [2] and the problem of speedup by making the necessary computation more efficient or reducing their amount – see [5], [11]. Since the KDS management is often strongly dependent on the method of the computation of the kinetic events, which is commonly in the form of solving polynomial equations (see further), various methods for this particular task have been considered. Because these polynomials are most often nontrivial (with degree at least four) it is not practical or even not possible to solve them analytically. The methods most commonly used for solving these equations are the eigenvalue methods [12], methods based on interval arithmetic [5], [11] or various kinds of hybrid methods [11]. Since the computation is highly time-consuming, it is often convenient to speed up the process by exploiting various features of the polynomial functions. For this purpose, such tools as the Sturm sequences of polynomials or Descartes' rule of signs are sometimes employed, allowing us to separate the roots more effectively [13]. Another method of simplifying the computation is to replace the original polynomial equations with different equations with the same positions of the roots and their multiplicities reduced to one as suggested in [11]. In this text we will show that this modification is not correct and may lead to distortions in the topology of the managed KDS if the further

root management is based on the polynomial equations with reduced root multiplicities. As far as the root location goes, this method of simplification is possible, however, we will show that finding the roots with even multiplicities may be omitted entirely as they are not necessary for the KDS management.

III. KINETIC DATA STRUCTURES

In order to preserve the properties of a data structure for the mobile data, a *flight plan* is added to the primitives in the construction set – a continuous motion function which describes its movement. Furthermore, a means of computation of the events is implemented and a queue for storing these events is utilized. The trajectory of each primitive is therefore described by a function (often restricted to a function type such as real polynomial) for which we are able to compute and sort its roots (that determine the aforementioned events). The properties (as well as the use) of the queue may be found in [6]; they are not discussed here as they are not important for our purposes.

A. Predicates and Certificates

Definition 1 (Predicate): Let us have a set $\mathbf{P} = \{P_1, P_2, \dots, P_n\}$ of n primitives, a data structure $\mathbf{DS}(\mathbf{P})$ constructed over these primitives and a finite set of discrete values $\mathbf{V} \subset \mathbb{R}$. The following function:

$$p : p(\bar{\mathbf{P}}) \rightarrow \mathbf{V} \quad (1)$$

where $\bar{\mathbf{P}} \subset \mathbf{P}$ is a subset of the set of the primitives of a given (pre-defined) size, is called a predicate of $\mathbf{DS}(\mathbf{P})$.

An example of a predicate may be for instance the orientation test or incircle test, see later. The predicates often consider only the location of the primitives, but it is not a general rule (e.g., the point weights are considered in the case of regular triangulation). The predicates *define* the data structures as they are used by the *certificates* to determine the correctness of these structures:

Definition 2 (Certificate): The evaluation of a predicate function p (as defined earlier) for a given subset $\bar{\mathbf{P}} \in \mathbf{P}$ in a given data structure $\mathbf{DS}(\mathbf{P})$ is called a certificate. With respect to the parameters of p and $\mathbf{DS}(\mathbf{P})$, a certificate may either yield a *failure*, a *correct state* or a *singular state*.

Let us show an example of the *incircle test* – a predicate for a Delaunay triangulation $\mathbf{DT}(\mathbf{P})$ over a given set of n points in Euclidean plane $\mathbf{P} = \{p_1, p_2, \dots, p_n\}$ (where $p_i = [x_i, y_i]$) [14] and the associated certificate. For the sake of simplicity, let us assume that all the triangles in the triangulation are oriented counter-clockwise.

$$I(p_i, p_j, p_k, p_l) = \text{sgn} \left(\det \begin{bmatrix} x_i & y_i & x_i^2 + y_i^2 & 1 \\ x_j & y_j & x_j^2 + y_j^2 & 1 \\ x_k & y_k & x_k^2 + y_k^2 & 1 \\ x_l & y_l & x_l^2 + y_l^2 & 1 \end{bmatrix} \right) \quad (2)$$

where $p_i, p_j, p_k, p_l \in \mathbf{P}$ and $\mathbb{V} = \{-1, 0, 1\}$. The function I determines the position of a point against a circumcircle of a triangle given by the other three points. We may see that it satisfies the formula given by (1). The associated certificate is then defined in such a way that it yields a *correct state* if the result of (2) is -1 (point p_l lies outside the circumcircle of the triangle given by $p_i p_j p_k$), a *singular state* if the result is 0 (point lies on the circle) and a *failure* if the result is 1 (point lies inside the circumcircle).

Definition 3 (Certificate function, certificate failure):

Let us have a kinetic data structure $\mathbf{KDS}(\mathbf{P}^k)$ defined on a set of n kinetic primitives $\mathbf{P}^k = \{P_1^k(t), P_2^k(t), \dots, P_n^k(t)\}$ (i.e., their properties are functions of time $t \in \mathbb{R}$). Let us have a subset $\bar{\mathbf{P}}^k \subset \mathbf{P}^k$ of a given predefined size as in Def. 1. The *certificate* function is then defined as:

$$c : c(t, \bar{\mathbf{P}}^k) \rightarrow \mathbb{R} \quad (3)$$

Let us assume that c yields a correct state, a singular state and a failure $\iff c > 0, c = 0, c < 0$ respectively. Given a time value $t_f \in \mathbb{R}$ and a subset $\mathbf{P}_f^k \subset \mathbf{P}^k$ for which $c(t_f, \mathbf{P}_f^k) = 0$ and there are $\varepsilon_1, \varepsilon_2 > 0$ such that $\forall t_1 \in (0, \varepsilon_1), t_2 \in (0, \varepsilon_2) : c(t_f - t_1, \mathbf{P}_f^k) > 0 \wedge c(t_f + t_2, \mathbf{P}_f^k) < 0$, we call t_f a time of certificate failure.

As we may see, unlike (1), the image of function (3) $\mathcal{I}(c) = \mathbb{R}$. This is caused by the fact that the certificate functions are used not only to determine whether the KDS is in a correct state but also to determine the time instants when a certificate failure occurs (if any). Therefore, the most important mathematical challenge in the management of a kinetic data structure is the task of computing the roots of the certificate functions.

B. Primitives Movement

As stated before, the kinetization of a data structure is based on the fact that some properties of the underlying primitives become a function of time. The most straightforward example of this behavior is of course movement – for instance a Delaunay triangulation constructed over a set of points which move over time becomes a kinetic Delaunay triangulation. An example of non-moving data may be for example sorting of time-dependent values.

It is obvious that in reality, the movement may follow virtually any type of trajectory, but in the kinetic data structures we usually limit the movement to polynomial type trajectories in order to keep the computations manageable. If some type of a more complex trajectory is required by the application, it is usually approximated by piecewise-polynomial curves.

C. Kinetic Delaunay Triangulation

Let us now consider the special case of kinetic $\mathbf{DT}(\mathbf{P})$: we will use a kinetic data set $\mathbf{P}^k = \{p_1(t), p_2(t), \dots, p_n(t)\}$ where $p_i(t) = [x_i(t), y_i(t)]$ is a point in the Euclidean plane with coordinates being functions of time. We may see

that since the used data structure is a (kinetic) Delaunay triangulation, the associated certificate is the incircle test. However, since the coordinates of the generating points are functions of time, the incircle test itself becomes a function of time, thus forming a certificate function:

$$I^k(p_i(t), p_j(t), p_k(t), p_l(t)) = \det \begin{bmatrix} x_i(t) & y_i(t) & x_i^2(t) + y_i^2(t) & 1 \\ x_j(t) & y_j(t) & x_j^2(t) + y_j^2(t) & 1 \\ x_k(t) & y_k(t) & x_k^2(t) + y_k^2(t) & 1 \\ x_l(t) & y_l(t) & x_l^2(t) + y_l^2(t) & 1 \end{bmatrix} \quad (4)$$

The exact properties of (4) depend solely on the nature of the movement of the kinetic primitives. If we only allow movement along polynomial trajectories, function (4) will become a polynomial function. The roots of this function will determine the time instants when any four points from \mathbf{P}^k become cocircular and the data structure reaches a state of certificate failure.

IV. ROOT CLASSIFICATION

Note that not all of the roots of a certificate function are usable as time instants of topologic event occurrence. These roots are easily recognizable by considering their multiplicities. Some of the existing solutions to the KDS management problem [11] recommend that prior to the actual root computation, their method replaces each certificate function $c(t)$ with a polynomial function $d(t)$ which has the same roots but all with multiplicities equal to one. This approach is not correct, as we may see in Fig. 2.

In this figure, we may see that the point p_4 moves tangentially to the circumcircle of the triangle $p_1 p_2 p_3$ (which is not moving). There is only one (double) root of the certificate function $c(t)$ which corresponds to the time instant when the triangulation reaches the singular state depicted in Fig. 2(b) (for this single time instant, both possible triangle configurations are Delaunay-legal). This situation is the simplest possible case of this type of event and it is recognizable by obtaining a double root of the certificate function. If we reduce the root multiplicity and schedule a single event for this time, the triangulation will cease to be Delaunay and eventually it may even cease to be a triangulation. Therefore, it is necessary to be able to recognize these cases. We may see that there is no certificate failure, because there is no time period for which the associated certificate function would yield a failure. According to Def. 3 the existence of such a time period is a necessary condition for a certificate failure. The following lemma shows how these situations can be detected for polynomial certificate functions:

Lemma 1: When determining the times of topologic events, the roots of a polynomial certificate function $c(t)$ may be divided into two groups as follows:

- Roots of even multiplicity may be ignored.
- Roots of odd multiplicity determine the time of a single topologic event.

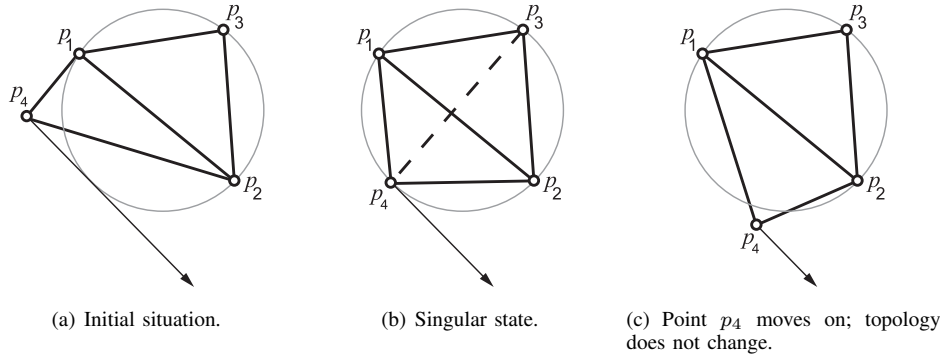


Figure 2. Three phases of tangential movement of a point against the circumcircle of a triangle.

Proof: Let us rewrite the polynomial $c(t)$ as:

$$c(t) = (t - t_0)^r \cdot q(t) \tag{5}$$

where t_0 is a root of $c(t)$ with multiplicity r and $q(t)$ is a polynomial function. Let us now find an arbitrary small $\varepsilon > 0$ such that there will be an interval $I = (t_0 - \varepsilon; t_0 + \varepsilon)$ such that $q(t)$ has no roots in I (i.e., r is the maximum positive integer that satisfies the formula); let $c_0(t) = (t - t_0)^r$.

If t_0 is of even multiplicity, we may say that $r = 2k$ and thus:

$$c_0(t) = [(t - t_0)^2]^k \tag{6}$$

We may see that $\forall t \in \mathbb{R} : c_0(t) \geq 0$ and since $q(t)$ does not have any roots in I (and thus the sign of its value does not change over I because it is a continuous function), we may clearly see that the sign of $c(t)$ does not change over I (if the zero at $t = t_0$ is ignored). Moreover, we may see that the certificate with the certificate function $c(t)$ does not fail for any time $t \in I$ and since t_0 is the only root of $c(t)$ in I , it does not mark a certificate failure.

If t_0 is of odd multiplicity, then $r = 2k + 1$ and we have:

$$c_0(t) = [(t - t_0)^2]^k \cdot (t - t_0) \tag{7}$$

We may see that the sign of $c_0(t)$ does change exactly once in the interval I and thus the sign of $c(t)$ changes too and the certificate function fails, determining the time of a single topologic event. ■

Lemma 1 shows that the multiplicities of the roots of $c(t)$ need to be taken into consideration when handling the kinetic data structures. If we, for instance, replace the certificate function $c(t)$ with other polynomial function $d(t)$ which has the same roots but all with multiplicities equal to one, we will incorrectly obtain a certificate failure for the cases similar to the one shown if Fig. 2(b). This would cause a nonexistent topologic event to be executed and lead to a topology distortions upon the event execution (thus breaking the precondition of having a specific data structure). Finally, due to these errors, the whole process of managing the KDS will probably become unstable as the distorted topology may

cause scheduling even more nonexistent events until it is virtually impossible (or at least meaningless) to handle the damaged data structure. Using the information in Lemma 1, we may safely ignore this case and avoid the damage to the data structure.

The situation depicted in Fig. 2 is specifically related to the special case of the kinetic Delaunay triangulation and, as stated before, it is recognizable by obtaining a double root of the certificate function. For different kinetic data structures, similar examples may include a point moving tangentially towards the convex hull of the data set for kinetic convex hull management or comparison of such time-dependent values as $v_1(t) = t^2$ and $v_2(t) = 0$ for kinetic sorting as shown in Fig. 3. Note that these situations are also mathematically recognizable by finding double roots of the associated certificate functions.

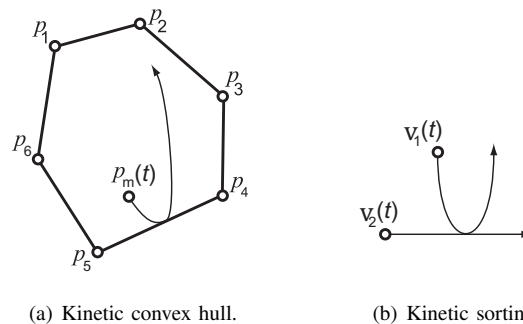


Figure 3. More examples of KDS configurations with roots of even multiplicity.

V. DISCUSSION

Lemma 1 shows that it is necessary to distinguish between the types of roots (given by their multiplicities) obtained during KDS management since not all of them are viable for topologic event determination. Even though the principle was presented on the example of kinetic Delaunay triangulation, the same situation will occur for any type of kinetic data

structure if the time change of the primitives is described by polynomial equations.

However, in practical applications, the limited precision of floating-point representation of numbers will most probably effectively prevent vast majority of these cases from occurring by slightly altering the polynomials in such a way that the multiple roots will be separated. In such cases, the roots will be processed in the usual way one after another and the resulting data structure *may* stay correct. However, this feature cannot be relied on, especially for the roots of multiplicities greater than two, and it is obviously numerically more stable to try to avoid these cases completely.

Even though the presented theory is demonstrated only on the example of kinetic Delaunay triangulation, it is applicable on any type of kinetic data structure that is based on polynomial certificate functions since Lemma 1 only considers the multiplicities of the roots of a polynomial certificate function and is thus independent on the specific meaning of these equations (and the nature of the parenting kinetic data structure).

VI. CONCLUSION

We show that it is not mathematically correct to simplify the polynomial certificate equations by reducing the multiplicities of all of their roots to one. It is not correct if done in order to speed up the computation by processing them as simple roots regardless of their original multiplicity during the KDS management. The presented theory shows that this simplification is only valid for roots of odd multiplicity. The roots of even multiplicity should either preserve even multiplicity or they should be removed entirely because they provide us with no information about the changes in the kinetic data structure. This would also simplify the certificate equation even further.

ACKNOWLEDGEMENT

The work was supported by the *UWB grant SGS-2010-028 Advanced Computer and Information Systems* and by *Ministry of Education project Kontakt II – LH11006 INGEM – Interactive Geometric Models for Simulation of Natural Phenomena and Crowds*.

REFERENCES

- [1] J. Basch, L. J. Guibas, and J. Hershberger, "Data structures for mobile data," *Journal of Algorithms*, vol. 31, no. 1, pp. 1–28, 1999.
- [2] G. Albers, L. J. Guibas, J. S. B. Mitchell, and T. Roos, "Voronoi diagrams of moving points," *International Journal of Computational Geometry and Applications*, vol. 8, no. 3, pp. 365–380, 1998. [Online]. Available: citeseer.ist.psu.edu/albers95voronoi.html
- [3] J.-A. Ferrez, "Dynamic triangulations for efficient 3d simulation of granular materials," Ph.D. dissertation, cole Polytechnique Fdrale De Lausanne, 2001.
- [4] M. Gavrilova, J. Rokne, and D. Gavrilov, "Dynamic collision detection in computational geometry," in *12th European Workshop on Computational Geometry*, Munster, Germany, 1996, pp. 103–106.
- [5] D. Russel, "Kinetic data structures in practice," Ph.D. dissertation, Stanford, CA, USA, 2007, adviser-Guibas, Leonidas.
- [6] L. H. Beni, "Development of a 3d kinetic data structure adapted for a 3d spatial dynamic field simulation," Ph.D. dissertation, Université Laval, Québec, 2009.
- [7] J. Erickson, L. J. Guibas, J. Stolfi, and L. Zhang, "Separation-sensitive collision detection for convex objects," in *SODA '99: Proceedings of the tenth annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1999, pp. 327–336.
- [8] L. J. Guibas, F. Xie, and L. Zhang, "Kinetic collision detection: Algorithms and experiments," in *ICRA*, 2001, pp. 2903–2910.
- [9] I. R. Goralski and C. M. Gold, "Maintaining the spatial relationships of marine vessels using the kinetic Voronoi diagram," in *ISVD '07: Proceedings of the 4th International Symposium on Voronoi Diagrams in Science and Engineering*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 84–90.
- [10] S. Goldenstein, M. I. Karavelas, D. N. Metaxas, L. J. Guibas, E. Aaron, and A. Goswami, "Scalable nonlinear dynamical systems for agent steering and crowd simulation," *Computers & Graphics*, vol. 25, no. 6, pp. 983–998, 2001.
- [11] L. J. Guibas and M. I. Karavelas, "Interval methods for kinetic simulations," in *SCG '99: Proceedings of the fifteenth annual symposium on Computational geometry*. New York, NY, USA: ACM, 1999, pp. 255–264.
- [12] K. W. Ellenberger, "Algorithm 30: numerical solution of the polynomial equation," *Commun. ACM*, vol. 3, no. 12, p. 643, 1960.
- [13] A. Ralston, *A First Course in Numerical Analysis*. McGraw-Hill, Inc.: New York, 1965.
- [14] O. Hjelle and M. Dæhlen, *Triangulations and Applications*. Berlin Heidelberg: Springer, 2006.

Advanced Space Filtering for the Construction of 3D Additively Weighted Voronoi Diagram

Michal Zemek, Martin Maňák, Ivana Kolingerová
 Department of Computer Science and Engineering
 University of West Bohemia
 Pilsen

mzemek@kiv.zcu.cz, manak@kiv.zcu.cz, kolinger@kiv.zcu.cz

Abstract—Spatial relationships among 3D spheres can be described by an additively weighted Voronoi diagram and this diagram can be used for advanced spatial analysis. The diagram can be constructed by an edge tracing algorithm. The problem is that tracing an edge is a time consuming operation, where many spheres are tested. Former approaches make it faster by using space filters and searching for spheres intersecting the filter. But they are inefficient when the spheres have very different radii. Our approach presented in this paper is designed to be fast even on this kind of data. It is based on modified space filters and the search for spheres intersecting the filter is performed in a power diagram.

Keywords-computational geometry, additively weighted Voronoi diagram, geometric filter

I. INTRODUCTION

Space partitioning schemes are often used in the analysis of space among 3D spheres. The analysis can include but is not limited to the computation of the volume and surface occupied by their union, inspecting the free volume or finding internal voids. This is especially important in the study of molecules or granular materials. To solve these tasks effectively, it is fundamental to have a good space partitioning scheme. Several kinds of Voronoi diagrams and their dual triangulations have been used in this area [1]. A diagram partitions the space into regions and each region belongs to a generator (in our case - a sphere). Especially aw-Voronoi diagrams (additively weighted) turned out to be useful [2], because they describe spatial relationships among spheres very well. But their regions can have non-linear boundaries, so their computation is not so easy as in the case of diagrams with linear boundaries.

The aw-Voronoi diagram can be constructed by the edge tracing algorithm [3], which we will describe later in Section IV-B. The basic variant of the algorithm is slow (at least quadratic time complexity), therefore, filtering approaches have been devised to decrease the number of spheres tested in the algorithm by considering only the spheres intersecting some filter (e.g., the union of a filled sphere and a half-space). Previous filtering approaches have been devised primarily for molecular data, where spheres have very similar radii and cannot be too close or too far. But in general data,

there can be very large spheres as well as very small and the distribution of their positions can be nonuniform. Previous approaches are inefficient on such data.

A. Contribution

In this paper we will present another filtering approach. The main idea is to compute a simpler power diagram for the input spheres in a preprocessing step and then use it in the construction of the aw-diagram when spheres intersecting a filter need to be found. Our filters are a modification of previous filters described in [4]. Both these filters perform well on non-uniformly distributed data, which is an advantage over the previous grid based filtering approach [5]. In contrast to [4], filters proposed in this paper are not affected by the size of the largest sphere in the input data. This is possible, because we use power diagrams and our improved version of a method for intersection detection [6] to search our filters for intersecting spheres. This way smaller and thus more effective filters than in the previous approaches can be used, but there is also some additional cost of searching in the power diagram. On some datasets, e.g., for spheres with very similar radii, our approach is slower than the previous approaches, but on dense sets of spheres with various radii, it substantially outperforms its predecessors.

B. Paper Outline

Section II summarizes the work related to the acceleration of the edge tracing algorithm. Section III describes intersection detection based on power diagrams. Section IV is dedicated to aw-Voronoi diagrams, the edge tracing algorithm and space filters. Our new filtering approach, which combines the filters and the intersection detection method, is described in Section V. Experimental results are given in Section VI and Section VII concludes the paper.

II. RELATED WORK

There are several algorithms for the construction of an aw-Voronoi diagram [3], [7], [8], [9], [10], [11] or its approximation [12]. The edge tracing and similar algorithms [3], [7], [8] are less complex and easier to implement than the others. They are based on a simple idea of tracing

Voronoi edges [13], which has been also used to construct the Voronoi diagram of convex objects [14].

Because the basic variant of the edge tracing algorithm [3] is slow, space filters are used to make it faster [5]. The basic idea is to compute a space filter, which limits the spheres tested by the edge tracing algorithm to the spheres intersecting the filter. In [5], a filter is the union of a filled sphere and a half-space. In the preprocessing, all input spheres are stored in a regular grid. To search the filter for intersecting spheres, grid cells covering the filter are visited and spheres associated to these cells are tested for an intersection with the filter. The grid based approach is limited to uniformly distributed input data. Another approach overcomes this problem by using Delaunay triangulation to search the filter [4]. The idea is to compute the Delaunay triangulation of sphere centers in the preprocessing and expand the size of each filter by the radius of the greatest sphere. The search for spheres intersecting the filter is performed in the triangulation and is limited to spheres having their centers inside the filter. The filter reduces its volume during the search. This approach can be inefficient for data with spheres of very different sizes.

III. POWER DIAGRAMS FOR SPACE FILTERING

Our approach for the intersection detection between spheres and aforementioned filters (thoroughly described in Section IV-C) utilizes power diagrams and it is based on a method [6] (it is used as a black-box for the intersection detection in Section V, its details are not necessary for understanding the rest of the paper, but are inevitable for the implementation). First, let us remind the concept of power diagrams, then we will describe the idea of the intersection detection.

Let $S = \{s_1, \dots, s_n\}$ be a set of spheres in \mathbb{R}^3 , where each sphere s_i has a center $c_i \in \mathbb{R}^3$ and a non-negative radius r_i . Let $\|x-y\|$ denote the Euclidean distance between points $x, y \in \mathbb{R}^3$. Power diagrams employ a so called *power distance*. The power distance of a point x from a sphere s_i is defined as $d_{pow}(s_i, x) = \|c_i - x\|^2 - r_i^2$. The *power region* of $s_i \in S$ is the set of points $PR(s_i) = \{x \in \mathbb{R}^3 : d_{pow}(s_i, x) \leq d_{pow}(s_j, x), \forall s_j \in S, s_j \neq s_i\}$. We say that s_i is the generator of $PR(s_i)$. If $PR(s_i)$ is empty, s_i is called *redundant*. The *power diagram* of S is the set of all power regions and is denoted by $PD(S)$. A power region $PR(s_i)$ is a convex polyhedron (possibly unbounded). If two power regions $PR(s_i)$ and $PR(s_j)$ share a face, we say that they are *neighbouring* and s_i is a *neighbor* of s_j .

A. Intersection Detection

The detection of intersections between spheres and a general object using power diagrams is discussed in [6], but we have slightly improved the described approach. The idea is as follows. A power diagram is traversed a region by region, for each explored region a decision is made, which of its

neighbors have to be also scheduled and explored. Step by step, the explored area is enlarged until a certain condition, guaranteeing that there is no unknown sphere intersecting Q , is fulfilled. Now let us formalize this condition.

We assume $Q \subset \mathbb{R}^3$ is a solid without cavities. Such Q will be referred to as a *query object* Q . Let us have $PD(S)$ and a query object Q and let R be a subset of faces of $PD(S)$ forming one or more bounded or unbounded polygonal surfaces without holes, such that these surfaces do not intersect Q . Then the set R will be referred to as the *rampart* of Q , R_Q for short (see Figure 1). The *interior* of R_Q is the part of \mathbb{R}^3 bounded by R_Q and containing Q . The *exterior* of R_Q is $\mathbb{R}^3 \setminus (R_Q \cup \text{interior of } R_Q)$. The generators of all non-empty power regions, which are adjacent to R_Q and lie in the interior of R_Q , are the *guardians* of R_Q . The generators of all non-empty power regions, which lie in the exterior of R_Q , are the *invaders* of R_Q .

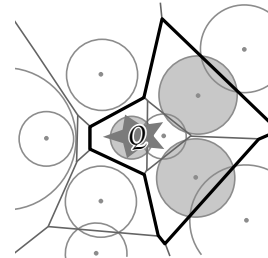


Figure 1. A rampart of Q – a thick black polyline, guardians – shaded circles (2D analogy).

Lemma 1: Given a set of spheres S and a query object Q . Let R_Q be a rampart of Q such that each guardian s_i of R_Q fulfills at least one of these conditions:

- s_i does not intersect Q , or
- s_i does not intersect any of its neighboring invaders.¹

Then no invader of R_Q intersects Q .

Proof: is an analogy to the proof described in [6] and, for a sake of brevity, it is omitted here. ■

This lemma leads to the following algorithm. At the beginning, all the regions are supposed to be unknown. Starting with a region containing some point of Q , the power diagram is traversed by a breadth-first search. Each time an unknown region $PR(s_i)$ is explored, it is marked as known and its generator and the faces shared with its unknown neighbors are tested. Let f be a face shared by the known region $PR(s_i)$ and its unknown neighbor $PR(s_j)$. The region $PR(s_j)$ is *not* scheduled if and only if both the following conditions are met:

- $Q \cap f = \emptyset$,
- $Q \cap s_i = \emptyset$ or $s_i \cap s_j = \emptyset$.

¹This condition is the difference from the original lemma in [6]. It reduces the size of explored area for a minimal additional cost.

This is done repeatedly until no scheduled region remains. In the end, the faces shared by the known and unknown regions make up the rampart R_Q satisfying Lemma 1 and all spheres intersecting Q have been found.

This algorithm does not deal with redundant generators. This issue is discussed in [6].

IV. CONSTRUCTION OF AW-VORONOI DIAGRAM

In this section, we will define the additively weighted Voronoi diagram, mention an algorithm for its construction and then describe filters that will help the algorithm to run faster. These filters are based on [4], but are smaller and thus more effective.

A. Additively Weighted Voronoi Diagram

The *aw-distance* of a point $x \in \mathbb{R}^3$ from $s_i \in S$ is defined as $d_{aw}(s_i, x) = \|c_i - x\| - r_i$. The *aw-Voronoi region* of $s_i \in S$ is the set of points $VR(s_i) = \{x \in \mathbb{R}^3 : \forall s_j \in S, s_j \neq s_i, d_{aw}(s_i, x) \leq d_{aw}(s_j, x)\}$. The *aw-Voronoi diagram* of S is the set of regions $VD(S) = \{VR(s_1), \dots, VR(s_n)\}$. These terms are illustrated on a 2D analogy in Figure 2(a).

In 3D, the boundary of a region consists of faces, their boundary consists of edges and the boundary of edges consists of vertices. An example of a region with boundary edges is shown in Figure 2(b). Without further restrictions, some regions, faces and edges can be unbounded, because they continue to infinity. This would complicate diagram representation and algorithms, therefore we will always assume this is handled, e.g., by a virtual sphere representing infinity. Under this assumption, all regions, faces and edges are bounded, with the only exception of pure elliptic edges.

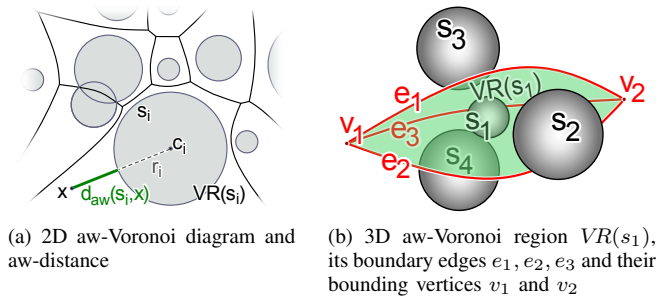


Figure 2. Additively weighted Voronoi diagram

From the geometric point of view, each face is a part of a plane or a hyperboloid, each edge is a part of a line, a hyperbola or an ellipse and each vertex is a point. But the geometrical interpretation of boundary elements is not so important. What matters is the information which input spheres define their geometry. Each region is given by one input sphere, each face by two, each edge by three and each vertex by four spheres. We will always assume that this information (which spheres define the geometry) is available for each boundary element. Only edges and vertices are

necessary to describe the diagram. Faces can be determined from edges. This simplifies the representation of a diagram to a graph of vertices and edges.

B. Diagram Construction Algorithm

The graph of Voronoi vertices and edges can be constructed by the *edge tracing algorithm* [3], [7], which works as follows. First, an initial vertex is found and four edges incident to this vertex are pushed onto a stack. Then, until the stack is empty, edges are popped from the stack and for each edge, the second bounding vertex is found. If it is a new vertex, three more edges are pushed onto the stack.

The algorithm finds the whole component of edge connectivity. Finding an initial vertex [7], elliptic edges without bounding vertices, handling infinity and discovering all components of the diagram is not necessary for understanding our paper. Important is to know how an edge is traced for the second bounding vertex, so let us describe it now.

When an edge is pushed onto a stack, only one of its two bounding vertices is known - the *start vertex* v_s . From the position of v_s and its four defining spheres, three spheres defining the edge and the direction of the edge can be determined. A candidate to the other bounding vertex can be found by taking a sphere and, together with the three spheres, computing its position. The candidate, which has its position closest to v_s , creates the other bounding vertex - the *end vertex* v_e . Comparing distances along edge can be realized via *angular distance* shown in Figure 3.

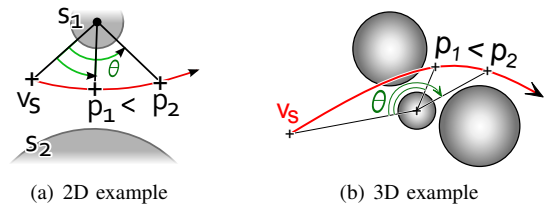


Figure 3. Comparing distances along an edge via angular distance θ - the point p_1 is closer to the vertex v_s than the point p_2

C. Space Filters

In a brute force approach, the end vertex is found by trying all spheres except the three spheres defining the edge, computing vertex candidates corresponding to these spheres and taking the one with minimal angular distance. This can be improved by using spatial filters. The idea [5] is to limit the space, which the sphere for the end vertex must intersect.

Each filter is defined individually for an edge from the knowledge of the start vertex v_s , the end vertex candidate v_e and the three spheres defining the geometry of the edge.

Let us first define an *ideal filter* $L(v_e)$. Its 2D analogy is shown in Figure 4(a). Each point x of the edge is the center of a filled ball $s(x)$ tangent to the defining spheres (or a half-space if x is at infinity). The ideal filter is

defined as the union of all such filled balls with centers $x \in [v_s, v_e]$ on the edge without the ball at the start vertex: $L(v_e) = \bigcup_{x \in \text{edge}[v_s, v_e]} s(x) \setminus s(v_s)$. The ball $s(v_s)$ cannot be intersected by any sphere since v_s is a true Voronoi vertex. If $L(v_e)$ is not intersected by another sphere, then v_e is the true end vertex, else the sphere intersecting $L(v_e)$ defines a candidate v'_e closer to v_s and another $L(v'_e) \subset L(v_e)$.

Because the shape of $L(v_e)$ is not trivial, we will cover it by a bigger filter $F(v_e)$, which is simpler. Let us first describe the filter for a non-elliptic edge with 2D analogy depicted in Figure 4(b). It is the union of two filters $F(v_e) = F_1 \cup F_2(v_e)$. The first filter F_1 is computed (in 3D) from supporting planes tangent to the spheres defining the edge as the filled ball passing through the corresponding tangent points, cut off by the supporting planes. The filter F_1 is static - it does not depend on v_e . The second filter $F_2(v_e) = s(v_e)$ is the filled ball centered at the end vertex candidate v_e or a half-space if v_e is at infinity. When the edge is an ellipse, we compute $F(v_e)$ as the smallest ball enclosing the union $\bigcup_{x \in \text{ellipse}} s(x)$. In this case, the filter is also static.²

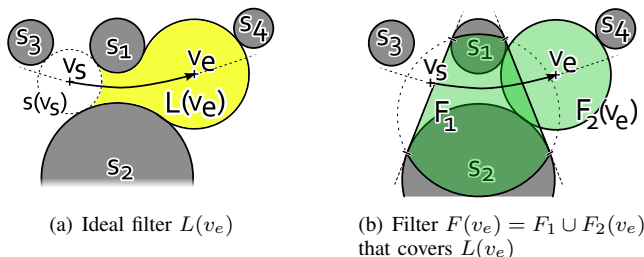


Figure 4. 2D analogy of filters for the given edge – s_1 and s_2 define the geometry of the edge, v_s is the start vertex and v_e is the end vertex candidate.

The situation in 3D is similar to the 2D analogy. The filter F_2 is a sphere or a half-space, the filter F_1 is a sphere without two half-spaces. Notice that the center of the sphere may or may not be included in F_1 as shown in Figure 5.

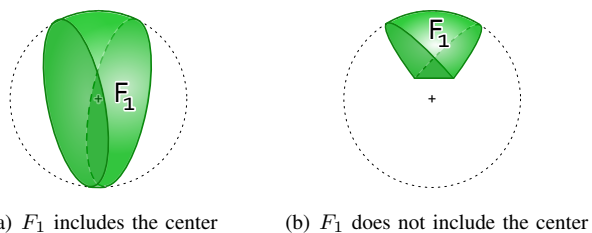


Figure 5. Two possible cases of F_1 in 3D

²Unlike the original filters [4], the proposed filters F_1, F_2 are not expanded by the radius of the biggest sphere of S . Another improvement is the reduction of the volume of F_1 by the two cutting planes.

V. PROPOSED METHOD

Here we will describe our filtering approach for end vertex search. It combines the aforementioned filters with the described method for the intersection detection.

First, a power diagram of S is computed in a preprocessing step. Then, during the construction of $VD(S)$, a filter $F(v_e)$ is created for each end vertex search. Using $PD(S)$ and the method for intersection detection described in III-A, a search for spheres intersecting $F(v_e)$ is performed. If such a sphere is not found, then v_e is the real end vertex. But if some sphere s_i intersecting $F(v_e)$ is found, it must be tested whether it defines a candidate v'_e closer to v_s than v_e . This is done by computing the positions of possible end vertex candidates v'_e and testing their angular distance against the angular distance of v_e . If v'_e is closer, this means that $L(v_e)$ is intersected by s_i , so $F(v_e)$ is replaced by $F(v'_e)$, possibly reducing the space to be further searched.

Now let us describe this processing of a filter $F(v_e)$ in detail. The filter consists of two parts, of filters F_1 and F_2 , and the search for spheres intersecting them is done independently of each other (with one exception – if a sphere intersects both filters F_1 and F_2 , its corresponding end vertex candidate is computed only once). A filter F_1 is static, it does not change during one end vertex search. A filter F_2 is dynamic – if a sphere intersecting F_1 or F_2 is found and it defines a closer end vertex candidate v'_e (i.e. with a smaller angular distance), the filter F_2 is updated. Its center is moved into the new vertex candidate v'_e and its radius is reduced so the resulting F_2 is tangent to the spheres defining v'_e . Let us note that although F_2 changes during its processing, the used algorithm for intersection detection still works correctly.³

Let us show the algorithm on an example in Figure 6. For the sake of simplicity, it is just a 2D analogy. The initial configuration is shown in Figure 6(a). There is a set of generators $\{s_1, \dots, s_{12}\}$, an edge e and a start vertex v_s . The start vertex v_s is given by a set of three (in 3D four) generators $\{s_1, s_2, s_3\}$ and a sphere inscribed to these generators. The edge e is given by a set of two (in 3D three) generators $\{s_1, s_2\}$ and the orientation. The task is to find another generator, different from s_1 and s_2 , which, together with s_1 and s_2 , creates the end vertex v_e , i.e., the vertex candidate on e closest (in terms of the angular distance) to v_s . The generator will be found among generators intersecting appropriate space filters. The initial filter $F(v_\infty) = F_1 \cup F_2(v_\infty)$ can be computed from the initial configuration. The filter F_1 is static, so it can be searched through first. In Figure 6(b), the search through F_1 starts from a region intersecting F_1 , e.g., the region $PR(s_1)$,

³ The main idea of the proof of this claim is as follows. If a sphere s_i causes an update of a filter, the updated F_2 is tangent to s_i . If F_2 intersects some sphere, then there must be a neighbor s_j of s_i such that either F_2 intersects the face generated by s_i and s_j , or $s_i \cap s_j \neq \emptyset$ and thus the region of s_j , intersecting F_2 , is to be scheduled and F_2 cannot ‘run off’ the searched area.

and explores generators s_1, \dots, s_5 . Although $s_4 \cap F_1 = \emptyset$, s_4 is explored because $PR(s_4) \cap F_1 \neq \emptyset$. Although the other explored generators intersect F_1 , none of them can create a suitable vertex candidate to reduce the dynamic filter. Next in Figure 6(c), the search through the dynamic filter $F_2(v_\infty)$ starts from a region intersecting $F_2(v_\infty)$. We have selected $PR(s_6)$ for this example, but the real selection should be based on some heuristic, which will be discussed later. So s_6 is the first generator scheduled for exploring. In Figure 6(d), s_6 is explored. Because s_6 intersects $F_2(v_\infty)$, it is used to compute a vertex candidate $v_1 \in e$. The angular distance of v_1 from v_s is less than the angular distance of v_∞ from v_s , therefore the filter is updated to $F_2(v_1)$. This is a big reduction - from a half-space to a sphere. From now on it is sufficient to check only generators intersecting $F_2(v_1)$. Because $PR(s_6) \cap F_2(v_1) = \emptyset$ and $s_6 \cap F_2(v_1) \neq \emptyset$, only neighbors of s_6 intersecting s_6 must be scheduled for exploring - at least one of them will have its power region intersecting $F_2(v_1)$. In this case, only s_9 is scheduled. Next in Figure 6(e), s_9 is explored, a closer vertex candidate v_2 computed and the filter updated to $F_2(v_2)$. Neighbors s_{10} and s_{11} are scheduled, because the edge (face in 3D) between s_9 and s_{10} and between s_9 and s_{11} intersect $F_2(v_2)$. Neither s_{10} nor s_{11} intersects $F_2(v_2)$, therefore they cannot create a closer end vertex candidate. Exploring s_{10} and s_{11} results in scheduling s_4, s_{12} , and s_2 . Next in Figure 6(f), s_4 is explored, a closer end vertex candidate v_3 computed and the filter updated to $F_2(v_3)$. The remaining generators s_{12}, s_2, s_1 and s_3 are explored, but none of them creates a closer candidate, therefore v_3 is the end vertex v_e .

Because a sphere intersecting F_1 can cause a reduction of F_2 , but not vice versa, it is better to process the filter F_1 first and then F_2 . It is also possible that F_2 becomes fully absorbed by F_1 after a filter update. In such a case the rest of the processing of F_2 can be skipped, the current end vertex candidate is the real end vertex. So it is useful to perform the $O(1)$ test whether $F_2 \subseteq F_1$ before the processing of F_2 and after each update of F_2 during its processing.

Still, filters F_2 are often very big at the beginning of their processing. We have developed two simple heuristics, each of them quickly and substantially reduces the size of F_2 . One of them tests the intersections between F_2 and all neighbors of the three spheres generating the current edge. Often, some of these neighbors intersect F_2 and thus reduce it.

The other heuristic localizes the center of F_2 in $PD(S)$ by traversing from a region to region, using a visibility walk [15]. The walk starts in the region of one of the spheres generating the current edge and locates the region containing the center of F_2 . A generator of each region explored during the walk is tested, whether it intersects F_2 . If so, F_2 is updated, the destination point is changed to the new center of F_2 and the walk continues from the lastly explored region. A performance of both heuristics is similar in practice.

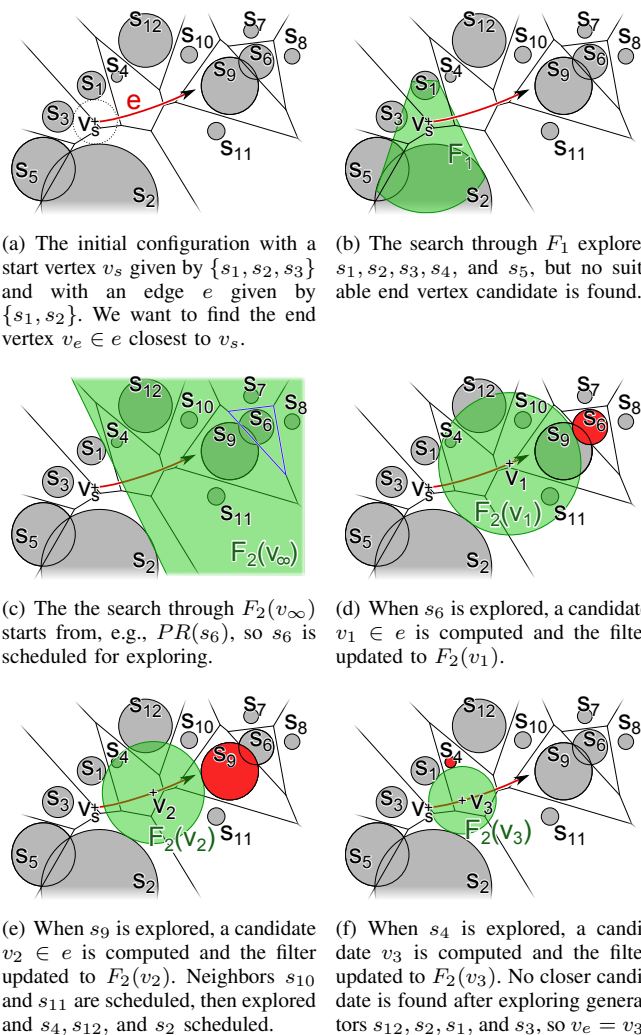


Figure 6. Algorithm example in 2D

VI. EXPERIMENTS AND RESULTS

We compared the proposed method against the method [4], based on Delaunay triangulation (hence it will be referred to as Delaunay method), because it is the best competitive method for general input data. We did not compare it against the brute force method [3], because it is too slow to be interesting. Both implementations are written in C# 3.5 and the experiments were done on Intel Core i7, 3.07GHz with 6GB RAM running Windows 7. We were interested in the running time consumed by the construction of aw-Voronoi diagrams and the average number of generators explored during one end vertex search. We distinguish a *touched* generator and a *tested* generator. To *touch* a generator means to find out whether the generator is already known (which takes $O(1)$ time). To *test* a generator means to compute its intersection with one part of a filter. Moreover in the proposed method,

it also includes a check of a local neighborhood of the generator, as described in III-A. For datasets used in our experiments, this additional check runs in $O(1)$ time in average case. But the hidden multiplicative constant is big – the test of one generator in the proposed method is roughly 10 times slower in comparison with Delaunay method. In consequence, for sets of spheres with similar radii and for sparse sets of spheres, the proposed method is roughly 2 times slower than Delaunay method. But with increasing differences in radii and increasing density of spheres, the filters used in Delaunay method (as well as the regular grid-based filtering approach mentioned in II) become ineffective, while the performance of the proposed method is almost unaffected.

This is illustrated in the following two experiments. In the first experiment, the density of the dataset is constant and the difference of radii changes. The used dataset contains 10000 points randomly scattered in a unit box and one sphere lying outside the box. The radius r of the sphere changes from 0 to 1. Figure 7 shows dependency of the construction time of the aw-Voronoi diagram on r , Figure 8 the average number of performed tests per one edge of the diagram. For a small value of r , Delaunay method outperforms the proposed method. But while the running time (as well as the number of tests) of the proposed method is minimally affected by r , the filters used in Delaunay method become inefficient for increasing r .

In the second experiment, the difference of radii is constant and the density changes. The dataset is formed by a unit box containing 10 congruent spheres with a radius 0.1 and n points, where $n \in \langle 100, 64000 \rangle$. Figure 9 shows that while the construction based on the proposed method runs approximately in $O(n^{1.2})$ time, the construction based on Delaunay method runs in $O(n^{1.9})$. Note that the proposed method outperforms Delaunay method starting at $n = 250$.

This corresponds with the average numbers of performed tests per one edge, shown in Figure 10. On the given range of n , the numbers of generators tested and touched by the proposed method increase less than 3 times, whereas the growth for Delaunay method is more than 500 times.

Now let us mention datasets, which are suitable neither for the proposed, nor for the previous filtering approaches. The filtering approaches are inefficient for datasets with many mutually intersecting spheres. The dataset used in the last experiment consists of n unit spheres with centers randomly scattered in a unit box and of one point. This point prevents the improving of the performance with a trick, where the radius of each sphere is reduced by the minimal radius. In consequence, filters are intersected by almost all spheres and the diagram construction runs approximately in $O(n^2)$, regardless of the used filtering approach. Here the original edge tracing algorithm outperforms the filter-based methods, nevertheless, it also runs in quadratic time. Recall that our filter is an approximation encapsulating the ideal filter.

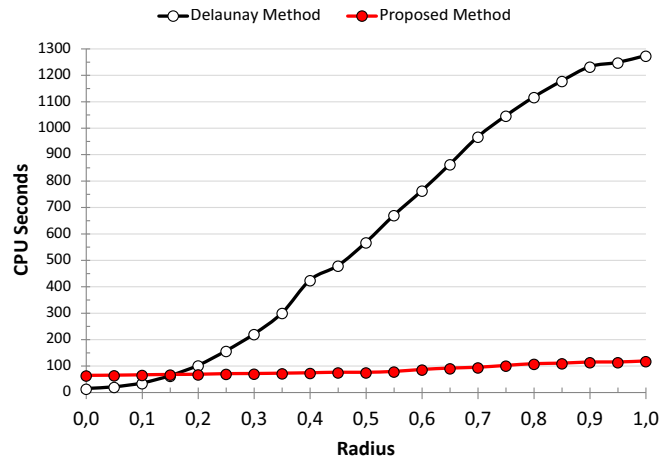


Figure 7. Aw-Voronoi diagram computation time – 10000 random points in a unit box and one sphere of the given radius.

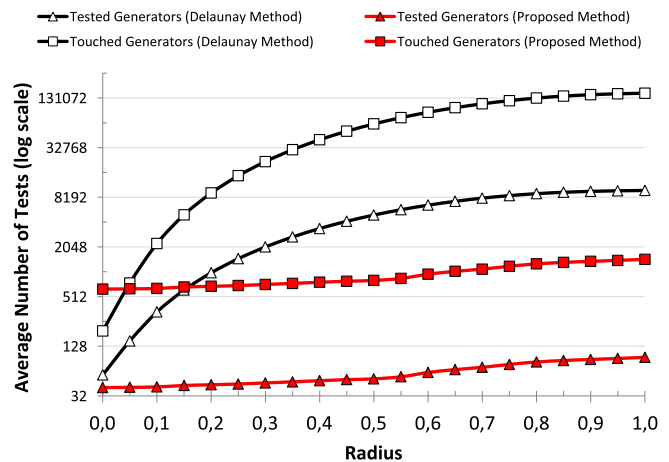


Figure 8. Average number of tests per one aw-Voronoi edge – 10000 random points in a unit box and one sphere of the given radius.

Another example of data unsuitable for our filtering approach is data with high probability that the power region of a sphere intersects the approximation filter but the sphere does not intersect the ideal filter. In this case many spheres will be explored but only a few of them will actually reduce the filter.

Our implementation runs in only one CPU thread. When a diagram is being constructed, many edges must be traced. With the increasing number of computation cores in modern hardware, it might pay off to trace more edges in parallel, using our filtering approach to speed up the tracing.

VII. CONCLUSION

We have proposed a new filtering approach for a construction of aw-Voronoi diagrams. The proposed approach combines filters based on [4] and a method for the intersection detection [6]. We have slightly improved both these two techniques. We have shown that on dense sets of

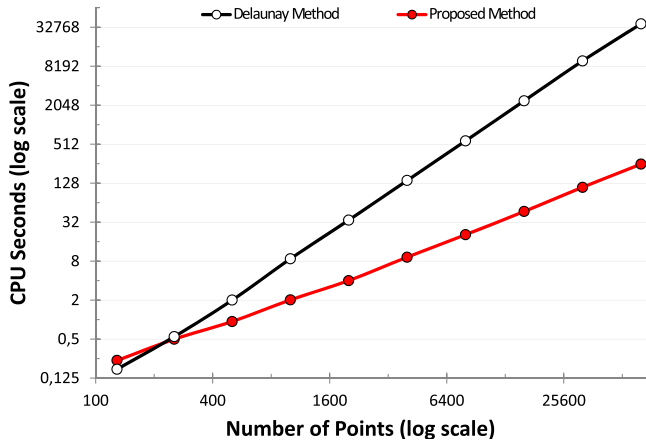


Figure 9. Aw-Voronoi diagram computation time – 10 spheres with a radius 0.1 together with an increasing count of points in a unit box.

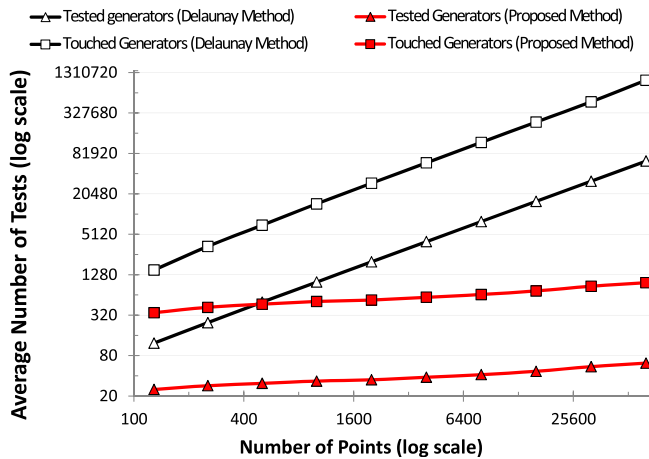


Figure 10. Average number of tests per one aw-Voronoi edge – 10 spheres with a radius 0.1 together with an increasing count of points in a unit box.

spheres with various radii the proposed filtering approach significantly outperforms the previous Delaunay approach.

As a future work, we recommend to investigate a hybrid method combining the proposed and Delaunay approaches together with the original edge tracing algorithm. First, a simple statistic on an input set of spheres would be computed and according to its result, the most suitable technique would be chosen for the diagram construction. Another open problem is to design filters suitable for a set of mutually intersecting spheres.

ACKNOWLEDGMENT

This work has been supported by the Czech Science Foundation under the project P202/10/1435 and by the University of West Bohemia under the project SGS-2010-028.

REFERENCES

- [1] A. Poupon, "Voronoi and Voronoi-related tessellations in studies of protein structure and interaction," *Current Opinion in Structural Biology*, vol. 14, no. 2, pp. 233 – 241, 2004.
- [2] D.-S. Kim, "A beta-complex solves all geometry problems in a molecule," in *The 6th International Symposium on Voronoi Diagrams in Science and Engineering*, 2009, pp. 254–260.
- [3] D.-S. Kim, Y. Cho, and D. Kim, "Euclidean Voronoi diagram of 3d balls and its computation via tracing edges," *Computer-Aided Design*, vol. 37, pp. 1412–1424, 2005.
- [4] M. Manak and I. Kolingerova, "Fast discovery of Voronoi vertices in the construction of Voronoi diagram of 3d balls," *International Symposium on Voronoi Diagrams in Science and Engineering*, vol. 0, pp. 95–104, 2010.
- [5] Y. Cho, D. Kim, H.-C. Lee, J. Y. Park, and D.-S. Kim, "Reduction of the search space in the edge-tracing algorithm for the Voronoi diagram of 3d balls," in *ICCSA (1)*, 2006, pp. 111–120.
- [6] M. Zemek and I. Kolingerová, "Power diagrams and intersection detection," in *ICCSA*, vol. 3, 2011, pp. 163–173.
- [7] N. N. Medvedev, V. P. Voloshin, V. A. Luchnikov, and M. L. Gavrilova, "An algorithm for three-dimensional Voronoi S-network," *Journal of Computational Chemistry*, vol. 27, no. 14, pp. 1676–1692, 2006.
- [8] D.-S. Kim, D. Kim, Y. Cho, and K. Sugihara, "Quasi-triangulation and interworld data structure in three dimensions," *Computer-Aided Design*, vol. 38, no. 7, pp. 808–819, 2006.
- [9] J.-D. Boissonnat and C. Delage, "Convex hull and Voronoi diagram of additively weighted points," in *ESA*, 2005, pp. 367–378.
- [10] D. Kim, Cho, and D.-S. Kim, "Region expansion by flipping edges for Euclidean Voronoi diagrams of 3d spheres based on a radial data structure," in *ICCSA (1)*, 2005, pp. 716–725.
- [11] D. Kim and D.-S. Kim, "Region-expansion for the Voronoi diagram of 3d spheres," *Computer-Aided Design*, vol. 38, pp. 417–430, 2006.
- [12] E. Yaffe and D. Halperin, "Approximating the pathway axis and the persistence diagram of a collection of balls in 3-space," in *SCG '08: Proceedings of the twenty-fourth annual symposium on Computational geometry*. New York, NY, USA: ACM, 2008, pp. 260–269.
- [13] M. Tanemura, T. Ogawa, and N. Ogita, "A new algorithm for three-dimensional Voronoi tessellation," *Journal of Computational Physics*, vol. 51, no. 2, pp. 191 – 207, 1983.
- [14] V. A. Luchnikov, N. N. Medvedev, L. Oger, and J.-P. Troadec, "Voronoi-delaunay analysis of voids in systems of nonspherical particles," *Phys. Rev. E*, vol. 59, no. 6, pp. 7205–7212, Jun 1999.
- [15] O. Devillers, S. Pion, and M. Teillaud, "Walking in a triangulation," *Internat. J. Found. Comput. Sci*, vol. 13, pp. 106–114, 2001.

A Novel Approach for Detection of Copy-Move Forgery

Mengyu Qiao¹, Andrew H. Sung², Qingzhong Liu³, Bernardete M. Ribeiro⁴

¹Dept. of Math and Computer Science, South Dakota School of Mines and Technology, mengyu.qiao@sdsmt.edu

²Dept. of Computer Science and Engineering, New Mexico Tech, sung@cs.nmt.edu

³Dept. of Computer Science, Sam Houston State University, qxl005@shsu.edu

⁴Dept. of Informatics Engineering, University of Coimbra, bribeiro@dei.uc.pt

Abstract—With the increasing popularity of digital media and the ubiquitous availability of media editing software, innocuous multimedia are easily tampered for malicious purposes. Copy-move forgery is one important category of image forgery, in which a part of an image is duplicated, and substitutes another part of the same image at a different location. Therefore, it is necessary to have reliable and efficient methods to detect copy-move forgery for applications in law enforcement, forensics, etc. In this paper, based on multi-resolution and multi-orientation curvelet transform, we propose a blind forensics approach for the detection of copy-move forgery. In detail, the input image is segmented into overlapping blocks, and then curvelet transform is applied to each block. Statistics of curvelet sub-bands are extracted and sorted. Finally, duplicated blocks are identified by comparing their similarity. The proposed approach intends to reduce the complexity, improve the accuracy of the detection, and potentially resist shifting manipulation.

Keywords—Copy-Move Forgery; Image Forensics; Curvelet Transform

I. INTRODUCTION

With the development of high-definition acquisition equipment, large capacity storage device, and high speed network, multimedia applications have become increasingly popular in our daily life. At the same time, the imperceptible manipulation of digital media for malicious purpose has been simplified by the pervasive availability of media editing software. While we are undoubtedly exposed to huge volume of digital media, our traditional confidence in the integrity of these media has also been eroded since doctored pictures, video clips, and voices are appearing with a growing frequency and sophistication in mainstream media outlets, scientific journals, political campaigns, and courtrooms [1].

In order to protect the integrity and reveal the manipulation of digital media, two types of countermeasures, proactive approach and reactive approach, are extensively investigated in previous studies. Proactive approach, including digital signature, watermarking, and etc., relies on preprocessing before distribution, which requires additional and shared information. However, there is no universally recognized standard, and the complexity greatly restricts its application. On the other hand, the reactive approach only requires digital media without any supplemental information. Due to the variety of manipulations and the diversity of individual characteristics of media, reactive approach usually

faces difficulties at a larger scope, and suffers from complicated and time-consuming problems.

Copy-move forgery is conducted by duplicating a part of targeted image and substituting another part of the same image to increase the occurrences of certain objects, or conceal an important region. As one of the major categories of image forgery, copy-move manipulation has its unique characteristic – the source and the destination of the duplicated parts are in the same image, so the noise patterns are similar to those from surrounding regions. This important characteristic invalids the application of methods for detecting other types of tampering, such as splicing, double compression, etc. [2, 3, 4, 5]. Figure 1 illustrates an example of copy-move forgery, where the original image (a) has four different original pictures and the tampered one (b) contains an additional picture.

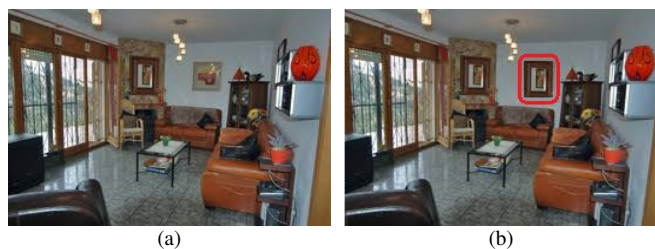


Figure 1. Example of Copy-Move forgery original image (a) and tampered image (b).

In recent years, multiple detection methods have been proposed to address copy-move forgery. As one of the common analysis of image forensics, moment statistics has been successfully applied for detecting a variety of tampering, including splicing, double compression, steganography, and etc. Mahdian and Saic proposed blur-invariant moments as features for detection [6]. Wang et al. presented first four Hu-moments for detection [7]. To reduce the dimensionality and improve efficiency, principal component analysis (PCA) [8] and singular value decomposition (SVD) [9] were applied to obtain reduced feature space. Features from other transform domains, such as discrete wavelet transform (DWT) [10] and Fourier-Mellin transform (FMT) [11] were proposed as alternative detectors.

To detect copy-move forgery, in this paper, we propose an approach based on multi-resolution and multi-orientation curvelet transform. Statistical features are extracted from curvelet sub-bands of overlapping blocks, and reduced

features are generated for similarity measure. Restrictive criteria are defined to suppress false-positive. Compared to exhaustive search in spatial domain, the proposed approach intends to reduce the complexity, improve the accuracy of the detection, and potentially resist shifting manipulation. The remainder of this paper is organized in this way: Section II introduces the curvelet transform, while Section III presents the proposed approach for detecting copy-move forgery. The experiments and results are described in Section IV. Conclusions are made in Section V.

II. INTRODUCTION TO CURVELET TRANSFORM

The concept of wavelet transform was developed to represent both location and spatial frequency in 1D signal. For 2D signal, like digital image, curvelet transform provides directional information at different scales. The definition of curvelet transform is derived from 2D ridgelet transform at multiple scales. If an image is denoted as $f(x,y)$, then the continuous ridgelet coefficients are described as [12]:

$$\mathfrak{R}_f(a,b,\theta) = \iint \psi_{a,b,\theta}(x,y) f(x,y) dx dy \quad (1)$$

where a is the scale parameter and $a > 0$, $b \in \mathbb{R}$ is the translation parameter, and $\theta \in [0, 2\pi)$ is the orientation parameter. A ridgelet can be defined as [12]:

$$\psi_{a,b,\theta}(x,y) = a^{-\frac{1}{2}} \psi\left(\frac{x \cos \theta + y \sin \theta - b}{a}\right) \quad (2)$$

where θ is the orientation of the ridgelet. Ridgelets are constant along the lines $x \cos \theta + y \sin \theta = const$ [12].

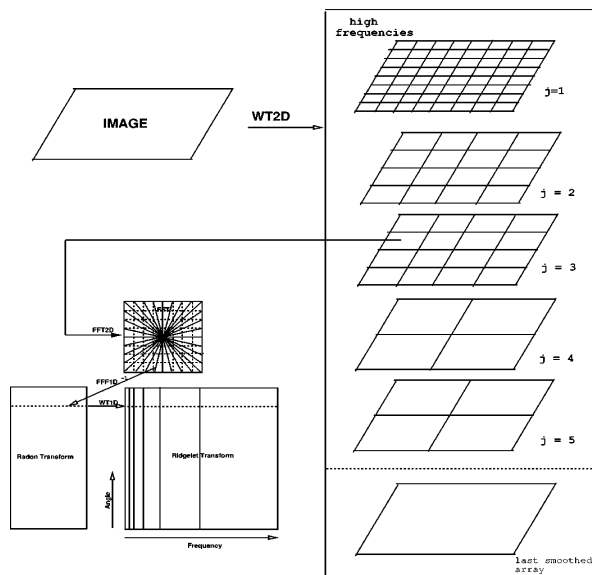
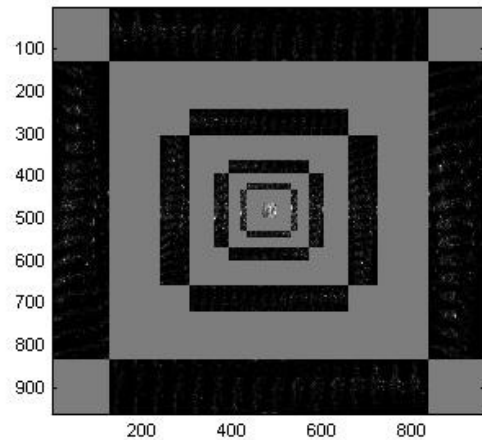


Figure 2. An overview of discrete curvelet transform.

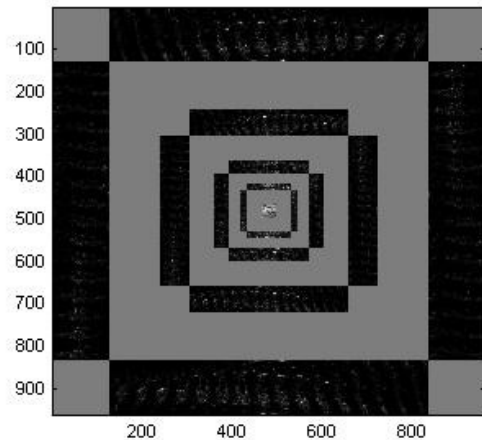
In curvelet transform, an input image is decomposed into a set of sub-bands, and each sub-band is then partitioned into several blocks for ridgelet analysis. The ridgelet transform is applied by combining the Radon transform and the 1-D

wavelet transform [12]. To achieve higher level of efficiency, curvelet transform is usually implemented in the frequency domain.

Figure 2 shows an overview of discrete curvelet transform, which generates a group of coefficient matrices in different scales and orientations [12].



(a)



(b)

Figure 3. Curvelet transform of original (a) and 90 degree rotated (b) images generated by using Curvelab-2.1.2 [14].

Since complex ridgelet transform has higher computational complexity, fast discrete curvelet transform is implemented by wrapping of Fourier samples [13]. The pyramid structure of curvelet transform generates multiple orientations at various scales, which greatly benefits the identification of duplicated regions in both accuracy and robustness. Multi-directional decomposition provides a more precise approximation of the relation between adjacent orientations, and further supports the tolerance of rotation and shifting manipulations. Figure 3 shows the curvelet transform of original (a) and 90 degree rotated (b) images, which are generated by using Curvelab-2.1.2 [14]. It is clearly illustrated that all curvelet coefficients rotate with the rotation of the image simultaneously. Therefore, the pattern

of individual orientation is preserved, and the transitions of adjacent orientations remain. This important property leaves us clue to improve copy-move forgery detection.

III. DETECTION METHOD

Inspired by the observation of multi-directional curvelet transform, we design an analytical approach to detect duplicated regions based on statistics of curvelet coefficients.

A. Structure of Approach

The proposed approach consists of two phases. In the first phase, the overlapping blocks of a source image are sorted according to statistics of multiple curvelet sub-bands. The detailed procedure of the first phase is described in Figure 4.

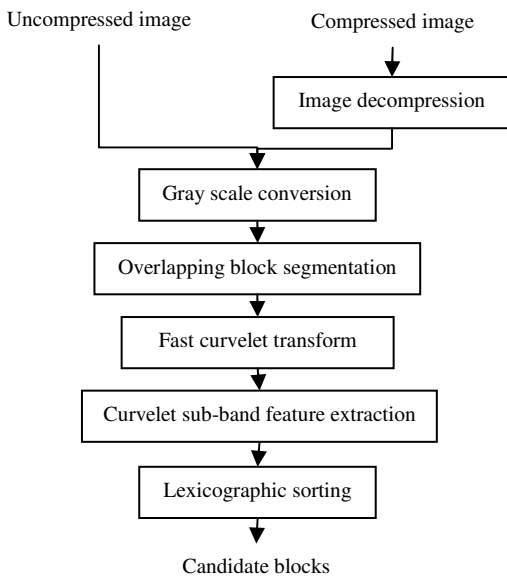


Figure 4. The structure of the first phase.

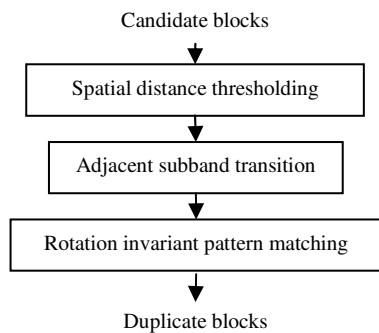


Figure 5. The structure of the second phase.

In the second phase, the number of candidate block pairs has been dramatically reduced, and the spatial distance between each pair of blocks is considered to reduce the false-positive of similar blocks from same object or texture.

Orientation related block features are extracted for pattern matching. Considering relation between adjacent orientations, the rotation or shifting effects could be minimized. The procedure of the second phase is illustrated in Figure 5.

B. Curvelet Sub-Band Feature Extraction

In order to extract statistical features, the gray scale image is segmented into a series of overlapping blocks. Then, we apply fast curvelet transform to individual blocks. Given a block $B[i,j]$ of dimension N by N , the curvelet transform could be obtained from:

$$CT(a,b,\theta) = IFFT\left(FFT(B[i,j]) \times FFT(\psi_{a,b,\theta}[i,j])\right) \quad (3)$$

According to the setting of block size N , each block is decomposed into 3 or 4 levels of scales. Different scales consist of various numbers of sub-bands. For 3 levels decomposition, the level 1, 2 and 3 include 1, 16, and 1 sub-bands respectively. We define the sub-band of curvelet transform as ct , so the 3 levels curvelet coefficients are denoted as:

$$CT = \left\{ (ct_{1,1}), (ct_{2,1}, ct_{2,2}, \dots, ct_{2,16}), (ct_{3,1}) \right\} \quad (4)$$

In order to reduce the feature dimension and resist rotation manipulation, we compute the mean values of each sub-band, and sort those in same level of scale. In 3 levels decomposition, the level 1 and 3 contain only one sub-band, so there is no further processing. Level 2 consists of 16 sub-bands, and the mean values of those are denoted as follows:

$$M_{CT\{2\}} = m_1, m_2, \dots, m_{16} \quad (5)$$

The sorted features of 3 levels decomposition could be obtained as

$$M_{CT} = \left\{ M_{CT\{1\}}, M'_{CT\{2\}}, M_{CT\{3\}} \right\} \quad (6)$$

M_{CT} will be used for lexicographic sorting. Then adjacent pairs of blocks in the rank could be provided to the second phase as candidate blocks.

C. Adjacent Sub-band Transition

In the rotation manipulation, all the sub-bands of ordinations shift at the same degree, thus the relation between adjacent sub-bands basically remains. Although the absolute values might vary, the distributions of transitions are very similar to the original ones regardless of shifting. Therefore, we obtain the differences between each adjacent pair of sub-bands in level 2 by:

$$D(i) = \left| m_i - m_{\text{mod}((i+1),16)} \right|, i = 1, 2, \dots, 16 \quad (7)$$

D. Rotation Invariant Pattern Matching

In order to minimize the directions of the candidate blocks, we sort the difference vector and use the sorted vector as a rotation invariant descriptor of each candidate

block. For each similar pair of blocks, we compute Euclidean distance between their descriptors as similarity measure. For 3 levels decomposition, the descriptor of level 2 sub-bands is used for similarity measure. The similarity between a pair of blocks p and q is obtained from:

$$S(p, q) = \sqrt{\sum_{i=1}^{16} |D_p(i) - D_q(i)|^2} \quad (8)$$

Considering small distortion introduced by rotation, compression, or noise adding, a selected threshold is set to determine duplicated blocks. For different resolutions of image and different levels of decomposition, the threshold should be adjusted accordingly.

IV. EXPERIMENTS AND RESULTS

The original 100 raw images were obtained in never compressed format used in our previous study of steganalysis [15]. We created copy-move forgery with various sizes of duplicated regions. Then the doctored images were compressed into JPEG format at quality factor of 90. Different parameter settings, which include block size and similarity threshold value, were tested to enhance the detection performance.

Figure 6 illustrates two copy-move forgery images and their identified masks with the duplicated regions. Table I shows the statistics of the detection results, which indicates good detection accuracy and stable detection performance.

TABLE I. DETECTION RESULTS OF 100 COPY-MOVE IMAGES

Average Precision	Maximum Precision	Minimum Precision	Standard Deviation of Precision
97.88%	100.00%	90.09%	0.024

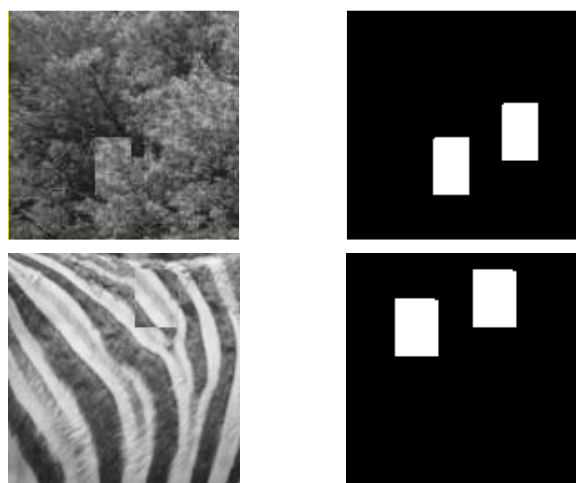


Figure 6. The copy-move forgeries and the identified duplicated regions.

V. CONCLUSIONS

In this paper, based on multi-resolution and multi-orientation curvelet transform, we presented an approach for detecting copy-move forgery of duplicated regions within

same image. Curvelet sub-band features effectively represent properties of individual region, and efficiently identify duplicated blocks. Experimental results show that the proposed approach obtains good performance in detecting duplicated regions even after JPEG compression. Moreover, this approach also preserves good potential in identifying rotation and scale manipulations in addition to the basic copy-move forgery.

ACKNOWLEDGMENT

The authors gratefully appreciate the Institute for Complex Additive Systems Analysis of New Mexico Tech, and South Dakota School of Mines and Technology for supporting this research.

REFERENCES

- [1] H. Farid, "Image forgery detection, a survey," *IEEE Signal Processing Magazine*, pp. 16-25, 2009.
- [2] H. Farid, "Exposing digital forgeries from JPEG ghosts," *IEEE Trans. Inf. Foren. Secu.*, vol. 4 (1), pp. 154-160, 2009.
- [3] B. Mahdian and S. Saic, "Using noise inconsistencies for blind image forensics, image and vision computing," vol. 27, no. 10, pp. 1497-1503, 2009.
- [4] M. Chen, et al., "Determining image origin and integrity using sensor noise," *IEEE Trans. Inf. Forensics Security*, vol. 3, no. 1, pp. 74-90, 2008.
- [5] M. Qiao, A. Sung, and Q. Liu, "Revealing real quality of double compressed MP3 audio," 18th ACM International Conference on Multimedia, pp. 1011-1013, 2010.
- [6] B. Mahdian and S. Saic, "Detection of copy-move forgery using a method based on blur moment invariants. *Forensic Science International*, vol. 171, no. 2, pp. 180-189, 2007.
- [7] J. Wang, G. Liu, Z. Zhang, Y. Dai, and Z. Wang, "Fast and robust forensics for image region-duplication forgery," *Acta Automatica Sinica*, 2009.
- [8] A.C. Popescu and H. Farid, "Exposing digital forgeries by detecting duplicated image regions," Technical Report TR2004-515, Department of Computer Science, Dartmouth College, 2004.
- [9] X. Kang and S. Wei, "Identifying tampered regions using singular value decomposition in digital image forensics," *International Conference on Computer Science and Software Engineering*, vol. 3, pp. 926-930, 2008.
- [10] M. K. Bashar, K. Noda, N. Ohnishi, H. Kudo, T. Matsumoto, and Y. Takeuchi, "Wavelet based multiresolution features for detecting duplications in images. *IAPR Conference on Machine Vision Application*, pp. 264-267, 2007.
- [11] S. Bravo-Solorio and A.K. Nandi, "Passive forensic method for detecting duplicated regions affected by reflection, rotation and scaling," 17th European Signal Processing Conference, 2009.
- [12] J.-L. Starck, E. J. Candès, and D. L. Donoho, "The curvelet transform for image denoising," *IEEE Transactions on Image Processing*, vol. 11, no. 6, pp. 670-684, 2002.
- [13] M. J. Fadili and J.-L. Starck, "Curvelets and ridgelets," *Encyclopedia of Complexity and System Science*, 2007
- [14] <http://www.curvelet.org/software.html> <retrieved: Sep. 16th, 2011>
- [15] Q. Liu, A. Sung, and M. Qiao, "Neighboring joint density based JPEG steganalysis," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 2:16, 2011.

Concurrent Differential Evolution for Uncertain Optimization Problems

Kiyaharu Tagawa
School of Science and Engineering
Kinki University
Higashi-Osaka 577-8502, Japan
tagawa@info.kindai.ac.jp

Takashi Ishimizu
School of Science and Engineering
Kinki University
Higashi-Osaka 577-8502, Japan
takasi-i@info.kindai.ac.jp

Abstract—Multi-core CPUs, which have more than one processor (core), have been introduced widely into personal computers. Therefore, in order to utilize the additional cores to execute various costly application programs, concurrent implementations of them have been paid attention to. In this paper, a concurrent program of the latest evolutionary algorithm, i.e., differential evolution, is described. Furthermore, the concurrent program of differential evolution, which is called concurrent differential evolution, is revised to reduce the computational time for solving optimization problems in the presence of a wide range of uncertainties. Many real-world applications can be formulated as an uncertain optimization problem in which a probabilistic objective function has to be evaluated by using Monte Carlo integration. Consequently, it usually takes a long computational time to solve the uncertain optimization problem. The results of the numerical experiments conducted on two classes of uncertain optimization problems show that the revised version can reduce the computational time apparently comparing with the conventional version.

Keywords—evolutionary algorithm; concurrent program; uncertain optimization problem; differential evolution.

I. INTRODUCTION

Differential Evolution (DE) is arguably one of the most powerful stochastic real-parameter optimization algorithms in current use [1], [2]. DE can be regarded as an evolutionary algorithm. However, comparing with typical evolutionary algorithms such as genetic algorithm, evolution strategy and particle swarm optimization, it has been reported that DE exhibits an overall excellent performance for a wide range of benchmark problems [3], [4]. Furthermore, because of its simple but powerful searching capability, DE has gotten numerous science and engineering applications [4].

The procedure of evolutionary algorithms for updating the individuals, i.e., tentative solutions, included in the population is called “generation alternation model”. Evolutionary algorithms usually employ either of two types of generation alternation models [5]. The first one is called “generational model”, while the second one is called “steady-state model”. The original DE proposed by R. Storn and K. Price has been based on the generational model [1]. According to the generational model, DE holds two populations, namely the old one and the new one. After generating a new population, a concurrent population is replaced all together by the new population. On the other hand, a new DE based on

the steady-state model has been proposed lately [6], [7]. The new DE is sometimes called “Sequential DE” (SDE) [6]. According to the steady-state model, SDE has only one population. Then each individual in the population is updated one by one. Comparing with the generational model, the steady-state model is usually suitable for parallelized evolutionary algorithms [10]. That is because evolutionary algorithms with the steady-state model need not synchronize the manipulation about any individual in each generation for replacing the old population by the new population.

Recently, multi-core CPUs have been introduced widely into personal computers. In order to utilize the additional cores to execute costly application programs, concurrent implementations of them are demanded [8]. In our previous paper [9], a concurrent program of DE, which is called Concurrent DE (CDE), was proposed. Exactly, the proposed CDE was a parallelized version of the above SDE.

In this paper, CDE is revised for solving uncertain optimization problems by using multi-core CPUs effectively. The revised CDE is called CDE for Uncertain optimization (CDEU). In many real-world optimization problems, a wide range of uncertainties have to be taken into account. Therefore, various evolutionary algorithms including DE for solving uncertain optimization problems have received increasing attention in recent years [12]–[14]. However, the conventional evolutionary algorithms applicable to uncertain optimization problems have not been parallelized for multi-core CPUs. Because the objective functions of uncertain optimization problems are approximated using Monte Carlo integration, the evaluation of them usually takes a very long time. CDEU is a promising optimizer for solving uncertain optimization problems with multi-core CPUs. The performance of CDEU is demonstrated in a comparison with CDE in two classes of uncertain optimization problems, namely noisy and robust optimization problems.

The rest of this paper is organized as follows. Section II describes SDE and a basic strategy of SDE used to generate a new individual. Section III describes CDE. Section IV explains uncertain optimization problems. Section V proposes CDEU for solving uncertain optimization problems. Section VI demonstrates the performance of CDEU in comparison with CDE. Finally, Section VII concludes this paper.

II. SEQUENTIAL DE (SDE)

In this section, we describe a new DE based on the steady-state model, i.e., SDE, and a basic strategy of SDE used to generate a new individual, i.e., a tentative solution.

A. Representation

The optimization problem can be formulated as shown in (1). The optimal solution of the optimization problem is represented by a D -dimensional real-parameter vector $\mathbf{x} = (x_0, \dots, x_{D-1})$ that minimizes the value of the objective function $f(\mathbf{x})$. Besides, each $x_j \in \mathbb{R}$ is limited to the range between the lower \underline{x}_j and the upper \bar{x}_j boundaries.

$$\begin{cases} \text{minimize} & f(\mathbf{x}) = f(x_0, \dots, x_j, \dots, x_{D-1}) \\ \text{subject to} & \underline{x}_j \leq x_j \leq \bar{x}_j, j = 0, \dots, D-1. \end{cases} \quad (1)$$

SDE is used to solve the optimization problem shown in (1). A tentative solution of (1) is represented by a real-parameter vector and called an ‘‘individual’’. SDE holds N_P individuals within the population for each generation. Therefore, the i -th individual $\mathbf{x}_i \in \mathbf{P}$ ($i = 0, \dots, N_P - 1$) arranged in the current population \mathbf{P} is represented as

$$\mathbf{x}_i = (x_{0,i}, \dots, x_{j,i}, \dots, x_{D-1,i}) \quad (2)$$

where, $\underline{x}_j \leq x_{j,i} \leq \bar{x}_j, j = 0, \dots, D-1$.

B. Strategy of SDE

In order to generate a new individual, SDE has borrowed the strategy of DE [6], [7]. In this paper, a basic strategy of DE named ‘‘DE/rand/1/exp’’ is described and used.

For each individual $\mathbf{x}_i \in \mathbf{P}$ ($i = 0, \dots, N_P - 1$), which is also called the target vector, three different individuals, say $\mathbf{x}_{r1}, \mathbf{x}_{r2}$ and \mathbf{x}_{r3} ($i \neq r1 \neq r2 \neq r3$), are selected randomly from the population \mathbf{P} . Then a new individual \mathbf{u} called the trial vector is generated from them as shown in (3). An index $j_r \in [0, D-1]$ is selected randomly. As a result, the trial vector $\mathbf{u} = (u_0, \dots, u_j, \dots, u_{D-1})$ differs from the target vector \mathbf{x}_i at least one element u_{j_r} . Besides, $\text{rand}_j[0, 1]$ denotes the random number generator that returns a uniformly distributed random number from within the range between 0 and 1. Both the scale factor S_F ($0 < S_F \leq 1$) and the crossover rate C_R ($0 \leq C_R \leq 1$) are control parameters specified by the user in advance.

$$\begin{cases} j = j_r; \\ \text{do} \{ \\ \quad u_j = x_{j,r1} + S_F \cdot (x_{j,r2} - x_{j,r3}); \\ \quad j = (j+1) \% D; \\ \} \text{ while } (\text{rand}_j[0, 1] < C_R \wedge j \neq j_r) \\ \text{while } (j \neq j_r) \{ \\ \quad u_j = x_{j,i}; \\ \quad j = (j+1) \% D; \\ \} \end{cases} \quad (3)$$

```

Randomly generate  $\mathbf{x}_i \in \mathbf{P}$  ( $i = 0, \dots, N_P - 1$ );
for ( $i = 0$ ;  $i < N_P$ ;  $i++$ ) Evaluate  $f(\mathbf{x}_i)$ ;
for ( $g = 0$ ;  $g < G_M$ ;  $g++$ ) {
  for ( $i = 0$ ;  $i < N_P$ ;  $i++$ ) {
    Generate  $\mathbf{u}$  from (3) and (4);
    Evaluate  $f(\mathbf{u})$ ;
    if ( $f(\mathbf{u}) \leq f(\mathbf{x}_i)$ ) {
       $\mathbf{x}_i = \mathbf{u}; \quad f(\mathbf{x}_i) = f(\mathbf{u})$ ;
    }
  }
}
Output the best  $\mathbf{x}_i \in \mathbf{P}$ ;
    
```

Figure 1. Pseudocode of SDE

If an element u_j of \mathbf{u} comes out of the range $[\underline{x}_j, \bar{x}_j]$ by using the strategy in (3), it is returned to the range as

$$u_j = (\bar{x}_j - \underline{x}_j) \cdot \text{rand}_j[0, 1] + \underline{x}_j. \quad (4)$$

C. Procedure of SDE

Figure 1 shows the pseudocode of SDE. The stopping condition is specified by the maximum number of generations G_M . Since SDE is based on the steady-state model, only one population \mathbf{P} is used. If a newborn trial vector \mathbf{u} is not worse than the target vector $\mathbf{x}_i \in \mathbf{P}$, the target vector \mathbf{x}_i is replaced by \mathbf{u} immediately. Therefore, the excellent trial vector \mathbf{u} can be used soon to generate offspring.

III. CONCURRENT DE (CDE)

A program is said to be concurrent if it can support two or more tasks in process at the same time. On the other hand, a program is said to be parallel if it can support two or more tasks executing simultaneously [8]. The difference between these definitions is the phrase in progress. The concurrent program performs multiple tasks in parallel if it runs on a multi-core CPU. Therefore, it can be expected that the execution time of an algorithm is reduced by using the concurrent program on the multi-core CPU.

The concurrent program of SDE is named Concurrent DE (CDE) [9]. Figure 2 shows the pseudocode of CDE. CDE employs a static allocation of tasks [8]. Therefore, the population \mathbf{P} is divided equally into N_T chunks. Each chunk can be regarded as a subpopulation including N_P/N_T individuals. Then a task updating the individuals in one chunk is assigned to one thread statically. By using a multi-core CPU, one thread is executed by one core. Consequently, N_T tasks can be parallelized at the most by CDE.

CDE is based on MapReduce framework [11]. First of all, an initial population \mathbf{P} is generated randomly. Then, in

```

Randomly generate  $\mathbf{x}_i \in \mathbf{P}$  ( $i = 0, \dots, N_P - 1$ );
// Map phase
for all  $n$  in parallel do {
    for ( $n = 0; n < N_T; n ++$ ) Thread( $n$ );
}
Barrier();
// Reduce phase
Output the best  $\mathbf{x}_i \in \mathbf{P}$ ;

Thread( $n$ ) {
    for ( $i = 0; i < N_P; i ++$ ) {
        if ( $i \% N_T == n$ ) {
            Evaluate  $f(\mathbf{x}_i)$ ;
        }
    }
    for ( $g = 0; g < G_M; g ++$ ) {
        for ( $i = 0; i < N_P; i ++$ ) {
            if ( $i \% N_T == n$ ) {
                Generate  $\mathbf{u}$  from (3) and (4);
                Evaluate  $f(\mathbf{u})$ ;
                if ( $f(\mathbf{u}) \leq f(\mathbf{x}_i)$ ) {
                     $\mathbf{x}_i = \mathbf{u}; f(\mathbf{x}_i) = f(\mathbf{u})$ ;
                }
            }
        }
    }
}

```

Figure 2. Pseudocode of CDE

Map phase, Thread(n) ($n = 0, \dots, N_T - 1$) are evoked. Each Thread(n) method is assigned to one thread and contracts a task for updating all individuals included in one chunk. Barrier() denotes the method that waits until all Thread(n) methods have been completed. Finally, in Reduce phase, the best individual is selected from \mathbf{P} .

IV. UNCERTAIN OPTIMIZATION PROBLEM

In many real-world optimization problems, a wide range of uncertainties have to be taken into account. Therefore, various evolutionary algorithms for solving uncertain optimization problems have received increasing attention in recent years [12]–[14]. Generally, uncertain optimization problems can be categorized into four classes. First, the objective function is noisy. Second, the decision variables may change after optimization, and the quality of the obtained optimal solutions should be robust against deviations from

```

Thread( $n$ ) {
    for ( $i = 0; i < N_P; i ++$ ) {
        if ( $i \% N_T == n$ ) {
            Evaluate  $F(\mathbf{x}_i, N)$ ;
            Evaluate  $D(\mathbf{x}_i, N)$ ;
        }
    }
    for ( $g = 0; g < G_M; g ++$ ) {
        for ( $i = 0; i < N_P; i ++$ ) {
            if ( $i \% N_T == n$ ) {
                Generate  $\mathbf{u}$  from (3) and (4);
                Evaluate  $f(\mathbf{u})$ ;
                if ( $f(\mathbf{u}) \leq F(\mathbf{x}_i, N) + \alpha \cdot D(\mathbf{x}_i, N)$ ) {
                    Evaluate  $F(\mathbf{u}, N)$ ;
                    if ( $F(\mathbf{u}, N) \leq F(\mathbf{x}_i, N)$ ) {
                         $\mathbf{x}_i = \mathbf{u}; F(\mathbf{x}_i, N) = F(\mathbf{u}, N)$ ;
                        Evaluate  $D(\mathbf{u}, N)$ ;
                         $D(\mathbf{x}_i, N) = D(\mathbf{u}, N)$ ;
                    }
                }
            }
        }
    }
}

```

Figure 3. Pseudocode of CDEU

the optimal point. Third, the objective function is approximated, which means that the objective function suffers from approximation errors. Fourth, the optimum of the problem to be solved changes over time. In this paper, the first and the second classes of optimization problems are discussed.

A. Noisy Optimization Problem

The evaluation of the objective function is subject to noise. Noise in the evaluation of the objective function may come from many different sources such as sensory measurement errors or randomized simulations. Mathematically, a noisy objective function can be described as follows:

$$F(\mathbf{x}) = \int_{-\infty}^{\infty} (f(\mathbf{x}) + \delta) p(\delta) d\delta \quad (5)$$

where, \mathbf{x} is a vector of decision variables and $f(\mathbf{x})$ is a time-invariant objective function considered in (1). δ is additive noise and $p(\delta)$ is the continuous density function of the noise δ . The noise δ is often assumed to be normally distributed with mean 0 and variance 1 such as $\delta = \mathcal{N}(0, 1)$.

In practice, the noisy objective function defined in (5) is approximated using Monte Carlo integration as

$$F(\mathbf{x}, N) = \frac{1}{N} \sum_{t=1}^N (f(\mathbf{x}) + \delta_t) \quad (6)$$

where, N is the number of random samples. The noise is given as follows: $\delta_t = \mathcal{N}(0, 1)$ ($t = 1, \dots, N$).

B. Robust Optimization Problem

The decision variables are subject to perturbations or changes after the optimal solution has been determined. Therefore, a common requirement is that a solution should still work satisfactorily when the decision variables change slightly. Such solutions are termed robust solutions. In order to search for robust solutions, a robust objective function can be described by using a vector of noises δ as follows:

$$F(\mathbf{x}) = \int_{-\infty}^{\infty} f(\mathbf{x} + \delta) p(\delta) d\delta \quad (7)$$

where, $\delta = (\delta_0, \dots, \delta_j, \dots, \delta_D)$, $\delta_j = \mathcal{N}(0, 1)$.

In practice, the robust objective function defined in (7) is also approximated using Monte Carlo integration as

$$F(\mathbf{x}, N) = \frac{1}{N} \sum_{t=1}^N f(\mathbf{x} + \delta_t) \quad (8)$$

where, $\delta_t = (\delta_{0,t}, \dots, \delta_{j,t}, \dots, \delta_{D,t})$, $\delta_{j,t} = \mathcal{N}(0, 1)$.

V. CDE FOR UNCERTAIN OPTIMIZATION (CDEU)

CDE can be applied directly to uncertain optimization problems if the objective function $f(\mathbf{x})$ is replaced by the uncertain one $F(\mathbf{x}, N)$ defined by (6) or (8). However, in order to solve uncertain optimization problems more effectively, we propose a revised CDE. The revised CDE is called CDE for Uncertain optimization (CDEU). The procedure of CDEU is described in Figure 3. Since CDEU differs from CDE in Thread(n) part, only Thread(n) is shown in Figure 3. α is a control parameter. Besides, $D(\mathbf{x}, N)$ denotes a standard deviation of N sampled objective function values, namely $f(\mathbf{x}) + \delta_t$ or $f(\mathbf{x} + \delta_t)$ ($t = 1, \dots, N$).

In order to reduce the evaluation time of the expensive $F(\mathbf{u}, N)$, CDEU prunes away the search for unpromising trial vectors estimated by using the time-invariant function $f(\mathbf{u})$. Even though the values of $f(\mathbf{u})$ and $D(\mathbf{u}, N)$ are required for CDEU, N sampled objective function values, $f(\mathbf{x}) + \delta_t$ or $f(\mathbf{x} + \delta_t)$, evaluated for calculating $F(\mathbf{u}, N)$ can be reused for calculating $D(\mathbf{u}, N)$ again.

VI. EXPERIMENT

Through the numerical experiment conducted on two classes of uncertain optimization problems, the performance of the proposed CDEU is compared with that of CDE.

A. Test Function

For evaluating the performance of CDEU, the following four test functions are used as time-invariant ones. All the test functions have $D = 20$ dimensional real-parameters.

- Sphere function

$$f_1(\mathbf{x}) = \sum_{j=0}^{D-1} x_j^2 \\ - 100 \leq x_j \leq 100, j = 0, \dots, D - 1.$$

- Schwefel's function

$$f_2(\mathbf{x}) = \sum_{j=0}^{D-1} \left(\sum_{k=0}^j x_k \right)^2 \\ - 100 \leq x_j \leq 100, j = 0, \dots, D - 1.$$

- Rastrigin function

$$f_3(\mathbf{x}) = \sum_{j=0}^{D-1} (x_j^2 - 10 \cos(2\pi x_j) + 10) \\ - 5.12 \leq x_j \leq 5.12, j = 0, \dots, D - 1.$$

- Griewank function

$$f_4(\mathbf{x}) = \frac{1}{4000} \sum_{j=0}^{D-1} x_j^2 - \prod_{j=0}^{D-1} \cos\left(\frac{x_j}{\sqrt{j}}\right) + 1 \\ - 600 \leq x_j \leq 600, j = 0, \dots, D - 1.$$

B. Performance Criteria

In order to evaluate the performance of a parallel program for a multi-processor machine, the speedup is often used. The well-known definition of the speedup is the ratio of serial execution time to parallel execution time. However, in the assessment of a concurrent program, the conventional definition of the speedup has to be modified [8]. In the multi-core CPU, the executions of threads, which are evoked from a concurrent program, are assigned to respective cores automatically by operating system. Although the programmer can specify the number of threads in his program, he can not specify the number of cores used by his program.

Furthermore, in order to evaluate the performance of evolutionary algorithms such as CDE and CDEU, a single execution of them is not statistically significant. That is because the evolutionary algorithm is a stochastic algorithm. Therefore, a new speedup is defined as follows:

$$S(N_T, M) = \sum_{m=1}^M \frac{T_m(1)}{T_m(N_T)} \quad (9)$$

where, $T_m(N_T)$ ($m = 1, \dots, M$) is the execution time of CDE or CDEU achieved by using N_T threads. In an ideal situation, speedup is equal to the number of threads.

Table I
NOISY OBJECTIVE FUNCTION VALUE

F_p	F_1	F_2	F_3	F_4
CDEU	-0.242 (0.024)	-0.176 (0.038)	2.244 (1.165)	0.759 (0.034)
CDE	-0.284 (0.024)	-0.194 (0.034)	2.534 (1.439)	0.701 (0.030)

Table II
EXECUTION TIME FOR NOISY OPTIMIZATION PROBLEM

(a) Sphere function: F_1					
N_T	1	2	4	6	8
CDEU	201 (10.8)	105 (8.3)	59 (10.6)	46 (5.0)	36 (7.5)
CDE	1085 (57.4)	565 (35.5)	306 (15.1)	244 (8.8)	189 (5.5)
(b) Schwefel's function: F_2					
N_T	1	2	4	6	8
CDEU	216 (9.1)	117 (7.8)	64 (5.5)	52 (7.4)	43 (6.9)
CDE	1116 (55.5)	579 (36.6)	315 (18.8)	255 (8.9)	193 (10.4)
(c) Rastrigin function: F_3					
N_T	1	2	4	6	8
CDEU	262 (11.6)	136 (10.8)	74 (6.8)	62 (8.4)	49 (5.6)
CDE	1185 (55.0)	614 (31.3)	329 (19.1)	265 (7.0)	208 (10.2)
(d) Griewank function: F_4					
N_T	1	2	4	6	8
CDEU	285 (7.2)	149 (10.4)	79 (4.7)	65 (7.9)	53 (7.6)
CDE	1198 (56.0)	619 (32.5)	333 (19.3)	271 (10.3)	210 (12.6)

C. Experimental Setup

CDEU and CDE were coded using the Java language, i.e., a popular language supporting multiple threads, and executed on a personal computer equipped with a multi-core CPU (CPU: Intel® Core™i7 @3.33[GHz]; OS: Microsoft Windows XP). The multi-core CPU has four cores that can respectively manipulate two threads at one time.

CDEU and CDE were applied respectively to some instances of uncertain optimization problems. During the experiments, the control parameters of them were fixed: the population size $N_P = 96$, the scale factor $S_F = 0.5$ and the crossover rate $C_R = 0.9$. For the stopping condition, the maximum number of generations was fixed to $G_M = 10^3$. Besides, $\alpha = 0.1$ is used for CDEU in all instances.

D. Result in Noisy Optimization Problem

CDEU and CDE were applied to four noisy optimization problems, in which the noisy objective functions $F_p(\mathbf{x}, N)$ were defined by (6) using the above test functions $f_p(\mathbf{x})$. The number of random samples was specified as $N = 100$.

Table III
ROBUST OBJECTIVE FUNCTION VALUE

F_p	F_1	F_2	F_3	F_4
CDEU	18.21 (0.159)	146.6 (6.191)	209.9 (0.856)	0.793 (0.006)
CDE	18.20 (0.172)	145.1 (4.830)	210.5 (0.814)	0.791 (0.004)

Table IV
EXECUTION TIME FOR ROBUST OPTIMIZATION PROBLEM

(a) Sphere function: F_1					
N_T	1	2	4	6	8
CDEU	17857 (127.6)	9016 (125.1)	4601 (132.7)	3908 (71.7)	2963 (75.1)
CDE	19516 (132.8)	9814 (139.6)	5009 (138.6)	4185 (84.1)	3190 (70.5)
(b) Schwefel's function: F_2					
N_T	1	2	4	6	8
CDEU	13550 (289.1)	6870 (155.9)	3535 (127.9)	3195 (90.0)	2339 (82.8)
CDE	22434 (119.9)	11280 (120.6)	5795 (126.0)	4949 (78.0)	3772 (61.3)
(c) Rastrigin function: F_3					
N_T	1	2	4	6	8
CDEU	24426 (198.2)	12295 (161.6)	6340 (133.0)	5612 (81.6)	4328 (48.9)
CDE	30424 (108.6)	15253 (91.7)	7858 (152.1)	6801 (61.6)	5191 (54.1)
(d) Griewank function: F_4					
N_T	1	2	4	6	8
CDEU	24057 (308.2)	12103 (222.9)	6171 (175.4)	5297 (95.6)	3912 (72.2)
CDE	28688 (90.3)	14443 (220.7)	7320 (90.7)	6062 (60.1)	4623 (58.8)

Table I shows the smallest noisy objective function values obtained by CDEU and CDE, which are averaged over $M = 20$ runs. The standard deviations of them also appear in parentheses. Similarly, Table II shows the execution times [ms] of CDEU and CDE using N_T threads. From Table I and Table II, CDEU has reduced the computational time without losing the quality of obtained solutions.

Figure 4 plots the speedups of CDEU achieved with N_T threads in the four noisy optimization problems. Similarly, Figure 5 plots the speedups of CDE. From Figure 4 and Figure 5, the speedups of CDEU and CDE have increased apparently with the increase in the number of threads.

E. Result in Robust Optimization Problem

CDEU and CDE were applied to four robust optimization problems in the same way with the noisy optimization problems. Table III compares the smallest robust objective function values, while Table IV shows the execution times. Figure 6 and Figure 7 also plot the speedups of CDEU and CDE achieved in the four robust optimization problems.

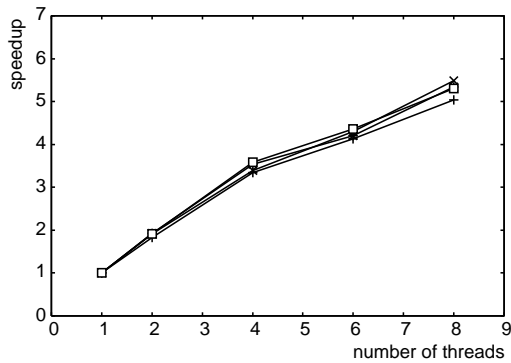


Figure 4. Speedups of CDEU in noisy optimization problems

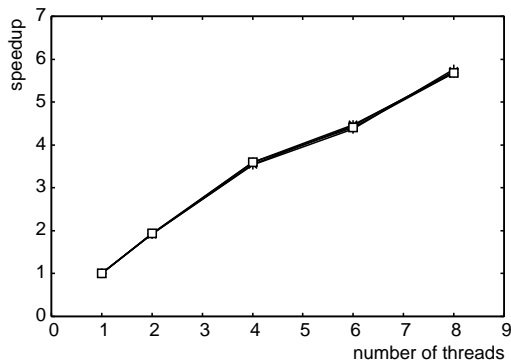


Figure 5. Speedups of CDE in noisy optimization problems

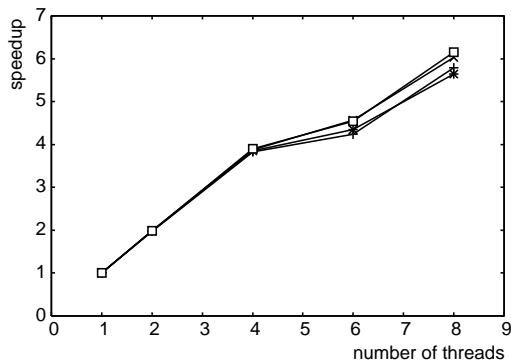


Figure 6. Speedups of CDEU in robust optimization problems

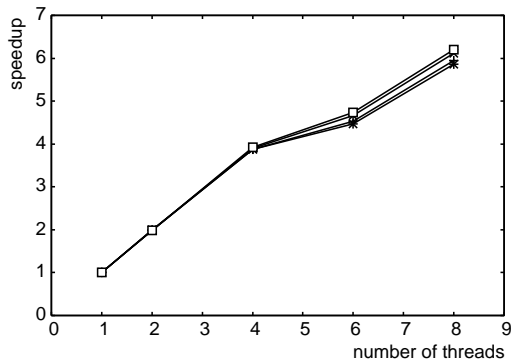


Figure 7. Speedups of CDE in robust optimization problems

VII. CONCLUSION

CDEU was proposed for the noisy and robust optimization problems. From the results of experiments, it was shown that CDEU could reduce the computational time comparing with CDE. Furthermore, CDEU could achieve the high speedup and the quality of obtained solutions as well as CDE.

In our future work, we would like to apply CDEU to real-world applications with various uncertainties.

ACKNOWLEDGMENT

The research was supported in part by the Grant-in-Aid for Scientific Research (C) (No. 21560432) from JSPS.

REFERENCES

- [1] R. Storn and K. Price, "Differential evolution - a simple and efficient heuristic for global optimization over continuous space," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
- [2] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Trans. on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.
- [3] J. Vesterstrom and R. Thomson, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," *Proc. of IEEE Congress on Evolutionary Computation*, pp. 1980–1987, 2004.
- [4] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution - A Practical Approach to Global Optimization*, Springer, 2005.
- [5] G. Syswerda, "A study of reproduction in generational and steady-state genetic algorithms," *Foundations of Genetic Algorithms 2*, Morgan Kaufmann Publ., pp. 94–101, 1991.
- [6] V. Feoktistov, *Differential Evolution in Search Solutions*, Chapter 6, Springer, 2006.
- [7] K. Tagawa, "A statistical study of the differential evolution based on continuous generation model," *Proc. of IEEE Congress on Evolutionary Computation*, pp. 2614–2621, 2009.
- [8] C. Breshears, *The Art of Concurrency - A Thread Monkey's Guide to Writing Parallel Applications*, O'Reilly, 2009.
- [9] K. Tagawa and T. Ishimizu, "Concurrent implementation of differential evolution," *Proc. of WSEAS Int. Conference on New Aspects of System Theory and Scientific Computation*, pp. 65–70, 2010.
- [10] B. D. Davison and K. Rasheed, "Effect of global parallelism on a steady state GA," *Evolutionary Computation and Parallel Processing Workshop, Proc. of Genetic and Evolutionary Computation Conference Workshops*, pp. 167–170, 1999.
- [11] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Proc. of 6th Symposium on Operating Systems Design and Implementation*, pp. 137–149, 2010.
- [12] Y. Jin and J. Branke, "Evolutionary optimization in uncertain environments – a survey," *IEEE Trans. on Evolutionary Computation*, vol. 9, no. 3, pp. 303–317, 2005.
- [13] S. Das, A. Konar, and U. K. Chakraborty, "Improved differential evolution algorithms for handling noisy optimization problems," *Proc. of IEEE Congress on Evolutionary Computation*, vol. 2, pp. 1691–1698, 2005.
- [14] H. G. Beyer and B. Sendhoff, "Evolution strategies for robust optimization," *Proc. of IEEE Congress on Evolutionary Computation*, pp. 1346–1353, 2006.

Energy-aware MPSoC with Space-sharing for Real-time Applications

Stefan Aust, Harald Richter
 Institute of Computer Science
 Clausthal University of Technology
 Clausthal-Zellerfeld, Germany
 stefan.aust|harald.richter@tu-clausthal.de

Abstract—Energy-awareness is an important design criterion for many mobile real-time applications such as phones, handhelds or cars. Normally, high-computing power excludes low electrical power consumption. However, with the two methods of space-sharing and adaptive clocking that are proposed in this paper, both can be reconciled. Space-sharing is an alternative design methodology for embedded systems that outperforms time-sharing with respect to simplicity and power dissipation. Adaptive clocking is an efficient means to further reduce power. The results were achieved by a set of measurements made with a single-chip multiprocessor (MPSoC) that implements space-sharing in a FPGA and by a model MPSoC that implements additionally adaptive clocking.

Keywords—low power; multiprocessor; FPGA; MPSoC

I. INTRODUCTION

A MPSoC is a multiprocessor on a single silicon chip that may contain up to ten or even hundreds of processor-memory modules (PMMs). In case of up to ten PMMs, we speak also of a multi-core processor if the memory modules are lumped together into a single first-level cache that is shared between all cores. The term many core architecture is used for hundreds of cores and more on the same chip that are connected together by appropriate means. Fully custom-designed chips have traditionally implemented MPSoCs, multi- and many-core architectures. During the last few years the capabilities of Field Programmable Gate Arrays (FPGAs) have increased significantly and thus FPGAs allow the implementation of these architectures as an alternative, although with less powerful PMMs today.

Furthermore, there are many real-time applications that demand more and more computing power but have limited energy resources as in mobile devices or controller units in cars, for example. Inefficient usage of the available electrical energy reduces the usability of mobile devices and reduces the operational range of cars and has therefore to be avoided. Energy waste also causes thermal dissipation of heat, and as a consequence requires additional energy for cooling sensitive components. Finally, inefficient usage of energy limits the life expectancy of electronic devices because of aging processes in the semiconductor materials. The pn-junctions in the transistors deteriorate with increasing heat exposure. Because of these facts, energy-awareness in computers systems is an important design criterion. In addition to that, high computing power and low energy consumption normally exclude one another, which is bad for

energy-aware embedded systems that require high performance CPUs.

In this paper, we suggest for energy-critical real-time applications the usage of MPSoCs on a single FPGA. This is possible by applying two methods that are presented by us: space-sharing and individual core clocking on the MPSoC. Space-sharing instead of time-sharing eliminates the problem of finding and guaranteeing a proper schedule of tasks that fulfills a given time constraint. As we will see, it allows also for a better energy management in embedded systems. This means that besides easier handling of complex real-time applications, space-sharing is able to save energy. In addition to that we propose an MPSoC with space-sharing and individual clocking of cores in this paper that optimizes the consumed energy by reducing the dynamic fraction of the chip's power dissipation to a minimum.

The paper is organized as follows: In Section 2, the architecture of the MPSoC is presented, together with the methods of space-sharing and individual core clocking. In Section 3, measurements of the energy consumption of MPSoCs on FPGAs are presented and discussed that do not use space-sharing or individual core clocking. In Section 4, a model MPSoC is investigated that uses both methods. Section 5 draws a conclusion of the performed work and gives an outlook to future work.

II. STATE-OF-THE-ART

Real-time applications demand the execution of a list of tasks within a given time limit or at a given time point. A common method to evaluate the soft or hard real-time capabilities of the underlying embedded system that executes that task list, is the analysis of how long each task list will need to be executed in the worst case (=worst-case execution-time analysis, WCET [1]).

However, the WCET analysis grows exponentially in CPU time with the number of tasks because it is a NP-complete problem. Many task scheduling strategies have been created to solve this by heuristic approaches, such as priority scheduling, earliest deadline first or round robin [2]. Also, many analysis tools as well have been developed and commercialized to simplify the WCET analysis for the designer of embedded systems [3,4].

With space-sharing, there is no need for scheduling tasks and therefore also no need for WCET tools as well. Space-sharing was introduced first by the authors in [5].

A. Space-Sharing

In time-sharing, a single processor is divided in its computing time among multiple tasks. Each task gets one time slice of the processor in a round-robin manner, and after a specific interval all tasks have got some of the computing power of the processor. After that, the whole procedure is repeated periodically. The more the number of tasks and the harder the required time constraints are, the more difficult it is for the processor to get all the tasks done before their deadlines come up. Thus, time-sharing requires that the cumulative CPU needs of all tasks do not exceed the processor's computing power and that a schedule can be found such that all time constraints are met.

In space-sharing, every task is allocated statically to an own processor. No scheduling is needed because tasks are never competing for the same computing resource. This means that the number of processors exactly matches the number of executable tasks (Figure 1).

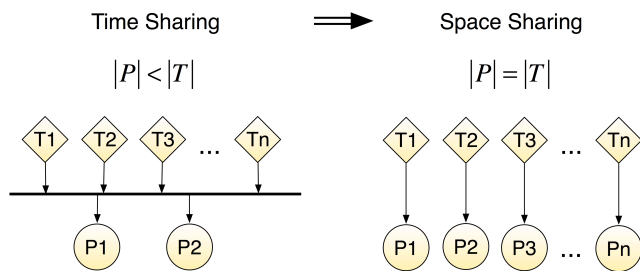


Figure 1. Time-sharing and space-sharing.

Space-sharing requires of course that so many processors are available as tasks are defined by the application programmer and that practical methods exist for allocating tasks to processors and for inter-task communication as well. On the other hand, complex real-time applications can comprise hundreds of tasks and thousands of inter-task communications. However, technological advancements are such that this can be handled by state-of-the art FPGAs. A Virtex-7 FPGA from Xilinx for instance is able to host hundreds of PMMs, provided that they have a low clock frequency compared to full-custom processors and only small on-chip memory. However, for many real-time applications this is sufficient because they have no need for GHz and GBytes in their feed forward and feed back control algorithms. This means that MPSoCs on a FPGA are a convenient way to implement space-sharing in real-time applications.

B. The MPSoC Architecture

Figure 2 shows the MPSoC we have implemented on various FPGAs from Xilinx. The MPSoC consists of n PMMs; where n is configurable, an interconnection network for inter-task communication, an on-chip shared memory and optionally of three hardware ports called bridges to peripheral I/O devices, network interfaces and external memory.

1) *Soft Core Processors*: The processors used in our MPSoC are so-called soft-core CPUs. This means that each processor exists only as a logic in a hardware description

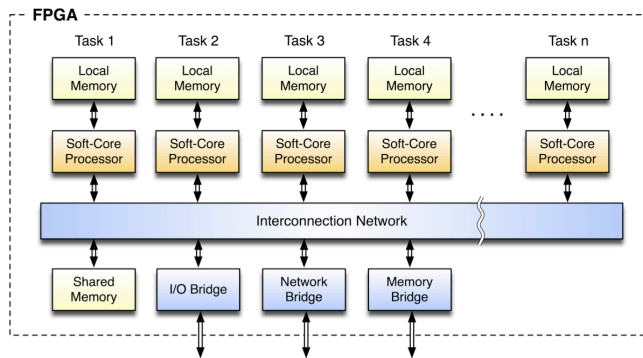


Figure 2. MPSoC architecture on a FPGA.

language, such as VHDL or Verilog that emulates a real processor. We have used a processor description from Xilinx of type MicroBlaze [6]. MicroBlaze emulates an in-order, non-superscalar 32 bit RISC CPU with a clock rate of up to 100 MHz. Because MicroBlaze is a simple RISC architecture it allows one to predict the needed CPU cycles for a given real-time task more easily than in the case of a fully featured CPU. Software development for MicroBlaze can be made in C or C++ because of compilers in the Xilinx EDK. Please note that a virtual processor that is emulated by a small area of the FPGA chip executes every compiled application code.

2) *Local Memory*: The local memories in our MPSoC are split into instruction and data stores according to Harvard architecture. The maximum memory sizes and increments depend on the FPGA type. For the Xilinx XC4VFX100 FPGA for example, local memories can be incremented in steps from 4 KB to 256 KB up to a maximum of 6768 KB for the whole chip.

3) *Non-Blocking Real-Time Interconnection Network*: An on-chip interconnection network establishes communication links between processors, i.e., between task pairs. The links carry messages in a point-to-point manner but it is also possible to implement multicast and broadcast in the 2x2 switches out of which the network is constructed. The network operates asynchronously to the PMM clocks and in circuit switching mode. Asynchronous communication requires a message FIFO at each network input to decouple message creating in a PMM from message transfer in the network. Circuit switching means that a path through the network is established for every point-to-point connection as long as the communication persists.

The most important feature of the interconnection network in our MPSoC is however that it is non-blocking by construction because it consists of multiple stages of 2x2 switches in Benes topology [7]. This non-blocking feature is mandatory for real-time. If the network would block, no upper limit could be guaranteed for the latency of message delivery from one task to another.

4) *Software Development for Space-Sharing*: The software development process for space-sharing is identical to that of time-sharing. The application is implemented in both cases by the code, i.e., the same set of tasks and the same inter-task communication. Only the way of mapping

the tasks and the way how inter-task communication is accomplished is different.

In time-sharing, all tasks are mapped onto time slots and communicate via system calls that a real-time operating system may provide. In space-sharing, all tasks are mapped onto physical areas on a silicon chip, and inter-task communication is implemented by an interconnection network. This is fully transparent to the application because the programmer can use the same API for inter-task communication. This allows for direct portability of code between conventional time-sharing and new space-sharing.

Once the application has been partitioned into tasks, the MPSoC can be built either manually or automatically by configuring in the VHDL or Verilog description of the MPSoC with so many PMMs as needed, plus an interconnection network to couple them. Furthermore, space-sharing allows the adapting of every local memory in its size exactly to the needs of the task. It is also possible to add a coprocessor to a PMM that executes parts of the task as a hardware accelerator. The processor description has to be modified and extended therefore. Finally, space-sharing allows for task isolation and memory protection by construction because each local memory in the MPSoC can be accessed only by its own processor. In space-sharing, inter-task isolation is a feature that requires a memory management unit and at least a basic operation system.

5) *The number of PMMs on the FPGA:* The number of achievable PMMs on the chip varies with the used FPGA type and the amount of external memory that is available on the FPGA board. In order to explore the achievable amount of PMMs per FPGA, we tested various types of FPGAs as listed in table I. On the Virtex-4 FPGA, which provides less than 5 percent of the capability of the recent Virtex-7 XC7V2000T FPGA, we could accomodate a maximum of 34 PMMs including 16 KB memory each.

TABLE I. LIST OF TESTED FPGAS

FPGA	FPGA	
	Chip	Board
Spartan-3	Xilinx XC3S1000	Digilent Starter-Kit
Virtex-4	Xilinx XC4VFX100	PLDA XpressFX
Virtex-5	Xilinx XC5VLX50T	Xilinx ML505

Independent from the used FPGA type, it holds that the summed computing and memory requirements cannot exceed the chip resources. Although only one FPGA and some external memory are involved, the cumulative processing power of all PMMs is nevertheless considerably high if the on-chip parallel processing is effectively organized. In addition to that, task-specific algorithms can be outsourced to a PMM coprocessor where they are executed in hardware, i.e., with maximum speed.

6) *Energy awareness:* The execution of every application needs consumption of electrical power. With space-sharing, only that chip area on a FPGA is occupied that is really needed by the application. It holds that chip areas that are not involved consume much less power than

those whose transistors are switching between low and high states. Furthermore, the sizes of the local memories in space-sharing matches the sizes of the programmed code and data structures because no surplus is generated by VHDL or Verilog chip synthesis tools. As a consequence, power dissipation is as low as the underlying FPGA chip technology allows and as low as the application demands. This is in contrast to time-sharing on a classic embedded system that has to use standard microcontrollers and standard memory chips with given transistor count and therefore power dissipation.

Please note that the energy consumption of our MPSoC is low because of space-sharing but it does not yet reach the absolute minimum. Only if individual PMM clocking is added as a new feature to space-sharing then the key is found for combining real-time and high CPU power with extremely low energy consumption.

III. INDIVIDUAL CLOCKING OF PMMS

Individual clocking of PMMs means that every PMM gets that clock speed that it needs to fulfill the time constraints of the task it executes. Individual clocking saves energy because the faster the processor operates the more dynamic power it consumes.

Space-sharing allows the clocking of every PMM at its best rate and even to vary it dynamically over time, if needed. Individual clocking was implemented by us by means of a clock rate controller in every PMM and by extensions to the application code that can adaptively adjust the clock rate to the various phases a task can have.

A. Clock Rate Controller

Figure 3 shows the set-up of the clock rate controller. It consists of a master oscillator and a clock generator for all PMMs and an individual clock divider for each PMM. The clock divider is set by the processor program and controls thereby the processor clock rate.

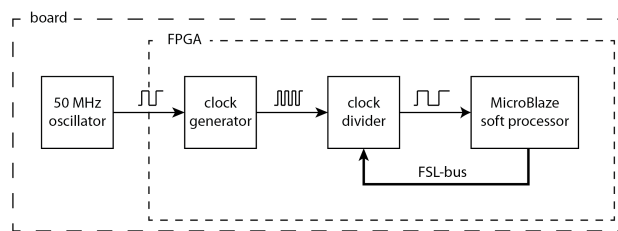


Figure 3. Set-up of the clock rate controller.

To benefit from the clock rate controller, the application programmer of the MPSoC must explicitly set the clock frequency for every task. If the task has several phases with different time constraints then he may set the rate in a fine-grain manner. A first step to accomplish this is to divide each task in clock segments according to Figure 4. After that, he can set the rate for every segment by two methods: either he uses a system call which becomes valid during runtime, or he uses a compiler directive which is evaluated during compile time. Both methods need the introduction of time constraints

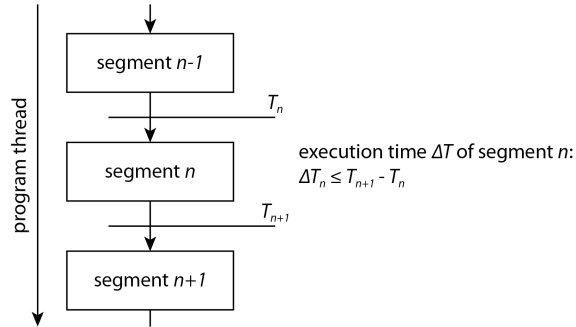


Figure 4. Clock rate adaptation by task segmenting.

that extend the real-time code. A paragon of real-time code extension by time constraints is given in [8].

B. System call for clock rate adaptation

Figure 5 shows an example code where the predefined routine `set_clock_rate` is responsible for the clock rate. The routine `check_new_data` periodically tests a sensor on the arrival of new data. This is accomplished at a slow sampling rate of 2 MHz. If new data is present, then the routine `calc_value` is executed at a rate of 40 MHz, which quickly processes the input. By this method, the processor clock varies dynamically during the program run and thus saves energy. The disadvantage of this method is that the programmer must define clock rates in the program. These clocks depend furthermore on the used processor type because more powerful soft processors can do more jobs at a lower clock rate. This means that the application code has to take into account the concrete MPSoC on which a task is executed and is thus less portable.

```

set_clock_rate(2);           => clock rate = 2 MHz
while(new_data = 0){
    new_data = check_new_data(&data);
};
set_clock_rate(40);         => clock rate = 40 MHz
value = calc_value(&data)
set_clock_rate(2);         => clock rate = 2 MHz
    
```

Figure 5. System call for setting processor clock rate.

C. Compiler directive for clock rate adaptation

The second and better method for clock rate adaptation is to add deadlines to the code segments according to Figure 4. Each deadline T_i defines, until which each segment, i.e., task phase, must be completed. Compiler directives accomplish the definition of deadlines in the code. The directives are formatted as a comment to avoid language extensions or system calls but they are not treated as comments. Instead, a compiler preprocessor reads all meaningful comments and evaluates them. In addition, the compiler knows for what concrete MPSoC it compiles the code, and its preprocessor can therefore count the number of clock cycles needed to execute a task segment in that code. With that information, the preprocessor can set the clock rate MPSoC-dependent, and the programmer does not have to care about that.

```

/*! initiate */
while(new_data = 0){
    new_data = check_new_data(&data);
};
/*! event ≤ initiate + 20 */           => 40 clock cycles / 20μs = 2 MHz
value = calc_value(&data)
/*! terminate ≤ event + 30 */         => 1200 instructions / 30μs = 40 MHz
    
```

Figure 6. Compiler directive for setting processor clock rate.

Figure 6 shows the same example application of Figure 5, but with additional deadlines of 20 and 30 μ s for the segments `event` and `terminate` that have to be met after the `initiate` segment. The resulting preprocessor evaluation is 2 MHz as a clock rate for the `event` and 40 MHz as clock rate for the `terminate` segment, which is the same as in method 1, but that result was achieved more conveniently.

IV. MEASUREMENTS OF POWER CONSUMPTION

We conducted several series of measurements on MPSoCs with and without individual clocking to obtain concrete Figures of the power dissipation of every MPSoC component. All measurements were made on the boards of Section II.

According to [9], total power consumption P of a chip consists of static power P_{stat} and dynamic power P_{dyn} :

$$P = P_{stat} + P_{dyn} \quad (1)$$

Leakage currents inside the semiconductor material cause static power consumption. There are two ways for leakage currents: the sub-threshold leakage, which is an inversion current across the device, and the gate-oxide leakage, which is a tunneling current through oxide insulation of a transistor gate. Both are technology-dependent and cannot be altered by the chip user. Dynamic power in contrast arises from switch activities of transistors. Equation (2) determines the dynamic power consumption [10] as a function of supply voltage V , clock frequency f and chip capacitance C .

$$P_{dynamic} = \sum CV^2 f \quad (2)$$

In the following, measurements are presented for the case of no individual clocking. In Section 5, that technique is added.

A. Static and Dynamic Dissipation

In Figure 7, the static and dynamic dissipation is shown that we have measured for the Xilinx Virtex-4 FPGA. The diagram shows that dynamic power grows nearly linearly with the number of processors. Deviations from a straight line are caused by the place and route function of the used chip synthesis tool, which was XST from Xilinx. All processors have been clocked with 100 MHz first, while the total power consumption of the FPGA was measured. After this, the clock was disconnected to measure static power consumption only. Dynamic power was obtained as the difference between both.

Figure 7 shows that in MPSoCs of 10 PMMs and more the dynamic power dominates in the case of Virtex-4 on the PLDA board. Not shown in Figure 7 but measured by us is that in numbers Spartan-3 needs 268 mW of dynamic dissipation, Virtex-4 needs 120 mW, and Virtex-5 requires 53 mW for every added processor. The low power of Virtex-5 accounts mainly from the 65 nm process, while Spartan-3 and Virtex-4 share the older 90 nm process technology.

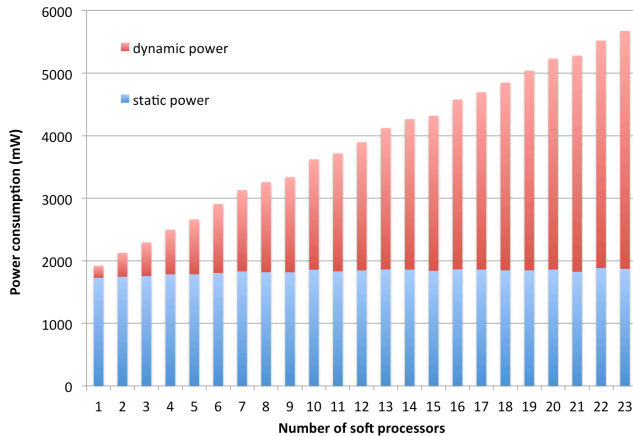


Figure 7. Static and dynamic power consumption versus the number of processors for Virtex-4.

B. Influence of Processor Clock Rates

A second measurement series was conducted to get the dynamic dissipation versus the processor clock rates. The number of processors was parameterized. In Figure 8, the results of several MPSoCs with 1 to 6 processors in a Spartan-3 FPGA are shown. The results were already predicted in theory by equation 2, but concrete values were not known.

C. Influence of Local Memory

Figure 9 shows the influence of local memory for a Virtex-4 FPGA at a clock rate of 100 MHz. The number of processors was again parameterized. As one can see in addition in Figure 9: if a PMM has no memory, about 100 mW remains that is needed by processor logic only.

D. Influence of Interconnection Network

As all PMMs, also the interconnection network has its own clock. In Figure 10, the influence of this clock rate on the dynamic power consumption of the FPGA is illustrated. The diagram shows that dynamic power dissipation increases linearly with a slope of about 0.85 mW per MHz for the tested Virtex-4 FPGA. Furthermore, we found out that the dynamic power consumption of the network mainly depends on the FIFOs at its inputs, not on the switching matrix itself. The FIFO in turn depends on the number of messages that have to be stored, their size and how big the differences between processor clock rates and network clock rates can become. Big differences need deep FIFOs to balance-out message sending and transporting, at least for a while.

E. Power Consumption Parameters

From the conducted measurements, we got the following numbers for parameters of an MPSoC that is implemented on a Virtex-4 for dynamic power consumption:

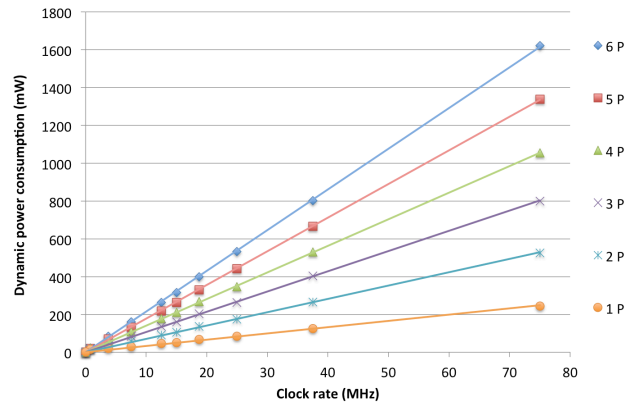


Figure 8. Dynamic power consumption versus processor clock rates for Spartan-3. The number of processors is parameterized.

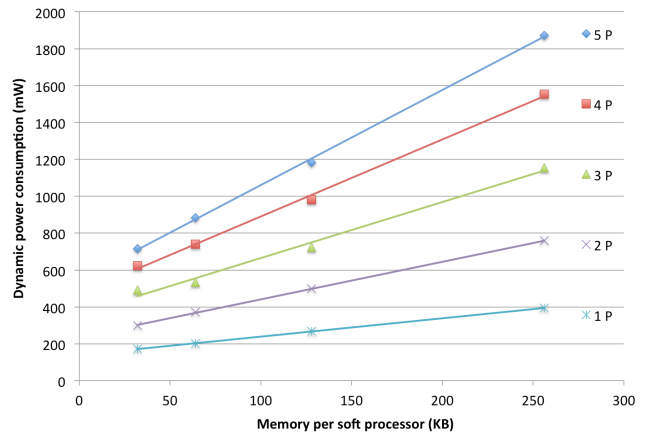


Figure 9. Dynamic power consumption versus size of local memory for Virtex-4. The number of processors is parameterized.

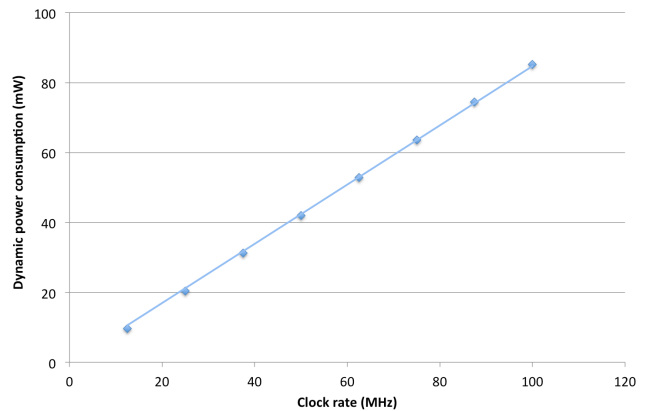


Figure 10. Dynamic power consumption of the interconnection-network versus its clock-rate for Virtex-4.

$$P_{processor} \approx 1 \frac{mW}{MHz} \tag{3}$$

$$P_{memory} \approx 0.01 \frac{mW}{KB \cdot MHz} \tag{4}$$

$$P_{network} \approx 0.85 \frac{mW}{MHz} \tag{5}$$

We used those parameters to define a model MPSoC that comprises 8 PMMs (Figure 11) in order to test the effectiveness of individual clocking.

V. EFFECTIVENESS OF INDIVIDUAL CLOCKING

The defined model MPSoC has individual clock rates and local memory sizes that reflect the requirements of an example application. The accumulative dynamic power consumption of this configuration is 529 mW without phase-adaptive clocking. However, we did not employ that for the model MPSoC in order to have a better comparison to a MPSoC of same PMM number and architecture but with phase-adaptive clocking. It turned out that such a MPSoC would consume 1.9 W with a chip-wide clock rate of 100 MHz and with fixed memory sizes of 64 KB for all PMMs. Out of this, a factor of 3.6 less dissipation is computed. According to that it can be stated that individual clocking of PMMs is an efficient method for energy saving, already without phase-adaptivity. With phase-adaptivity, one can obtain the least possible power dissipation.

A. Set-Up of Model MPSoC

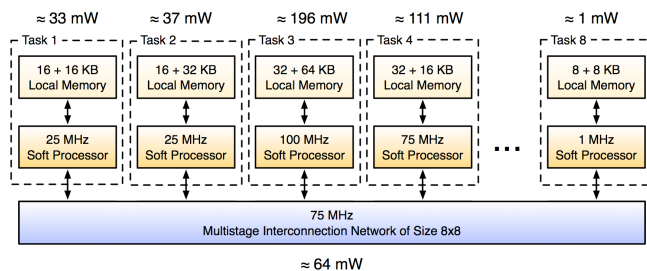


Figure 11. Set-Up of Model MPSoC.

VI. CONCLUSION AND FUTURE WORK

In this paper, the concepts of space-sharing and adaptive clocking have been explained. Space-sharing eliminates the need to find a proper schedule for a list of tasks that have to be executed within a given time interval or at a given time point. It also eliminates the analysis of worst-case execution time of tasks in embedded systems, and the tools created

here for. Additionally, space-sharing allows for a better electrical power management of real-time systems, for simpler task isolation and memory protection and for minimal power dissipation if combined with adaptive clocking. A convenient implementation of space-sharing is accomplished by a single-chip multiprocessor (MPSoC) in which the processor-memory modules are clocked individually due to the task phases and where the sizes of the local memories and the processor performance exactly match the application requirements. Furthermore, we conducted series of measurements to obtain concrete Figures for the power dissipation of the proposed MPSoC on Spartan-3, Virtex-4 and Virtex-5 FPGAs from Xilinx.

Future work will create a tool for the automatic FPGA configuration out of a XML-based description of the task set and inter-task communication of any given real-time application, such that the application can be ported into space-sharing and thus executed by our MPSoC without code modification to obtain minimal chip area consumption and thus very low power dissipation.

REFERENCES

- [1] P. Marwedel, "Embedded System Design," 2nd edition, Dordrecht; Heidelberg, Springer, 2011.
- [2] G. C. Buttazzo, "Hard Real-Time Computing Systems. Predictable Scheduling, Algorithms and Applications," Boston; Dordrecht; London, Kluwer Academic Publishers, 1997.
- [3] Rapita Systems Ltd., www.rapitasystems.com (last checked: 11-06-20).
- [4] Syntavision GmbH, www.syntavision.com (last checked: 11-06-20).
- [5] S. Aust and H. Richter, "Space Division of Processing Power For Feed Forward and Feed Back Control in Complex Production and Packaging Machinery," Proc. World Automation Congress (WAC 2010), Kobe, Japan, Sept. 2010, pp. 1-6.
- [6] Xilinx, "MicroBlaze Processor Reference Guide," October 2009.
- [7] S. Aust and H. Richter, "Real-time Processor Interconnection Network for FPGA-based Multiprocessor System-on-Chip (MPSoC)," The 4th International Conference on Advanced Engineering Computing and Applications in Sciences (ADVCOMP 2010), Florence, Italy, Oct. 2010, pp. 47-52.
- [8] A. Leung, K. V. Palem, and A. Pnueli, "TimeC: A Time Constraint Language for ILP Processor Compilation," Constraints, vol. 7, no. 2, 2002, pp. 75-115, doi: 10.1023/a:1015131814255.
- [9] N. S. Kim, T. Austin, D. Baauw, T. Mudge, K. Flautner, K. S. Hu, M. J. Irwin, M. Kandemir, and V. Narayanan, "Leakage Current: Moore's Law Meets Static Power," IEEE Computer, vol. 36, issue 12, Dec 2003, pp. 68-75.
- [10] L. Shang, A. S. Kaviani, and K. Bathala, "Dynamic Power Consumption in Virtex™-II FPGA Family," Proc. of the 2002 ACM/SIGDA tenth international symposium on Field-programmable gate arrays (FPGA '02), Monterey, CA, Feb. 2002, pp. 157-164.

Image Restoration by Revised Bayesian-Based Iterative Method

Sigeru Omatu, Hideo Araki

Osaka Institute of Tecnology

Asahi-ku, Osaka, 535-8585, Osaka, Japan, Hirakata, Osaka, Japan

omatu@rsh.oit.ac.jp, araki@is.oit.ac.jp

Abstract—A restoration method of the degraded images based on Bayesian-based iterative method is proposed. An iterative method is developed by treating images, point spread functions, and degraded images as probability measures and by applying Bayes’ theorem. The method functions effectively in the presence of noise and is adaptable to computer operation.

Keywords-point spread function, image restoration, Bayesian rule

I. INTRODUCTION

To make a clear image from degraded image, many enhancement techniques have been developed until now. The main technique is to use one of sharpness filters that are based on derivatives of the image with respect to pixels [1]-[2]. These methods used one or more masks to approximate a derivative operation. Although they could enhance the image, they would enlarge also noises.

The present method for enhancement of images is to use the Bayesian rule. This method was first proposed by [3]. The Bayesian rule reflects optimal estimation in a sense to minimize the cost function under noisy observation and an iterative algorithm was proposed to find the optimal solution. The algorithm include two parts, the first one is to estimate a point spread function (PSF) from the estimated image and the second one is to estimate the original image by using the estimated PSF. Thus, this algorithm might be optimal when the observed image is similar to the original image, that is, in case of a high S/N ratio. Therefore, the results will depend on the initial guesses of PSF.

In this paper, we propose a new algorithm to speed up the convergence and find better restoration compared with the results of [3]. The idea is to select the observed image as an initial guess of the restored image and every iteration we use the observation image instead of an estimated image when we estimate the PSF.

First, we will show the principle of the Bayesian-based iterative method proposed by Richardson[3]. Then we will state the proposed method. After that the simulation results will be illustrated to show the effectiveness of the present method.

II. PRINCIPLE OF IMAGE RESTORATION

In image enhancement, the ultimate goal of restoration techniques is to improve a given image in some sense. Restoration is a process that attempts to recover an image that has been

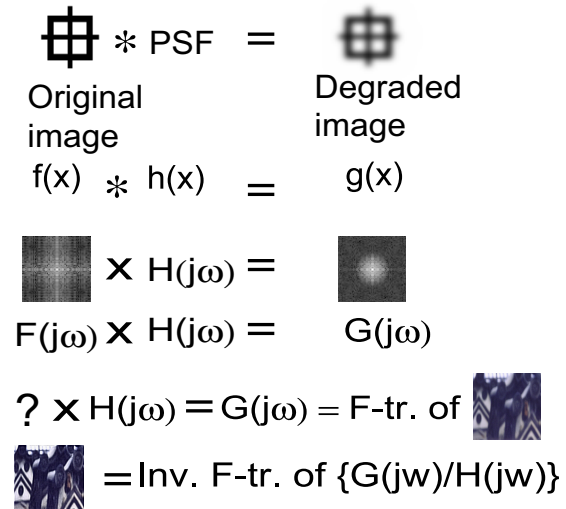


Fig. 1. The restoration principle.

degraded by using a priori knowledge of the degradation phenomenon. As shown in Fig. 1, the degradation process may be modeled as an operator \mathbf{H} in case of noiseless situation. It operates on an input image $f(x, y)$ to produce a degraded image $g(x, y)$. For the sake of simplicity, we denote $f(x, y)$ by $f(x)$, $g(x, y)$ by $g(x)$, $h(x, y)$ by $h(x)$, etc. In equation form, we have

$$g(x) = \mathbf{H}f(x) = h * f(x) = \sum_{y=-\infty}^{\infty} h(x - y)f(y). \quad (1)$$

where $h(x)$ is an impulse response and $*$ denotes the operation of convolution.

Based on the convolution theorem, the frequency domain representation of Eq.(1) becomes

$$G(j\omega) = H(j\omega)F(j\omega) \quad (2)$$

where $G(j\omega)$, $H(j\omega)$, and $F(j\omega)$ are Fourier transforms of $g(x)$, $h(x - y)$, and $f(y)$, respectively. As shown in Fig.1, Given $G(j\omega)$ and some knowledge about $H(j\omega)$, the objective of restoration techniques is to recover $F(j\omega)$ which means to recover the original image $f(x)$ via the inverse Fourier transform.

III. RICHARDSON'S ITERATIVE METHOD

We will review the iterative method by Richardson [3] in this section. Given the degraded image g , the point spread function h and the original image f are estimated based on Bayes' theorem. It will be effective to estimate the original image f from the observed image g . It was assumed that g , h , and f are discrete and are not necessarily normalized. The numerical values of g , h , and f are considered as measures of the frequency of the occurrence of them at those points. h is usually in normalized form. Energy of f originating at a point is distributed as g at points according to the energy indicated by h . Thus, g represents the resulting sums of the energy of f originating at all points.

In the notation of this problem the usual form of the Bayes' theorem is stated as the conditional probability of f , given g . It was assumed that the degraded image g was of the form $g = h * f$, where $*$ denotes the operation of convolution such that

$$g(x) = h * f(x) = \sum_{y=-\infty}^{\infty} h(x-y)f(y). \quad (3)$$

Note that f and g are intensity functions of original image and observed image, respectively and h is the weighting function depending on image measurement devices. We assume that the input image and the weighting function which means the restoration mechanism are unknown. The values of f , g , and h are not limited within $[0,1]$. We normalize and denote them by f' , g' , and h' . Thus, we have

$$f'(x) = \frac{f(x)}{\sum_{x=-\infty}^{\infty} f(x)} = \frac{f(x)}{F} \quad (4)$$

$$g'(x) = \frac{g(x)}{\sum_{x=-\infty}^{\infty} g(x)} = \frac{g(x)}{G} \quad (5)$$

$$h'(x) = \frac{h(x)}{\sum_{x=-\infty}^{\infty} h(x)} = \frac{h(x)}{H} \quad (6)$$

where F, G , and H could be equal since the restoration process is conservative. Note that f , g , and h are nonnegative and the total sums are equal to one. Thus, we could regard them as probability measures and $f'(x_1)$ as the probability measure of the original image $f(x_1)$ at x_1 . This means that the possibility of the existing intensity of the original image $f(x_1)$ at x_1 . Similarly, $g'(x_2)$ and $h'(x_1)$ mean the possibility of the existing intensity of the observed image $g'(x_2)$ at x_2 and the possibility of the transition weight from the input image $f(x_1)$ at x_1 to the output image $g(x_2)$ at x_2 . Therefore, we have

$$P(g(x_2)|f(x_1)) = P(h(x_2 - x_1)) = h'(x_2 - x_1). \quad (7)$$

The above relation can be derived by using the following

relation.

$$\begin{aligned} P(g(x_2), f(x_1)) &= P(g(x_2)|f(x_1))P(f(x_1)) \\ &= P(h * f(x_2), f(x_1)) \\ &= P(h(x_2 - x_1), f(x_1)) \\ &= P(h(x_2 - x_1))P(f(x_1)) \end{aligned}$$

where we have used independence assumption between original image and restoration mechanism.

Using the Bayes' theorem we have

$$\begin{aligned} P(f(x)|g(x_2)) &= \frac{P(g(x_2)|f(x))P(f(x))}{\sum_{x_1=-\infty}^{\infty} P(g(x_2)|f(x_1))P(f(x_1))} \\ &= \frac{f'(x)h'(x_2 - x)}{\sum_{x_1=-\infty}^{\infty} f'(x_1)h'(x_2 - x_1)}. \quad (8) \end{aligned}$$

If we multiply the both sides of Eq.(8) by $P(g(x_2)) = g'(x_2)$ and take the summation with respect to x_2 , we get

$$\begin{aligned} P(f(x)) &= f'(x) \\ &= f'(x) \sum_{x_2=-\infty}^{\infty} \frac{h'(x_2 - x)g'(x_2)}{\sum_{x_1=-\infty}^{\infty} f'(x_1)h'(x_2 - x_1)}. \quad (9) \end{aligned}$$

Considering $F = G = H$ and multiplying them both sides of the above equation, we have

$$f(x) = f(x) \sum_{x_2=-\infty}^{\infty} \frac{h(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} f(x_1)h(x_2 - x_1)}. \quad (10)$$

Using the above equation, Richardson[3] proposed the following recurrence procedure to find the original image $f(x)$.

$$\begin{aligned} f_{n+1}(x) &= f_n(x) \sum_{x_2=-\infty}^{\infty} \frac{h_n(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} h_n(x_2 - x_1)f_n(x_1)} \quad (11) \\ n &= 0, 1, 2, \dots \end{aligned}$$

In order to derive the recursive equation of the PSF function $h(x)$, we will set $x_3 = x_2 - x$. Then from Eq.(8) we have

$$P(f(x_2 - x_3)|g(x_2)) = \frac{f'(x_2 - x_3)h'(x_3)}{\sum_{x_1=-\infty}^{\infty} f'(x_1)h'(x_2 - x_1)}. \quad (12)$$

Multiplying both sides of the above equation by $P(g(x_2)) = g'(x_2)$, we have

$$\begin{aligned} P(f(x_2 - x_3)|g(x_2))P(g(x_2)) &= g'(x_2) \frac{f'(x_2 - x_3)h'(x_3)}{\sum_{x_1=-\infty}^{\infty} f'(x_1)h'(x_2 - x_1)}. \quad (13) \end{aligned}$$

Using the Bayes' rule, we have

$$\begin{aligned} & P(f(x_2 - x_3)|g(x_2))P(g(x_2)) \\ &= P(f(x_2 - x_3), g(x_2)) \\ &= P(g(x_2)|f(x_2 - x_3))P(f(x_2 - x_3)) \\ &= h'(x_3)f'(x_2 - x_3). \end{aligned} \quad (14)$$

From Eqs.(14)and (14), we have

$$\begin{aligned} & h'(x_3)f'(x_2 - x_3) \\ &= g'(x_2) \frac{f'(x_2 - x_3)h'(x_3)}{\sum_{x_1=-\infty}^{\infty} f'(x_1)h'(x_2 - x_1)}. \end{aligned} \quad (15)$$

Taking the summation of both sides of Eq.(15) with respect to x_2 , using the relation of Eqs.(4), (5), and (6), and noting that

$$\sum_{x_2=-\infty}^{\infty} f'(x_2 - x_3) = 1, \quad (16)$$

we have the following relation.

$$h(x) = h(x) \sum_{x_2=-\infty}^{\infty} \frac{f(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} f(x_1)h(x_2 - x_1)}. \quad (17)$$

Thus, using the same recursive relation as Eq.(11), we have

$$h_{m+1}(x) = h_m(x) \sum_{x_2=-\infty}^{\infty} \frac{f_n(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} f_n(x_1)h_m(x_2 - x_1)}. \quad (18)$$

In order to check the convergences of the recursive relations given by Eqs.(11) and (18), the following relations are used.

$$\sum_{x_2=-\infty}^{\infty} \frac{h(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} h(x_2 - x_1)f_n(x_1)} = 1, \quad (19)$$

$$\sum_{x_2=-\infty}^{\infty} \frac{f(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} f(x_1)h(x_2 - x_1)} = 1. \quad (20)$$

(21)

Thus, we use the following criteria to stop the iterations.

$$1 - \epsilon < \sum_{x_2=-\infty}^{\infty} \frac{h_m(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} h_m(x_2 - x_1)f_n(x_1)} < 1 + \epsilon, \quad (22)$$

$$1 - \epsilon < \sum_{x_2=-\infty}^{\infty} \frac{f_n(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} f_n(x_1)h_m(x_2 - x_1)} < 1 + \epsilon. \quad (23)$$

Using the above relations, Richardson has proposed the following iterative algorithm(Richardson's Iterative Method).

Step 1. Set $n = 0, m = 0$, the initial guesses of $h_0(x)$, and $f_0(x)$, and small positive number ϵ .

Step 2. Solve the following equations:

$$f_{n+1}(x) = f_n(x) \sum_{x_2=-\infty}^{\infty} \frac{h_m(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} h_m(x_2 - x_1)f_n(x_1)} \quad (24)$$

$$h_{m+1}(x) = h_m(x) \sum_{x_2=-\infty}^{\infty} \frac{f_n(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} f_n(x_1)h_m(x_2 - x_1)}. \quad (25)$$

Step 3. If the following inequalities hold

$$\left| \sum_{x_2=-\infty}^{\infty} \frac{h_m(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} h_m(x_2 - x_1)f_n(x_1)} \right| < 1 - \epsilon \quad (26)$$

and

$$\left| \sum_{x_2=-\infty}^{\infty} \frac{f_n(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} f_n(x_1)h_m(x_2 - x_1)} \right| < 1 - \epsilon, \quad (27)$$

then stop, otherwise $n \leftarrow n + 1, m \leftarrow m + 1$ go to Step 2.

The above iteration has no proof of convergence that means the results obtained by the above iteration may result in the good results or may not.

IV. PROPOSED ALGORITHM

In order to get the better results compared with Richardson's algorithm, we consider a new method based on the property of degraded images such that the blurred images are similar to the original images. In the Richardson's algorithm, if the bad estimation of $h_m(x)$ at the beginning stage, corresponding recovered images would become different images. After obtaining the bad estimation of recovered images, worse estimation of the point spread function. As a result, the iteration will produce worse and worse estimation of the point spread function and recovered images. Assuming the degraded images are not so far from the original images, we use the blurred image to estimate the point spread function $h_m(x)$ instead of the recovered image that is the estimated image. Therefore, we have proposed the following algorithm:

Step 1. Set $n = 0, m = 0$, small positive number ϵ , and $f_0(x) = g(x)$. Set the initial guesses of $h_0(x)$.

Step 2. Solve the following equations:

$$f_{n+1}(x) = f_n(x) \sum_{x_2=-\infty}^{\infty} \frac{h_m(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} h_m(x_2 - x_1)f_n(x_1)} \quad (28)$$

$$h_{m+1}(x) = h_m(x) \sum_{x_2=-\infty}^{\infty} \frac{g(x_2 - x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} g(x_1)h_m(x_2 - x_1)}. \quad (29)$$

Step 3. If the following inequalities hold

$$\left| \sum_{x_2=-\infty}^{\infty} \frac{h_m(x_2-x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} h_m(x_2-x_1)f_n(x_1)} \right| < 1 - \epsilon \quad (30)$$

and

$$\left| \sum_{x_2=-\infty}^{\infty} \frac{f_n(x_2-x)g(x_2)}{\sum_{x_1=-\infty}^{\infty} f_n(x_1)h_m(x_2-x_1)} \right| < 1 - \epsilon, \quad (31)$$

then stop, otherwise $n \leftarrow n + 1$ and go to Step 2.

V. VARIATION USING THE REVERSE FUNCTION

We consider more simple form of the proposed algorithm. We set the dominator of Eq.(11) by

$$L_{nm}(x_2) = \sum_{x_1=-\infty}^{\infty} h_m(x_2-x_1)f_n(x_1). \quad (32)$$

It is the convolution sum between the original image $f_n(x_1)$ and the point spread function $h_m(x_2-x_1)$. Therefore, if $L_{nm}(x_2) = g(x_2)$, then the estimated image $f_n(x_1)$ becomes the true original image. Furthermore, we have

$$f_{n+1}(x) = f_n(x) \sum_{x_2=-\infty}^{\infty} \frac{h_m(x_2-x)g(x_2)}{L_{nm}(x_2)}. \quad (33)$$

We define $r_{nm}(x_2)$ by

$$r_{nm}(x_2) = \frac{g(x_2)}{L_{nm}(x_2)} \quad (34)$$

which means the ratio between the observed degraded image and the degraded image obtained by using the estimated point spread function. Then Eq.(33) becomes

$$f_{n+1}(x) = f_n(x) \sum_{x_2=-\infty}^{\infty} h_m(x_2-x)r_{nm}(x_2). \quad (35)$$

We define the reverse function of $k(x)$ by $k(x) = h(-x)$. Then Eq.(??) becomes

$$f_{n+1}(x) = f_n(x) \sum_{x_2=-\infty}^{\infty} \frac{k_m(x-x_2)r_{nm}(x_2)}{\cdot} \quad (36)$$

If we represent the convolution sum by Fourier transform, we have

$$\begin{aligned} f_{n+1}(x) &= f_n(x)FT^{-1}(FT(k_n(x-x_2))FT(r_{nm}(x_2))) \\ &= f_n(x)FT^{-1}(K_n(j\omega)R_{nm}(j\omega)) \end{aligned} \quad (37)$$

where FT and FT^{-1} denote the Fourier transform and inverse Fourier transform. Since $K_m(j\omega) = H_m(-j\omega)$, we could save the computational time by half.



(a) Original image



(b) Degraded image



(c) Richardson method



(d) Proposed method

Fig. 2. The comparison for Example 1.

VI. SIMULATION RESULTS

In order to show the effectiveness of the proposed method, we will consider gray image (Example 1) and color image(Example 2). The computer specification used here is shown in TABLE I. In Example 1 the gray image of 64×64 was made using Photoshop. The color image of 512×512 of Example 2 was cropped from the standard sample data of high-resolution color images [4]. The degraded images are made by using Gaussian filters with the standard deviation $\sigma = 2.0$. We used the stopping parameters of m and n when maximum iteration number k is given. In these simulations, we changed those parameters (m, n, k) in three cases, that is, (10,100,10), (5,100,10), and (5,5,100). TABLE II shows simulation results for three cases with PSNR (Peak Signal-to-Noise Ratio). In Fig. 2 shows the simulation results of the gray image with (10,100,10). From this Fig. 2 the proposed method restored more clear image compared with the results by Richardson's method [3]. In Figs.3-6 we show the simulation results of Example 2. The original image is shown in Fig. 3 and the degraded image is shown in Fig. 4. The restored images by [3] and the proposed method are shown in Fig. 5and Fig. 6.

TABLE I
COMPUTING ENVIRONMENT

OS	Windows XP
CPU	AMD Athlon(tm)64 X2 Dual Core
Memory	2GB

TABLE II
PSNR BETWEEN ORIGINAL IMAGE AND RESTORED IMAGE

Threshold value(m,n,k)	Degraded image	Richardson	Authors
(10,100,10)	14.5	15.7	16.6
(5,100,10)	14.5	16.5	16.0
(5,5,100)	14.5	9.2	16.2



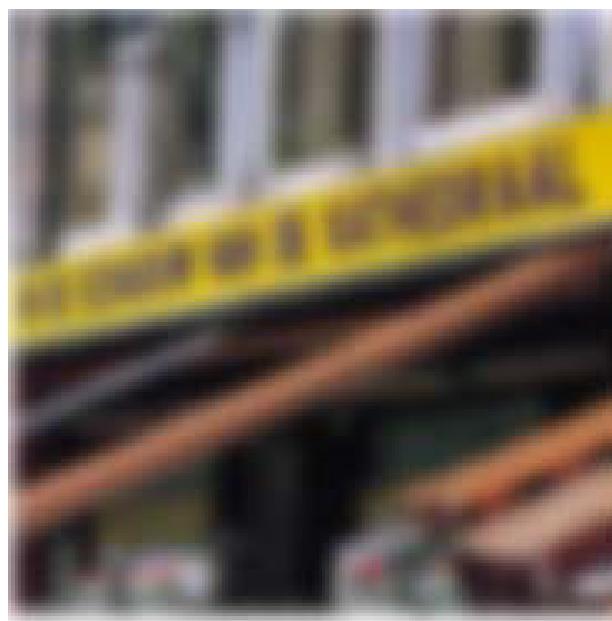
Original image

Fig. 3. Original image for Example 2.



Richardson's method

Fig. 5. Richardson's method for Example 2.



Degraded image

Fig. 4. Degraded image for Example 2.



Proposed method

Fig. 6. Proposed method for Example 2.

VII. CONCLUSIONS

In this paper, a method of restoration of the degraded images by using Bayesian-based iterative method. The simulation results showed that the proposed method could restore the degraded images more clearly compared with the Richardson's method while the threshold values of (n, m, k) must be determined by trial and error. Furthermore, the computation load has been decreased by half by introducing the ratio between the observed degraded image and the degraded image.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant-in-Aid for Scientific Research (B) (23360175). The authors would like to thank JSPS to support this research work.

REFERENCES

- [1] T. Young and K. Fu, Handbook of Pattern Recognition and Image Processing Independent Component, Academic Press, New York, (1986)
- [2] R. Duda, P. Hart, and D. Stork, Pattern Classification, John Wiley & Sons, New York, (2001)
- [3] W. Richardson, Bayesian-Based Iterative Method of Image Restoration, Journal of Optical Society of America, Vol. 62, pp.55-59, (1972)
- [4] Standard Color Digital Images of High-Resolution (CMYK/SCID), JIS X 9201:2001, (2001)

Automatic Error Detection in Gaussian Processes Regression Modeling for Production Scheduling

Bernd Scholz-Reiter, Jens Heger

BIBA - Bremer Institut für Produktion und Logistik GmbH

at the University of Bremen

Hochschulring 20, 28359 Bremen, Germany

e-mail: {bsr|heg}@biba.uni-bremen.de

Abstract— In the application field of production, scheduling with dispatching rules is facing the problem that no rule performs globally better than any other. Therefore, machine learning techniques can be used to calculate estimates of rule performances and select the best rule for each system state. A number of estimates are of poor quality and lead to a wrong selection of rules. Motivated by this problem, to further stabilize the selection approach a general approach, to automatically detect ‘faulty’ estimates from regression models is introduced and analyzed in this paper. Therefore, different models are learned and if their estimates differ strongly, it is likely that at least one model delivers poor estimates. Additionally, a difference-threshold for our example data is defined. As a machine learning technique, we use Gaussian process regression with different covariance functions (kernels). The results have shown that our automatic detection works in most cases and poorly tuned models can be detected.

Keywords-Gaussian processes; dispatching rules; machine learning; scheduling; multiple classifier techniques.

I. INTRODUCTION

Alpaydin described machine learning in the following way: “The goal of machine learning is to program computers to use example data or experience to solve a given problem” [8]. In regression, this means that the given data is analyzed and used to calculate a regression function, which can be used to get estimates for unknown data points. In Fig. 1, an example is depicted, where the regression function gives an estimate for point c .

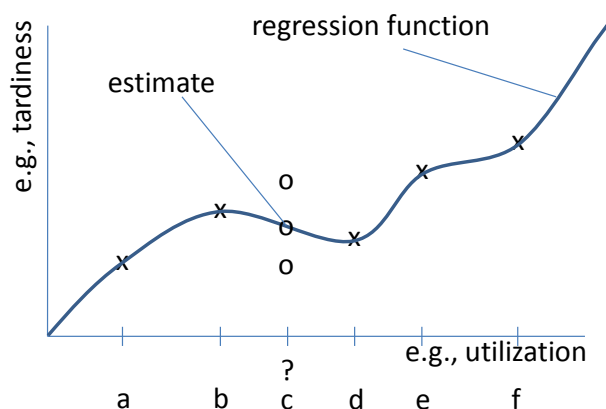


Figure 1: An example of regression function

Our general motivation is the optimization of a production system, e.g., reducing tardiness of jobs. Decentralized scheduling with dispatching rules is applied in many fields of logistics and production, which are characterized by high complexity and dynamics. Instead of calculating a global plan (a schedule for all jobs and machines), dispatching rules work in an autonomous, decentralized way. Dispatching rules, which are a special kind of priority rules, assign a job to a machine. Each time, the machine has finished a job and more jobs are waiting, the next job to be processed is selected by calculating a priority for each of the waiting jobs. This priority can be based on attributes of the job, the machines or the system. The job with the highest priority is chosen to be processed next. Dispatching rules have been developed and analyzed in the scientific literature for many years; see e.g., [1], [2] and [3]. The most well known rules are *Shortest Processing Time first* (SPT), *Earliest Due Date* (EDD) and *First in System First Out* (FSFO). Many dispatching rules perform well on different scenarios, but no rule has been found, which outperforms other rules across various objectives. For this reason, approaches to switch between rules depending on the current system conditions have been proposed. Most of these approaches use learning techniques (e.g., neural networks) to estimate the performance of each dispatching rule and select the best [4].

To calculate the performance of rules, we perform simulation runs of the production system with several dispatching rules and different system settings, e.g. different utilization levels. Since lots of possible variants exist and simulation runs are time consuming, we want to perform as few simulation runs as possible and use machine learning to estimate performances (e.g., tardiness) of not explicitly simulated settings. The results of simulation running of a production system are the learning data for models we learn with Gaussian Process Regression. Based on these models, the best dispatching rule for the current scenario is selected [5]. This procedure avoids costly and unnecessary simulation runs. It is practically not possible to run preliminary simulation runs for all parameter combinations.

In most cases, this approach works successfully, but in some cases the learning fails and results in improper

regression curves. This leads to a wrong selection of dispatching rules and thus to a poor performance of the production system. Our goal is to automatically detect, if the learned model might be ‘faulty’. In this case, the learning data is supplemented using additional simulation runs.

In this paper, we concentrate on the problem of improper regression curves and their automatic detection. This is a general problem, which not only occurs in dispatching rule selection.

In this study, we use synthetic data instead of simulation runs. On the one hand these simulation runs are time consuming and on the other hand we need many different functions to test our error recognition. Therefore, we are using synthetic functions, which are similar to our simulation runs and concentrate on the error detection.

Gaussian processes (GP) is one promising machine learning technique [5]. --> These techniques have been introduced in 1996 and were promoted in the machine learning community by Williams et al. [6]. Analyses reveal their good prediction performance in comparison with other techniques [7]. As a further advantage, their formalism allows providing a measure of prediction quality with each predicted value in a natural way. Additionally, they are – their mathematical background aside – relatively easy to handle.

The obvious problem with machine learning is that learned models can only estimate values. Sometimes, their estimates differ strongly from the original value. These cases lead to wrong decisions. To provide more stability and to automatically detect wrong estimates, we suggest combining similar models. If one model strongly differs from the comparison data, further learning steps, such as adding more data, are necessary.

This paper is organized as follows: in Section 2, we introduce Gaussian processes and general problems with machine learning techniques. Section 3 comprises our approach and the settings of the performed experiments followed by the results in Section 4. The paper concludes with a short summary and provides directions towards future research.

II. PROBLEM DESCRIPTION AND STATE OF THE ART

Regression models are used to provide estimates of values, which are not exactly known, because it is too costly to calculate them or just not possible. In our application field, we experienced that a number of tuned models provided poor estimates, which is a general problem in regression. Therefore, in this paper, we determine how it is possible to automatically detect such bad models.

A. Gaussian Processes

1) Introduction

O’Hagan [9] represents an early reference from the statistics community for the use of a GP as a prior over functions, an idea which was only introduced to the machine learning community by Williams et al. [6].

We have a simulation model implicitly implementing a (noisy) mapping between a vector of state variable (in our case containing, e.g., utilization) and the objective function (mean tardiness) $y = f(x) + \epsilon$. The learning consists of finding a good approximation $f^*(x)$ of $f(x)$ to make predictions at new points x .

To tune such a model using GP requires some learning data as well as a so-called covariance function. This covariance function, sometimes called kernel, specifies the covariance between pairs of random variables and influences the possible form of the learned function f^* .

Since we want to check and compare the tuned models, we use three different kernels, which are well suited for our application. These are the squared exponential (SE) covariance function (1), which is a common choice in Gaussian Process Regression. Additionally, we use 2 functions of the *Matérn* class (2), with parameter $d=3$ and $d=5$ (see [6], chapter 4), which are a good choice in many engineering applications.

$$k_y(x_p, x_q) = \sigma_f^2 \exp\left(-\frac{1}{2l^2}(x_p - x_q)^2\right) + \sigma_n^2 \delta_{pq} \quad (1)$$

$$\text{cov}(f(x_p), f(x_q)) = k(x_p, x_q) = \sigma_f^2 f\left(\frac{\sqrt{d}}{l}r\right) \exp(-\sqrt{d}r) \quad (2)$$

with $f(t)=1+t$ for $d=3$

$f(t)=1+t+t^2/3$ for $d=5$.

The formulas further include the so-called hyperparameters. These parameters of a covariance function can be used to fine-tune the GP-model. The *squared exponential* covariance function used in our experiments has three hyperparameters. There is the length-scale l , the signal variance σ_f^2 and the noise variance σ_n^2 . The *Matérn* functions have the signal variance σ_f^2 and factor l as well. Additionally, since hyperparameters can be interpreted as length-scale parameters in the case of the *squared exponential* covariance function, further optimization is possible. Rasmussen and Williams [10] describe the hyperparameters informally like this: “how far do you need to move (along a particular axis) in input space for the function values to become uncorrelated”. Thus, the *squared exponential* covariance function implements an automatic relevance determination (ARD) [11], since the inverse of the length-scale determines how relevant an input is. A very large length-scale value means that the covariance will become almost independent of that input. ARD has been used successfully for removing irrelevant input by several authors, e.g., Williams et al. [6].

Gaussian processes provide a quality estimate of their predicted value, exemplarily denoted by the shaded area in Fig. 2. Fifteen noisy training points are given and since

there is noise, the standard deviation close to the training points is small, but not exactly zero. In between two points as well as at the beginning and the end, the quality of the estimates decreases.

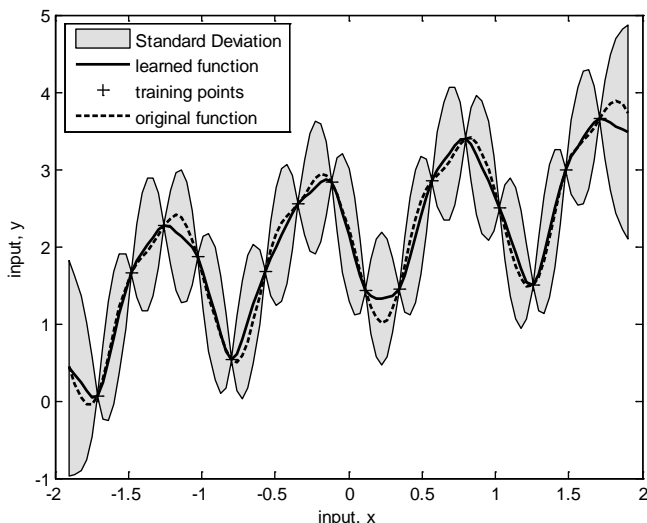


Figure 2. Example of a

Gaussian process regression function with noisy training points observed. The mean prediction is shown as a black line and the shaded area denotes twice the standard deviation. The full underlying original function is also shown.

B. Application and Example

Learning with Gaussian processes is done by selecting a covariance function and setting its free hyperparameters. To learn or optimize the hyperparameters, the marginal likelihood should be maximized [10]. Further, the setting of the hyperparameters aims to a minimization of the generalization error, which denotes the average error on unseen test examples. This is done with cross-validation by splitting the training data in learning and test data. An optimization of the training error does not take place, this may lead to an over-fitting of the data (see below).

C. Machine learning – model quality

A typical task in data mining is learning a model that bases on available data. These models can be regression models or classifiers. The problem evaluating such a model is that it may have an adequate prediction capability, but might fail to predict future unseen data [13]. This problem is called overfitting, because the model fits well on the training data, but the general quality might be poor. To estimate the generalization performance in this context, a procedure called cross-validation is recommended. The idea of cross-validation originates in the 1930s and has been further developed by Mosteller and Wallace and others in the 1960s [12].

1) Cross-Validation

Cross-Validation is a statistical method for the evaluation of learning algorithms by dividing data into two parts. One is used to learn a model, the other used to validate it. The basic form of cross-validation is the k -fold cross-validation. In k -fold cross-validation, the data is first partitioned into k equally sized folds. Subsequently, k iterations of training and validation are performed. Within each iteration, a different fold of the data is held-out for validation, while the remaining $k-1$ folds are used for learning [13].

A special case of k -fold cross-validation is ‘leave-one-out’ cross-validation. In this case, k is set to the number of instances in the data. This means, that during each iteration, all data points are used for learning except one, which is used for testing. Leave-one-out cross-validation is used especially, when the available data are very rare.

2) Bootstrapping

Another method for assigning measures of accuracy to sample estimates is *bootstrapping*, which was introduced by Efron and Tibshirani [16]. Bootstrapping is a method that uses resampling to create sets of data derived from one original data set. The bootstrap process can be described in the following steps: b bootstrap samples are generated from the original data set. Each of these samples has n elements, which were generated by sampling with replacement n times. By calculating the value of the estimator of the replicates the bootstrap replicates can be obtained. The variance of the estimates can be determined by computing the variance of the estimates for the samples. The assumptions gained from bootstrapping are similar to those gained from cross-validation, i.e., stability of the algorithm on the dataset, which should closely approximate the real world [14]. More details and a comparison of cross-validation and bootstrapping was conducted by Kohavi [14].

3) Problem description

In Fig. 3, two different kernel functions are used to learn a model with 15 training points. Cross-validation is used to optimize the hyperparameters. The learned function 2 provides estimates close to the original function. Learned function 1 is not that close and has the form of a linear average function. Errors or problems like this occur regularly and deteriorate the estimates and the decisions based on them.

Cross-validation and bootstrapping are good accuracy estimation methods and are applicable for parameter setting or performance estimation. Still, it is hard to detect with these methods, if the learning model has obviously failed, like function 1 in Fig. 2 did. These cases are responsible for a high amount of the total error of the learning model. As a solution for this problem, this paper presents the approach of combining different models for automatic error recognition.

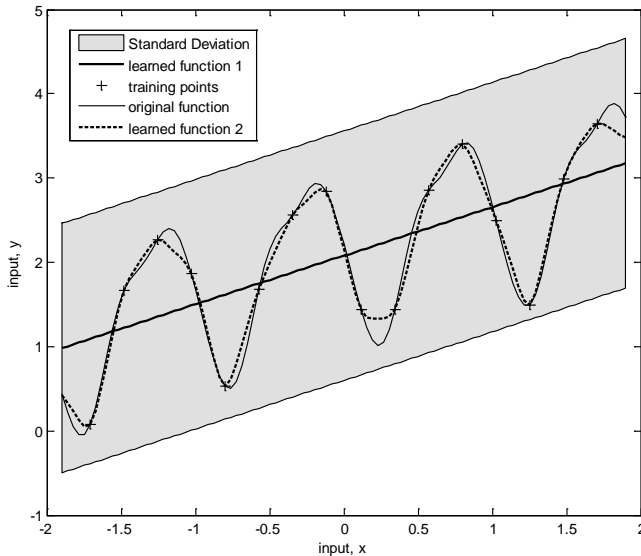


Figure 3: An example for two learned regression curves, where the learned function 1 one provided bad estimate and learned function 2 provides good estimates close to the original function

III. APPROACH AND EXPERIMENTS

The presented approach aims to stabilize the learning process by detecting ‘faulty’ regression curves automatically. Therefore, we tune different models with different kernels, e.g. covariance functions. If there are great differences between these models, it is obvious, that at least one model does not fit to the learning data. In these cases, more learning data can help to improve the models.

Generally, adding more learning data can also reduce errors. But this contradicts the objective of reducing costly simulation runs by using machine learning.

A. Experimental setup

For our experiments we use the framework provided by Rasmussen and Nickisch [10], [15]. Three covariance functions (*squared exponential* and *Matérn 3 & 5*) are implemented as learning kernels. The learning data is synthetically generated by a neural network based covariance function in combination with a periodic component (see framework documentation [15], covariance functions ‘NNone’ and ‘Periodic’). Hyperparameters are set to $l=1$; $sf=1$ (NNone) and $l=1/12$, $p=1$, $sf=1$ (Periodic) and noise variance = 1. This way, we get smooth functions, which are similar to the curves resulting from simulation runs with dispatching rules scheduling. But since we need a lot of functions to get a general result, we use these synthetic functions instead of simulations runs.

B. Experiments

For the analysis, 1000 random functions are generated as described before, with a discretization level of 101 on the x-level. We have used 13 learning data points and the hyperparameters are optimized using cross-validation.

When using Gaussian processes regression, a covariance function needs to be selected. The most common choice is the *squared exponential*, but depending on the application other covariance functions or their compositions might fit better to the application data. In this paper two different approaches are analyzed: First, one covariance function is set as the default learning kernel and others are used to check the results for errors.

If no preferable covariance function is known in advance a more general approach is analyzed. In this case two or more covariance function are used to learn a model and each of their predictions are used to calculate a mean resulting prediction.

1) One main covariance function is selected

In the first set of experiments the *squared exponential* covariance function is set as the main learning model. To detect the cases, the considered model is of less quality, the *Matérn* functions with parameter $d=5$ ($d=3$ leads to very similar results) comes into operation. The learning error is determined by calculating the difference between the *squared exponential* and the original function. The differences between *Matérn* and *squared exponential* are also calculated. If these values correlate, a threshold level for the difference between the *squared exponential* and the *Matérn* functions indicates a higher error in the learning model.

2) Predictions based on multiple covariance functions

In the second set of experiments the *squared exponential* is not set as the standard model. Instead, both models are used equally. That means, the learning error in these experiments is the difference between the original function and the mean prediction of both models. The difference between both models is also calculated. If these data correlates, errors in the examined models can be detected, without knowing, which model works better to the application data in advance.

IV. RESULTS

Fig. 4 depicts the results of the differences between two learning models (*squared exponential* to *Matérn 5*) and the error to the original function (*squared exponential* to *original data*). Some correlating high errors for both are highlighted. Fig. 6 does the same for (mean (*squared exponential-Matérn 3*) to *original data*).

The first set of experiments show that the difference between the two models correlates to the error. There are many data pairs in Fig. 4 showing this for high values exemplarily. The higher the difference in the two models, the higher the error of the *squared exponential* learning model. Fig. 5 shows that there is a linear dependency between both values. Many data points are close to 0, which means, that most times the learning worked well and the difference between the models is small. This makes it easy to find different threshold levels to divide the functions into

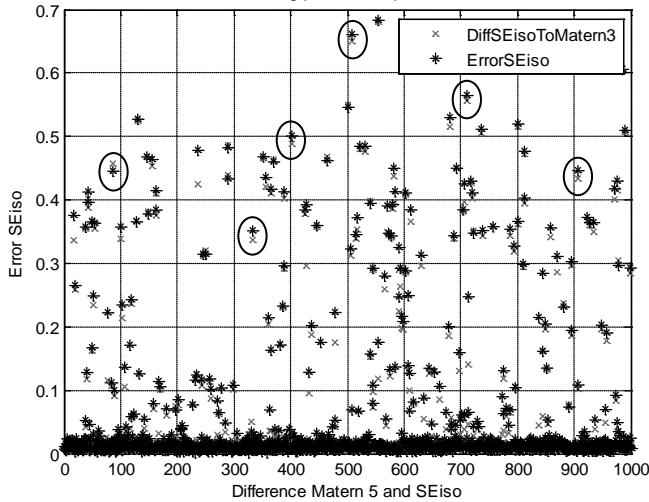


Figure 4: 1000 synthetic functions learned by the squared exponential and the Matérn 5 function. The error between the original function and the squared exponential function are depict as well as the difference between squared exponential and Matérn.

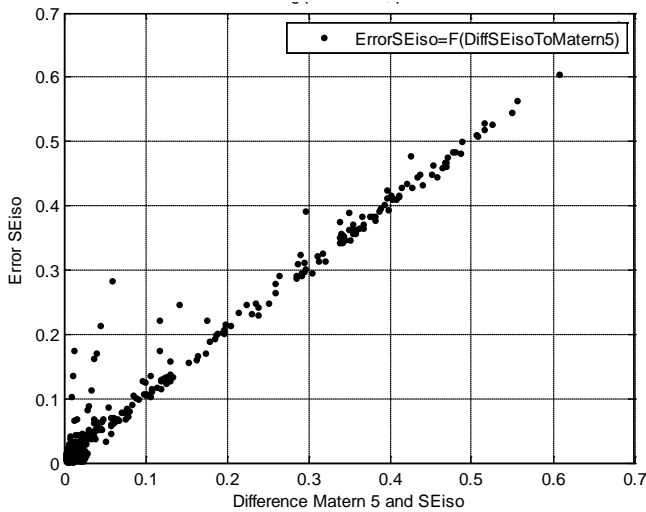


Figure 5: Correlation between differences in the two learned models (squared exponential and Matérn 5) and the error between the squared exponential function and the original data

TABLE I. THRESHOLD AND RESULTS DISTRIBUTION

threshold	number of functions above threshold in %	total error of functions above threshold in %
0.25	9.0	61.5
0.2	10.7	64.8
0.15	11.8	69.2
0.1	14.4	74.4
0.075	15.5	76.1
0.05	18.2	78.2
0.03	20.8	81.3
0.02	32.3	82.8

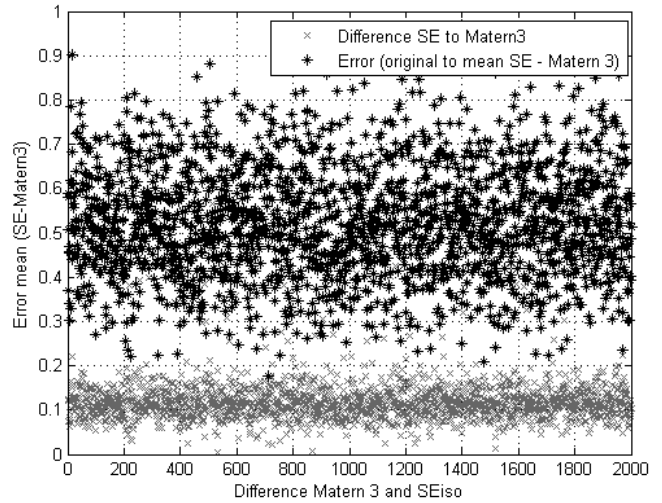


Figure 6: 2000 synthetic functions learned by the squared exponential and the Matérn 3 function. The error between the original function and the mean of squared exponential function and Matérn 3 are depict as well as the difference between squared exponential and Matérn 3.

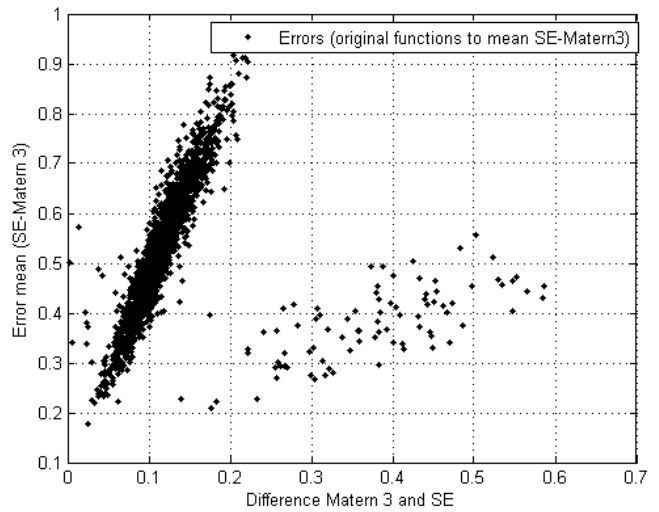


Figure 7: Correlation between differences in the two learned models (squared exponential and Matérn 3) and the error between the squared exponential function and the original data

TABLE II. THRESHOLD AND RESULTS DISTRIBUTION

threshold	number of functions above threshold	total error of functions above threshold in %
0.14	23.85	29.16
0.143	21.95	26.67
0.145	21.05	25.77
0.147	19.65	24.02
0.15	17.90	21.88
0.16	12.35	14.65

good and faulty learning models. Possible settings are depicted in table 1. A threshold level of 0.03 seems to be appropriate for this experimental setting. If the difference between *squared exponential* and *Matérn 5* is higher than 0.03, about 20% of the functions are detected, which are responsible for about 80% of the total error. By adding more data to the learning process of these recognized 20% of functions up to 80% of the total error can be reduced.

The second set of experiments show similar results. There is also a linear dependency between the difference of the models and the total error of the learning model consisting of both covariance functions. That means, if the two learning models differ, the total error gets higher. Fig. 7 shows that in most cases at least one model differs from the original values, because there are only a few functions close to $[0, 0]$. This is different to the results depict in Fig. 5. The effect can be seen in table 2. The 19% of functions, which are over the threshold level of 0.147 are responsible for about 24% of the total error.

The results demonstrate that there is a correlation between the difference in the learning models and the total error. This is a promising result, however, the approach to work with the mean value of two learning models is only good for a small improvement. One possible reason for this can be the minimized number of learning data. At least one model does not provide good estimates in most cases, which reduces quality of the mean value estimates of both.

Even if the approach brings only a small improvement in our experiments here, the approach can be very useful in the real-life application, because there are regularly a few models, which are extremely wrong, e.g., some regression curves go up to infinity. If these cases can be found automatically, this stabilizes the general learning approach a lot.

V. CONCLUSION AND FURTHER STEPS

The presented experiments show that automatic error detection, i.e., faulty tuned models, in Gaussian processes with different kernels is a promising approach. The experiments have shown, that if estimates from different kernels differ strongly, the difference to the original function is high.

The benefit from the approach is, that if less regression models are 'faulty', the 'learning' is more stable and less unfavorable decisions are made. The approach to use different learning kernels, which all seem appropriate for this application, is promising. Since deviations in the models are a clear indication that at least one model provides poor estimates.

Next steps are to further improve the approach where the average estimates of two learning models are used. Some preliminary tests to find the most appropriate kernel and use this kernel mainly, can be a promising approach.

Nevertheless, the presented approach brings more stabilization to the learning process and further steps will be to implement it in the application of production scheduling in the future.

ACKNOWLEDGEMENTS

The authors are grateful to the generous support by the German Research Foundation (DFG), grant SCHO 540/17-1.

REFERENCES

- [1] J. H. Blackstone, D. T. Phillips, and G. L. Hogg, "A state-of-the-art survey of dispatching rules for manufacturing job shop operations," *International Journal of Production Research*, vol. 20, no. 1, pp. 27–45, 1982.
- [2] R. Haupt, "A survey of priority rule-based scheduling," *OR Spektrum*, vol. 11, no. 1, pp. 3–16, 1989.
- [3] S. S. Panwalkar and W. Iskander, "A survey of scheduling rules," *Operations Research*, vol. 25, no. 1, pp. 45–61, 1977.
- [4] W. Mouelhi-Chibani and H. Pierreval, "Training a neural network to select dispatching rules in real time," *Computers & Industrial Engineering*, vol. 58, no. 2, pp. 249 – 256, 2010, scheduling in Healthcare and Industrial Systems.
- [5] B. Scholz-Reiter, J. Heger, and T. Hildebrandt, "Gaussian processes for dispatching rule selection in production scheduling," *Proceeding of the International Workshop on Data Mining Application in Government and Industry 2010 (DMAGI10) As Part of The 10th IEEE International Conference on Data Mining.*, 2010.
- [6] C. K. I. Williams and C. E. Rasmussen, "Gaussian processes for regression," *Advances in Neural Information Processing Systems*, vol. 8, pp. 514–520, 1996.
- [7] C. E. Rasmussen, "Evaluation of gaussian processes and other methods for non-linear regression," *PhD thesis, Department of Computer Science, University of Toronto*, 1996.
- [8] E. Alpaydin, *Introduction to Machine Learning (Adaptive Computation and Machine Learning Series)*. The MIT Press, 2004, vol. 14, no. 1.
- [9] A. O'Hagan, "Curve fitting and optimal design," *Journal of the Royal Statistical Society*, vol. 40, no. 1, pp. 1–42, 1978.
- [10] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, December 2006.
- [11] R. M. Neal, *Bayesian Learning for Neural Networks (Lecture Notes in Statistics)*, 1st ed. Springer, August 1996.
- [12] F. Mosteller and D. L. Wallace, "Inference in an authorship problem," *Journal of the American Statistical Association*, vol. 58, no. 302, pp. 275–309, 1963. [Online]. Available: <http://www.jstor.org/stable/2283270>
- [13] P. Refaeilzadeh, L. Tang, and H. Liu. (2008) Cross-validation, glossar, Arizona State University. last checked 07.07.2011. [Online]. Available: www.public.asu.edu/~ltang9/papers/ency-cross-validation.pdf
- [14] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *International joint conference on artificial Intelligence*, 1995, pp. 1137–1143.
- [15] C. E. Rasmussen and H. Nickisch. (2011) Gpml matlab code version 3.1. last checked: 07.07.2011. [Online]. Available: www.gaussianprocesses.org
- [16] Efron, B. and Tibshirani, R.: An introduction to the bootstrap, Chapman & Hall, 1993

Scalable Resource Provisioning in the Cloud Using Business Metrics

Włodzimierz Funika (1,2)

(1) AGH - University of Science and Technology

Department of Computer Science

al. Mickiewicza 30, 30-059 Kraków, Poland

(2) ACK CYFRONET AGH, ul. Nawojki 11, 30-950 Kraków, Poland

phone: (+4812) 6174466, fax: (+4812) 6339406

email: funika@agh.edu.pl

Paweł Koperek

AGH - University of Science and Technology

Department of Computer Science

al. Mickiewicza 30, 30-059 Kraków, Poland

phone: (+4812) 6174466, fax: (+4812) 6339406

email: pkoperek@gmail.com

Abstract—Currently cloud infrastructures are in the spotlight of computer science. Through offering on-demand resource provisioning capabilities and high flexibility of management cloud-based systems can seamlessly adjust to the constantly changing environment of the Internet. They can automatically scale according to a chosen policy. Despite the usefulness of the currently available tools in this area there is still much space for improvements. In this paper we introduce a novel approach to automatic infrastructure scaling, based on the observation of business-related metrics. We present details on a tool based on this concept, which uses a semantic-based monitoring and management system, called SAMM. At the end we discuss the capabilities of the new mechanisms.

Keywords-cloud computing; monitoring; scaling; resource provisioning; business metrics

I. INTRODUCTION

While becoming a very promising trend for IT [1], cloud computing platforms offer great flexibility and pricing options which are very interesting especially from the end user point of view. The service consumer pays only for the actually used resources and need not worry about providing/maintaining them. All the required computing power, memory and disk space can be used on demand [2]. Along with easy allocation and de-allocation, many cloud environments offer the ability to automatically add and remove new resources based on the actual usage. These *automatic scaling* capabilities can provide a great value. With such a tool it is possible to seamlessly deal with peak load situations and to reduce the costs of handling a stream of service requests which form predictable patterns.

Usually the rules behind a scaling mechanism are based on the observation of *generic* metrics, e.g. the CPU load. This approach does not require spending additional time to develop customized tools and can be easily applied to many different systems. Nevertheless, it is far from perfect. The decision on what to do with the system is based mainly on low level data which indicate that the system is already undergoing high load or some resources are not being used. One has also to keep in mind that it always takes some time to execute a particular action. Launching a VM instance, connecting it to load balancing facilities and redirecting

requests to a new node may take a while. Therefore the action should be executed at such a moment that it will actually improve the situation, instead of generating undesirable overload of the system.

It is common for monitoring tools to provide not only generic, low level information, but also describe the state of resources with metrics tailored to a specific technology or even a particular instance of the system. In many situations such information indicate how much resources will be required in the near future. For example, if the length of the requests queue for computational intensive tasks is rapidly growing, we may be sure that new virtual machines should be deployed. On the other hand, when request buffers on virtual machines are getting empty, the number of running virtual machines may be reduced.

Based on such observations we developed a different approach to automatic scaling. We propose to use higher level data, including customized metrics relevant only to a single system, as decision-making criteria for an auto-scaling mechanism. For example, a resources pool could be extended based on the requests queue length. In this approach we assume that it is far more easier for the user to define triggers for dynamic resource provisioning, when they use concepts directly bound to the application to be scaled.

In this paper we present a modified version of Semantic-based Autonomic Monitoring and Management - *SAMM* ([3], [4]) system, which is using the new paradigm. SAMM is a result of our previous research which was aimed to help administrators meet SLA conditions. To be able to handle this task, the tool monitors a set of customized, high level metrics. They describe the state of a system under observation in the context of business objectives defined in a Service Level Agreement ((SLA)). SAMM was designed to independently modify the application behaviour to prevent it from breaking the contract. Since this level of autonomy is not always desired, we decided to enhance the system with support for custom rules which trigger specified actions.

The rest of paper is organized as follows: Section 2 presents the already existing approaches in the area of

automatic scaling. Next, in Section 3, we provide more details on the enhancements to SAMM and define (Section 4) an environment which was used to test it. In Section 5 we discuss the results obtained. Finally, Section 6 concludes the paper with a short summary and outlines plans for future work.

II. RELATED WORK

Depending on a user's cloud usage model [5], automatic scaling may be understood in different ways. If the user consumes a ready-to-use software application (*Software as a Service model* [6]) the service provider is the party which is responsible for providing a proper amount of resources. There has to be enough computing power and network bandwidth provisioned to meet the conditions of agreements with clients. In this situation, automatic scaling from the end user perspective is only a feature of the system, which allows to easily satisfy the business needs when they arise. For example, a company which uses an on-line office software suite may need to provide such software to ten new employees. Instead of buying expensive licenses and installing the software on their workstations, the IT department requests for additional ten accounts in the online service.

In the *Platform-as-a-Service* model ([7], [8]) the situation is similar. The service provider is also responsible for the automatic scaling of application. However, usually the user has to explicitly request the provisioning of such resources. Providers are able to influence the applications by defining technical constraints for the platform. This way they may ensure that the architecture of the software deployed allows to add and remove some resources dynamically without disrupting normal operation. The algorithm used in decision making may be fine tuned to the underlying hardware and internal resource handling policies of the provider. Usually the user can influence the automatic scaling behavior by setting the upper and lower boundaries of automatic scaling. This prevents from unlimited consumption of resources and therefore from exceeding an assumed budget.

The last model - *Infrastructure-as-a-Service* (e.g. [9]) relies on virtualizing the infrastructure elements like machines and network connections between them. The user has to install and configure everything from scratch and on its top develop their own applications. On the other hand the environment can be customized in many ways, beginning with virtual hardware resources (e.g. CPU power, storage size) and ending with their own server software. Automatic scaling in this model is understood as provisioning on-demand more resources (e.g. virtual machines, virtual storage devices). The user may delegate this task to the service provider ([10]). Adding and removing certain elements is then usually triggered by user-defined rules specifying what action should be taken when a particular threshold is exceeded. These thresholds are limited to a set of metrics predefined by the provider (e.g. CPU usage,

storage usage). Many *IaaS* providers also share an API for operations related to managing acquired resources. With such a manner of interaction with infrastructure, the user may create own management tools which can implement custom automatic scaling policies.

In [11] the authors showed that automatic scaling algorithms working with application-specific knowledge can improve the cost-effectiveness ratio of application deployed in cloud environments. Choosing metrics from a set of traditional system usage indicators as CPU usage, disk operation, and bandwidth usage can be not helpful enough. The authors decided that the deadline for jobs executed by system will be used as a key factor for triggering the auto-scaling of the system.

These examples show that currently existing automatic scalability mechanisms can be improved. The presented tools focus on maximizing resources usage, which does not have to be the most important factor from the user point of view, e.g. it may be more important to have a system with very short request response time instead of high CPU usage. There are attempts to change this situation, but there is no generic tool which would be oriented towards easy adaptation to specific systems.

III. MODIFICATIONS TO THE SAMM SYSTEM

Our approach to automatic scaling is based on the assumption that for each application it is possible to choose a set of metrics, which can be used to estimate how much resources will be required in the nearest future. On the most basic level it should be sufficient to be able to predict whether more resources will be required, or some of those currently running may be stopped. With this point of view, the user is able to determine simple thresholds related to triggering some actions influencing the system in a desired way.

The set of metrics under discussion is tightly coupled with an application for which it is to be chosen. Since the metrics with the same names are often used in different contexts, they may have completely different semantics, e.g. the memory usage may be defined as physical RAM usage or virtual memory usage. To handle so much different information, it is required to use a data representation capable of describing all the used concepts. Furthermore, the monitoring system used in this situation has to provide easy ways to create extensions. There should be a possibility to provide support for new data acquisition techniques. In modern complex software systems, measurements will have to be gathered with help of several very different technologies.

Therefore we decided as a starting point for this research to use the result of our previous work on automatic scaling - the SAMM ([3], [4], [12]) system. SAMM is a monitoring and management software, flexible enough to meet the above requirements. By using ontologies to describe resources and metrics available for observation, it has capabilities to express different system architectures and monitoring

facilities. Owing to its module-based architecture based on OSGi bundles and services, it can be extended by support for new technologies without much effort. Reimplementing and replacing the existing components is also feasible.

To meet the requirement of being able to define rules in a convenient way, we came to a new decision-making module. For this purpose we used the Esper event processing engine ([13]). The internal architecture of SAMM with the enhancements introduced is depicted in Figure 1.

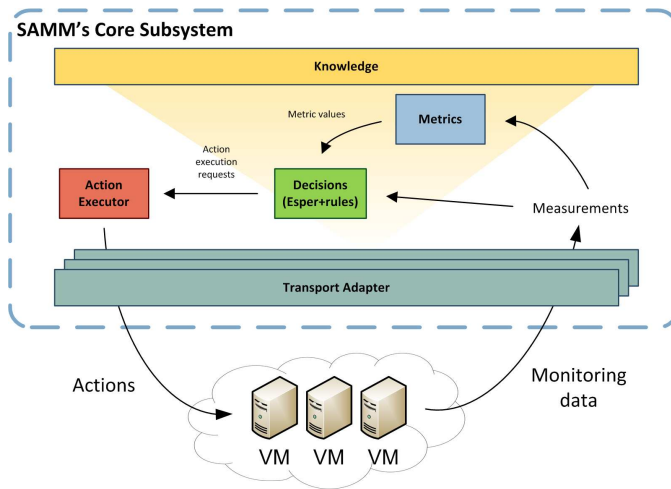


Figure 1. SAMM's architecture after introducing described changes

The flow of measurement activities is as follows:

- 1) Measurements are being collected by the *Transport Adapters*. *Transport Adapters* are an abstraction layer over different technologies used for data collection, e.g. there are separate adapters for Java Management Extensions (JMX)([14]) and Eucalyptus ([15]). They translate internal SAMM measurement requests to language specific constructs for a particular technology, e.g. to Java method invocations.
- 2) *Metrics* module processes measurements according to formulas defined in form of Java classes. The values obtained in this way are sent further to the *Decisions* module.
- 3) Values coming from *Transport Adapters* and *Metrics* modules are processed directly by the *Decisions* module. The Esper event processing engine captures specific events and based on them can trigger an action execution. Rules that describe which events should trigger which actions are defined by the user. It is optional to trigger an action.
- 4) Whenever the *Decisions* module detects exceeding a threshold, an action execution request is sent to *Action Executor*. This component tries to modify the infrastructure by using a specific *Transport Adapter*, since actions can be executed only with help of particular

communication protocols. The exact steps executed by an action are provided as a Java code.

Once SAMM has been enhanced, measurement and metrics values are processed by Esper as events. Owing to this, conditions specified in rules may be very flexible and may include e.g. aggregation functions or consider values only from within a particular interval of time. Since custom queries can be used, the only constraint is the flexibility of the Esper query language.

IV. EVALUATION OF THE APPROACH

To evaluate our approach to automatic resources provisioning we applied the business-metric based scaling policy to a sample application - a simple service for numerical integration. To be able to easily scale the number of nodes used for computations, the *Master - Slave* model was applied. In the application's architecture (presented in Fig. 2) there is a main *master* node, responsible for dispatching the requests and one or more *slave* nodes performing numerical integration. We do not discuss the exact algorithm of numeric integration or its parameters since it is out of scope of interest of the paper.

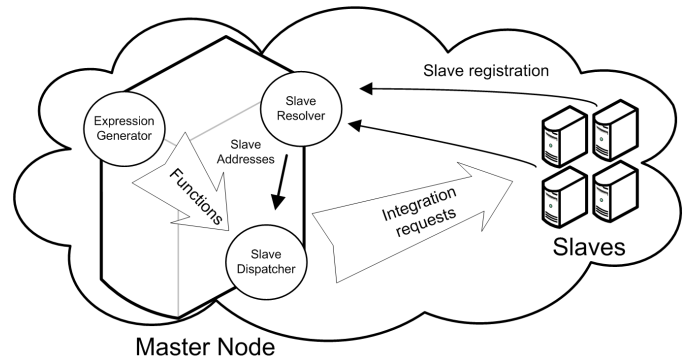


Figure 2. Test application architecture

The *Master* node is built from three components:

- **Slave Dispatcher** - handles the queue of incoming numerical integration requests. If there are any requests in the queue, **Slave Dispatcher** passes them to one of the registered slaves. The *slave* is chosen by performing the following steps:
 - 1) Retrieve a proxy for another *slave* from **Slave Resolver**
 - 2) If the *slave* is capable of handling a next request, send it to this node, else go to point 1.
 - 3) If there are no slaves capable of handling the request - wait e.g. for 5 seconds and start looking for an appropriate node from the beginning.

A slave node is capable of handling a next function if it does not overflow the buffer of numerical integration requests. The size of buffer was set to 25 requests.

The **Slave Dispatcher** is the main component of *Master* node.

- **Slave Resolver** - maintains the set of proxies to *slave* nodes. In this component *slave* nodes register at the beginning of their work. The information about the addresses of nodes are later shared with **Slave Dispatcher**.
- **Expression Generator** - generates the functions for which integration operations are further performed on *slave* nodes. The functions may be generated infinitely or read from a file in chunks of a specified size.

One of the main assumptions about the test application is that always at least one *slave* node has to be running. It was also decided that at most ten instances could get started automatically. The stream of integration requests in the test scenario was tailored to these parameters. The system running all ten *slave* nodes at once could handle without problems the load used in the test scenario.

A. Test Environment

The development work and tests were carried out using the FutureGrid project environment [16]. The **India** Eucalyptus ([15]) cluster was used. The following virtual machine types were provided:

- `m1.small` - 1 CPU, 512 MB of RAM, 5 GB of storage space
- `c1.medium` - 1 CPU, 1024 MB of RAM, 5 GB of storage space
- `m1.large` - 2 CPUs, 6000 MB of RAM, 10 GB of storage space
- `m1.xlarge` - 2 CPUs, 12000 MB of RAM, 10 GB of storage space
- `c1.xlarge` - 8 CPUs, 20000 MB of RAM, 10 GB of storage space

The cluster is built up from 50 nodes and each node is able to run up to 8 `m1.small` instances. *Slave* nodes in our application do not use much storage space and memory. To have got a fine-grained level of the management of the computing power, we decided to use `m1.small` instances for them. The *Master* node application had higher memory requirements, thus we deployed it on a `c1.medium` instance.

B. Case Study

To evaluate the quality of our approach we compared two strategies of automatic scaling. The first one exploits a generic metric - the CPU usage. The second strategy uses a business metric - average time spent by computation requests while waiting in *Slave Dispatcher's* queue for processing. Upper or lower limits for such a metric could be explicitly included in a *Service Level Agreement*, e.g. the service provider might be obligated to ensure that a request won't wait for processing for longer than one minute.

The triggering rules used in the first approach are as follows:

- Start another *slave* node virtual machine: the average CPU usage of *slave* nodes from the last 300 seconds was higher than 90%
- Stop a random *slave* node virtual machine: the average CPU usage of *slave* nodes from the last 300 seconds was less than 50%

The triggering rules used in the second approach are:

- Start another *slave* node virtual machine: the average wait time of request from within the last 300 seconds was higher than 35 seconds
- Stop a random *slave* node virtual machine: the average wait time of request from within the last 300 seconds was less than 10 seconds

The presented parameters were tuned up specifically to the infrastructure on which we carried out the tests and the current load of the cluster. The infrastructure was used in parallel with other users, thus, e.g., the startup time of virtual machines differed over time.

For the evaluation we prepared a test scenario consisting of 120 steps. Each step included adding new requests of numerical integration to *Slave Dispatcher's* queue and waiting for 60 seconds. We had to ensure that two auto-scaling strategies will have to handle exactly the same situation. It was therefore decided to generate the exact functions to process before the actual test and store them in files. Later, when the test was run, *Expression Generator* was reading functions from these files and passing further for processing.

The number of requests added to the queue was equal to

$$ExprNum(n) = 100 + 5 * (n \bmod 80)$$

(where n is the iteration number) of each step of workload. The exact values of formula's constants were chosen in such a way, that a setup of ten virtual machines running constantly during the test scenario could handle the workload. The workload simulates a basic situation requiring automatic scalability capabilities: a constant increase in load in a period of time with a rapid drop right after it.

V. TEST RESULTS

To compare the automatic scaling exploiting the selected strategies, we gathered data about the average wait time (Figure 3), *Slave Dispatcher's* input queue length (Figure 4) and the number of running instances (Figure 5) when executing the test scenario. To sum up the differences we computed the average values of those metrics. Table I presents the results.

At the beginning (the first half of an hour), according to the growing demand for computing power, SAMM started some *slave* nodes. They were kept busy, because the number of integration requests was growing.

While observing the system from the CPU usage point of view, more resources should still be allocated - there was

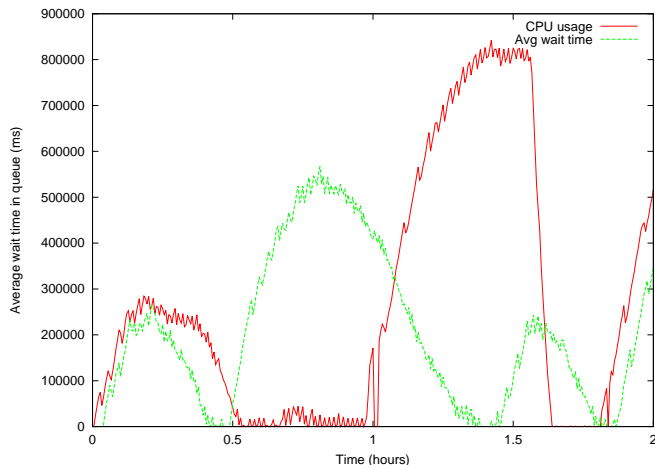


Figure 3. Average wait time during test scenario execution for both strategies (CPU usage and Avg wait time)

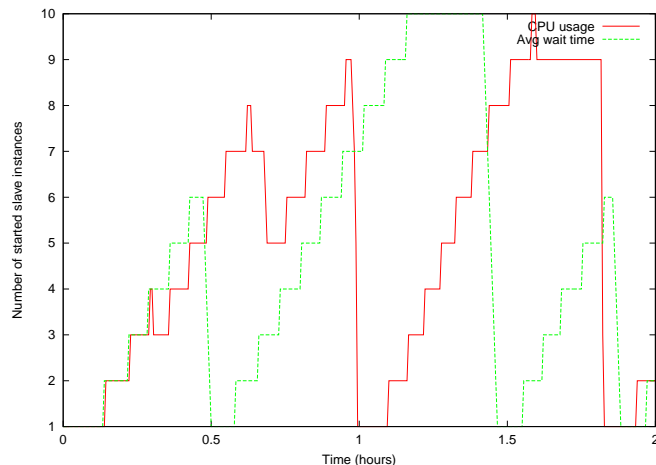


Figure 5. Number of running instances during test scenario execution for both strategies (CPU usage and Avg wait time)

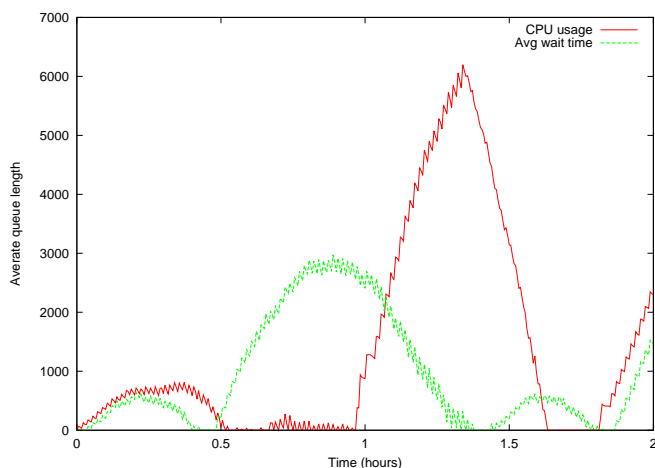


Figure 4. Queue length during test scenario execution for both strategies (CPU usage and Avg wait time)

not enough computing power to process incoming requests. The number of *slaves* was increased until the requests were handled faster than the new ones were added. Once the queue got completely empty, CPU usage dropped and SAMM terminated the unused virtual machines. However, some minutes later, the queue got refilled. Again, more resources were acquired. At the end, after eighty minutes (when the workload rapidly drops), the number of requests in the queue was still high (over 6000).

In the second approach, the rapid drop of average wait time at the beginning also indicated that some resources should get released. The virtual machines were fully used, but the wait time for requests was acceptable. When more and more requests kept joining the queue, the average time required to start the numerical integration got extended. Even if the machines were not utilized 100%, the quality of the

service was dropping according to the *average wait time* metric. SAMM therefore acquired more resources to improve the situation. In a longer perspective such a behavior resulted in a shorter response time in comparison with the CPU usage-based scaling strategy.

Table I
AVERAGE METRIC VALUE FOR AVGWAITTIME AND CPU STRATEGIES

Metric	AvgWaitTime	CPU
Average instances number	4.53	4.96
Average wait time (ms)	203987.08	266362.25
Average queue length	934.06	1392.72

VI. CONCLUSIONS AND FUTURE WORK

Using the average wait time metric has a positive impact on the system when considering its operation from the business point of view. Since the end users are mostly interested in making the time required to wait as short as possible, the amount of the resources involved should be increased according to this demand. By improving this factor, the potential business value of the presented service grows. The system was automatically scaled by SAMM not only from the technical point of view but also from the business value perspective. On the other hand, since the CPU usage was not the main concern, the system may be used not in the most efficient way. If there would be an SLA which does not cover the request wait time, it would be more reasonable to use the CPU usage as a trigger for automatic scaling. Choosing the most appropriate metric to use is up to the user - they have to decide which one is the most important from their perspective.

The actual improvement introduced by the dynamic resource provisioning highly depends on an actual workload. In our test scenario the system shortened the time spent on

waiting by 62 seconds. However there are also situations in which the strategies bound to generic metrics can be as good as the one based on business metrics, but it would require more efforts to choose the most relevant ones and to tune correct limit values.

The possibility to dynamically increase the amount of resources involved in providing a service is a response for the need to efficiently operate in a very quickly changing environment such as the Internet. Adding resources on-demand significantly extends the capabilities of a running system, which is enabled to serve both small and big numbers of users. There is no unused capacity, so the operational costs can be lower.

Making the automatic scaling tools more aware of business rules makes them more useful, especially when they are used in privately held cloud systems. Deep knowledge about the elements running inside the system, provides better insight into scaling rules. They can be fine-tuned to particular hardware and software setup, so the balance between the allocated and spare resources can be properly maintained. For the crucial applications the system could provide all the required computing power and low priority tasks can be automatically maintained. The service provider is able to do more with the same or even smaller amount of computing power.

Our research in automatic scaling area is ongoing. We plan to further extend SAMM with a web interface which would facilitate using its functionality. Another interesting goal of the research is to add support for other cloud stacks, e.g. Open Stack [17] or Open Nebula [18], thus making SAMM interoperable in a heterogeneous environment.

Acknowledgments: This research is partly supported by the AGH University grant. The experiments were carried out using the FutureGrid project resources ¹.

REFERENCES

- [1] Buyya, R. et al. *Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility*. FGCS, vol. 25(6), 2009, 599-616
- [2] Zhang, Q., Cheng, L., Boutaba, R. *Cloud computing: state-of-the-art and research challenges*. Journal of Internet Services and Applications 1, 7-18 (2010).
- [3] Funika, W., Kupisz, M., Koperek, P.: *Towards autonomic semantic-based management of distributed applications*, Computer Science Annual of AGH-UST, vol. 11, 2010, pp. 51–63, AGH Press, Krakow, 2010
- [4] SAMM Project website, <http://code.google.com/p-sammmanager> (accessed: 13.05.2011)
- [5] Rimal, B. P., Choi, E., Lumb, I. A *Taxonomy and Survey of Cloud Computing Systems*, pp.44–51, 2009 Fifth International Joint Conference on INC, IMS and IDC, 2009
- [6] Google Inc, Google Apps website, <http://www.google.com/apps> (accessed: 26.04.2011)
- [7] Google Inc, App Engine project website, <http://code.google.com/appengine> (accessed: 26.04.2011)
- [8] Amazon Web Services LLC, <http://aws.amazon.com/elasticbeanstalk> (access: 09.05.2011)
- [9] Amazon Web Services LLC, <http://aws.amazon.com/ec2> (accessed: 26.04.2011)
- [10] Amazon Web Services LLC, <http://aws.amazon.com/autoscaling> (accessed: 13.05.2011)
- [11] Mao M., Li J., Humphrey M., *Cloud Auto-scaling with Deadline and Budget Constraints*, 11th IEEE/ACM International Conference on Grid Computing (Grid 2010)
- [12] Funika, W. et al., *A Semantic-Oriented Platform for Performance Monitoring of Distributed Java Applications*, in: M. Bubak, G. D. van Albada and J. Dongarra and P. M.A. Sloot (Eds.), Proc. ICCS 2008, volume III, LNCS 5103, Springer, 2008, pp. 233-242.
- [13] Esper Project website, <http://esper.codehaus.org> (access: 10.05.2011)
- [14] Oracle Corporation, website, <http://www.oracle.com/technetwork/java/javavase/tech/javamanagement-140525.html> (accessed: 14.05.2011)
- [15] Eucalyptus Systems, Inc. website, <http://www.eucalyptus.com> (accessed: 13.05.2011)
- [16] FutureGrid Project website, <https://portal.futuregrid.org> (accessed: 13.05.2011)
- [17] Open Stack project website, <http://www.openstack.org> (accessed: 13.05.2011)
- [18] Open Nebula project website, <http://opennebula.org> (accessed: 13.05.2011)

¹<https://portal.futuregrid.org>

e-Reverse Logistics for Remanufacture-to-Order: An Online Auction-based and Multi-Agent System Supported Solution

Bo Xing, Wen-Jing Gao, Kimberly Battle, Fulufhelo Vincent Nelwamondo, and Tshilidzi Marwala

Faculty of Engineering and the Built Environment (FEBE)

University of Johannesburg

Johannesburg, South Africa

bxing2009@gmail.com, wgao2011@gmail.com, kbattle@uj.ac.za, vnelwamondo@gmail.com, tmarwala@uj.ac.za

Abstract—Due to the rapid obsolescent nature of consumer products, the remanufacture-to-stock strategy, in which remanufacturers tend to collect certain amount of end-of-life products, remanufacturing them as many as they can and keep these remanufactured products in stock waiting for customers come to buy, is not always an optimal solution. Under this circumstance, remanufacture-to-order policy, as an effective complement, provides a good trade-off for remanufacturers between meeting consumers’ demand and, in the meantime, keeping the inventory cost at a lower level. To remanufacture the used items, the manufacturer must retrieve them from the market where they are dispersed among consumers. This is accomplished by means of a reverse logistics chain that is comparable to the new product distribution system in reverse. However, the current reverse logistics do not respond to remanufacture-to-order at an efficient level. Therefore it is a necessity to develop a novel infrastructure, which can deal with these issues. This paper presents a framework called e-reverse logistics that aims at filling this gap. The major features and architecture of the proposed e-reverse logistics are detailed in this work.

Keywords—EoL (End-of-Life) product recovery; RMTO (Remanufacture-to-Order); RL (Reverse Logistics); auction; MAS (Multi-Agent System)

I. INTRODUCTION

The EoL (End-of-Life) product recovery field has grown considerably during the past decades, due to its economical benefits and environmental requirements. Among various recovery options, remanufacture is an industrial process where EoL products are restored to like-new condition.

As remanufacturers strive to gain a competitive advantage by shortening their quote-to-delivery cycles; driving out operational costs and non-value add processes; keep lower inventories and better customer service etc., many remanufacture companies consider to employ the strategies associated with “RMTO (Remanufacture-to-Order)”, where used products are acquired and available as needed to meet needs of remanufacture. RL (Reverse Logistics) plays a key role in achieving this goal. In the literature, the research of RL involves many aspects such as network design and inventory management. Various methods have also been applied to deal with RL; see [1] for a recent survey. It can be seen from this review that the problems encountered in RL always require multi-objective optimization. Therefore, artificial intelligence seems to be a

promising and most widely used tool to deal with these issues.

In [2], an attempt was made to design the structure of e-RL (e-Reverse Logistics) to cope with RMTO. Consequent to the previous study, in this research, a novel solution based on auction theory and agent technology is posed to realize e-RL in the context of RMTO. Present study focuses on the implementation of integrated decision-making processes of various actors within an e-RL process.

The rest of the paper is organized as follows: Section II identifies several key challenges of employing RMTO strategy; the proposed e-RL architecture is detailed in Section III; Section IV describes auction based and multi-agent system supported solution for realizing e-RL; finally, the conclusion and future work of the current research are drawn in Section V.

II. CHALLENGES FOR REMANUFACTURE-TO-ORDER

Remanufacturing has been considered as the transformation of used units, consisting of components and parts, into units that satisfy exactly the same quality and other standards as new units. As seen from Figure 1, the general activities involved in EoL product remanufacturing are as follows:

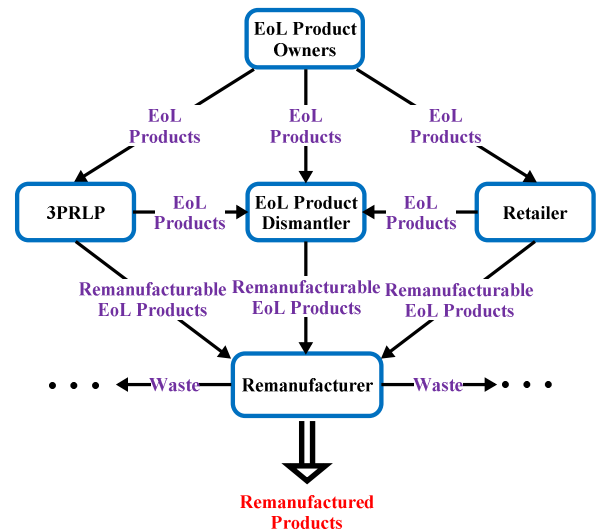


Figure 1. RL process for EoL product remanufacturing.

- Normally, the initial actor group can be identified as the EoL product owners. They are regarded to be the

source of the EoL products emergence. In order to obtain a remanufactured product, an EoL product has to be first collected from its last owners.

- The second group is represented by a set of EoL collectors such as 3PRLP (Third Party Reverse Logistics Provider), authorized EoL product dismantler, and retailer. Though the functionalities of these three types of collectors are slightly different, they act the common role of collecting as many of EoL products as possible from end consumers.
- Remanufacturer is often treated as the third acting group in the practice of EoL product remanufacturing. It is mainly responsible for EoL product disassembly, cleaning, testing, and reassembly. In real world, 100% recovery rate of EoL product is not always achievable, so the remanufacturer not only has to procure new components to finish a remanufactured product, but generating certain amount of product waste and residue as well.

A. The Importance of RMTO

Due to the fact that high uncertainty involved in remanufacturing process such as the problem of imperfect correlation between supply of cores and demand for remanufactured units, the part matching problem and the uncertainties in the quantity and timing of returned products, etc., RMTO is often a practice in industry. Typically an RMTO remanufacturer deals with a customer's enquiry through the following stages:

- First, remanufacturer will receive an order from customer through its ordering system.
- The first stage is an initial evaluation to determine whether the remanufacturer wishes to make a bid for the order. The outcomes of this stage are the decisions to prepare or refuse a bid, and possibly to seek further clarification on the request if it accepts the bid. The availability of cores is one of the reasons that influence remanufacturer's decision.
- In the second stage, the remanufacturer decides how the cost estimates will be prepared. This means specifying how much time should be spent in the estimation process.
- The third stage is the process of preparing the cost estimates themselves. This includes specifying and configuring in detail how the job will be made and also deciding upon material and process plan of the job.
- The final stage is to set the price and lead time to bid. Here the question is to decide the margin of profit to attach to the cost estimates.

After these four stages, the proposal is put to the customer, who may accept it, reject it, or may ask for further negotiations. A further negotiation may just be a request for a lower price or could be a joint exploration of ways to change the specification to reduce the cost. Another possibility on behalf of the customer is to ask the remanufacturer for a new price for a specific delivery date, different to the one proposed by the RMTO remanufacturer [3].

B. The Role of Reverse Logistics

In order to shorten the lead time of a required remanufactured product, it is essential to have, at any time, the parts needed to assembly a new product, which has been ordered. It is made possible if there is on the shelf an exhaustive and comprehensive stock of spare parts (both new and reusable ones). This solution results obviously in a significant cost for any remanufacturer and is not always technically achievable in practice. The alternative strategy is to be able to create an effective required parts supply system to proceed to the remanufacturing process [4]. Under these circumstances, RL, in recent years, have emerged as an important research area due to a tendency towards stricter environmental regulations in industry and an awakening to the economic attraction of recovering products rather than taking the disposal alternative [1]. There are several definitions of RL available in the literature. In this research, the following definition provided by [5] is accepted: RL *“is a process in which a manufacturer systematically accepts previously shipped products or parts from the point for consumption for possible recycling, remanufacturing, or disposal.”*

However, TRL (Traditional Reverse Logistics) process suffers many disadvantages such as core collection speed slow, core delivery delays, and core delivery mistake. Among them, one major challenge faced by the RMTO industry is how to achieve satisfying dearth of good information system while dealing with the inherent variability in TRL. The variability might be caused by a variety of reasons, for example: fluctuating demands, uncertainty of job arrivals, product variety or technological changes. Facing various constraints, the next questions which would probably be asked are: what is the best way to get effective information? How would it be possible to control information to minimize the variability? Meanwhile, there is lacking of system architecture that actually realizes RL.

III. PROPOSED E-REVERSE LOGISTICS ARCHITECTURE

In this section, a brief overview of proposed e-RL is provided. As shown in Figure 2, the realization of e-RL relies on an in-depth understanding of TRL activities together with computer support and communication technologies, namely EPC (Electronic Product Code), NFC (Near Field Communication), and cloud computing.

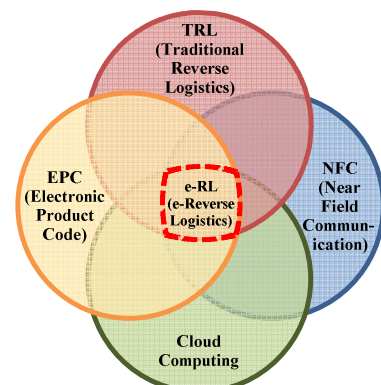


Figure 2. What is e-RL?

In the “e-RL” circle, research is focused on Internet-based decision making models with various distributed sources and human interaction, which would allow the decision structures to be adaptive and more responsive. Additionally, information exchange is required to influence other parties’ local decisions, to act coherently and to achieve better results for the entire system. Briefly, the e-RL framework works as follows:

- **Step 1:** The EoL product owners use their NFC enabled mobile device to publish pieces of used product information online, by simply scanning the ready-to-discard products’ integrated EPC tag.
- **Step 2:** When a remanufacturer lodges a core request to a 4PLP (Forth Party Logistics Provider), she will search online to look for appropriate EoL product information (i.e., an event been published by used product’s last owner). Once 4PLP captures pieces of information, she will immediately forward them to the remanufacturer for further confirmation.
- **Step 3:** Supported by cloud computing, remanufacturer can verify the EoL product information through EPCglobal network, and this will in turn help remanufacturer to evaluate the residue value of a certain used product. Once the remanufacturer decides which core is suitable for his needs, he will send the feedback to 4PLP.
- **Step 4:** When a 4PLP receives a confirmation message, she will distribute the collection orders among a set of other companies with actual transport capacity such as 3PRLP, EoL product dismantler, and retailer.
- **Step 5:** Once a collector successfully gets the order from 4PLP, she will handle any return from the customer to remanufacturer including the physical movement of the goods and so on.

The purpose of establishing e-RL is to achieve lower core acquisition costs for remanufacturer, lower barriers for new suppliers to enter the market of remanufacture, more convenient and environmental conscious ways of discarding EoL products for end customers, and consequently better EoL products remanufacture market efficiency which will in turn make RMTO possible.

Though e-RL is theoretically promising, it still faces some challenges in the real-life implementation: first, the core information provided by 4PLP is multiple and the remanufacturer needs a way to determine which one is the best; second, in order to acquire the collection order from 4PLP, the EoL product collectors such as retailer, 3PRLP and dismantler need an environment that they could negotiate with 4PLP and end customers simultaneously so that a optimal collection solution can be achieved; third, instead of being passively join the EoL product recovery, which in some degree may decrease the efficiency of whole e-RL process, the end customers also need an environment to support them actively choosing the best option to discard an EoL product; last but not the least, the e-RL process involves many dynamic factors, information and communication technologies do cease these problems to certain degree, but still a more advanced method should be proposed to cope with these dynamic influences.

IV. ONLINE AUCTION BASED AND MULTI-AGENT SYSTEM SUPPORTED SOLUTION FOR E-REVERSE LOGISTICS

In this work, the e-RL process is treated as a cooperative distributed system that integrates participating business entities in RMTO, including various EoL product owners, several EoL product collectors, one or more 4PLPs and remanufacturers. This architecture enables and facilitates common economic services and commerce transactions between the consumers and suppliers, such as brokering, pricing, and negotiation, as well as cross-enterprise integration and cooperation in whole reverse logistics. In the conceptual architecture, the e-RL exists as a collection of economically motivated software agents.

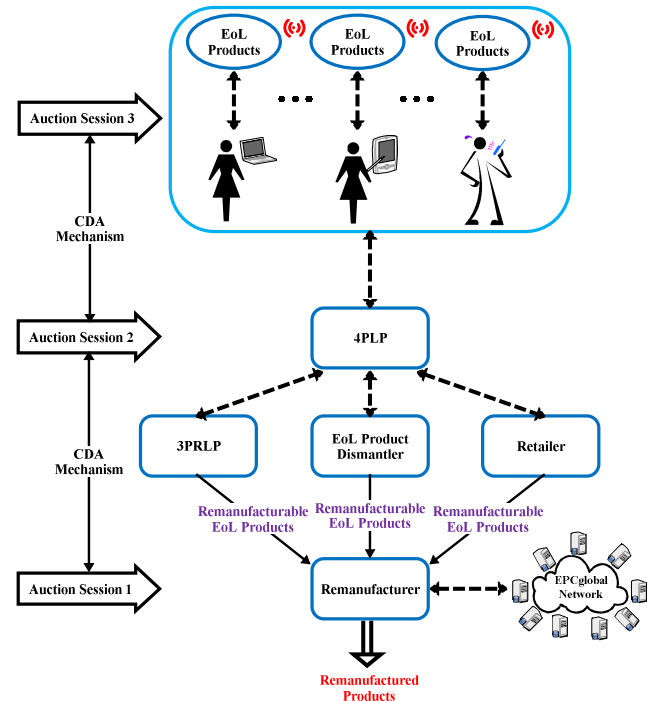


Figure 3. e-RL for RMTO and Auction Sessions.

The online auction based and MAS supported e-RL solution incorporates three auction sessions (see Figure 3) and each of which is viewed as a separate market environment. Different auction mechanisms are employed in every auction session and each participant agent (consumer or supplier) in certain auction session acts independently and contracts to buy or sell at a price agreed upon privately.

A. Auction Sessions

The three auction sessions are detailed in the following subsections:

1) *Auction Session 1_Reverse Vickrey Auction:* In this session, a remanufacturer will publish the core request information, and a 4PLP will look for a suitable core. The underlying auction mechanism of this session is reverse Vickrey auction. In practice, the reverse auction is often used to refer to purchasing auctions (i.e., the roles of the buyer and the seller are swapped). The reason that Vickrey mechanism is chosen lies in that this auction has shown that

bidding one's true evaluation is the dominant strategy. As such, the Vickrey auction eliminates the need for strategizing. Since there is an easy dominant strategy the bidders do not have to think about what they should do. They just play their dominant strategy and bid their true valuation. Thus makes it a very attractive auction in terms of its efficiency compared with other auction types.

Basically, there are three sets of agents in this session as shown below:

- **Core_Request_Agent:** Representing the role of remanufacturer to ask for cores via Internet.
- **Search_Agent:** It works for 4PLP to look for core information online.
- **Auctioneer_Agent:** It has the capability to carry out the buying task in this session.

In general, Auction Session 1 works as follows: the `core_request_agent` formulates a purchase-order and assigns it to an `auctioneer_agent` in Session 1. The `auctioneer_agent` receives a request from the `core_request_agent` to process the purchase-order. Consequently, the `auctioneer_agent` formulates an "announcement" including all the auction parameters supplied by the `core_request_agent` such as the required item details, quantity, minimum price and desired required deadline. This "announcement" is encapsulated inside a "call-for-bid" message that is sent out to all potential bidder agents registered with the current auction session. At a certain time, the `auctioneer_agent` send "propose" messages to all `search_agents` indicating the start of the bidding process. `Search_agents` express their preferences in the format of universal bidding language and send "bid" message back to the `auctioneer_agent`. When the purchase deadline reaches, the `auctioneer_agent` ceases to accept any messages and send a "reject" message for late arriving messages. Immediately after this, the bid evaluation process commences by comparing the structure of aggregated bids against certain classes of bids that indicate an optimal allocation of winners and calculation of payments is ultimately guaranteed. Otherwise, the `auctioneer_agent` runs a heuristic algorithm in order to arrive at a computationally feasible approximate winner allocation. Finally, the `auctioneer_agent` sends "inform" messages to all `search_agents` as well as the `core_request_agent` to notify them of the auction result. Then each agent is responsible for notifying its representative human player of the auction outcome at the end of this auction session.

2) *Auction Session 2_Reverse Combinatorial Auction:* In this session, 4PLP will publish the EoL product collection information, and different collectors will bid for packages of various EoL products. Reverse combinatorial auction mechanism will be employed in Auction Session 2. In combinatorial auction setting, the bidders can place bids for sets of items instead of just placing one bid for each item for sale. This is arguably a more attractive option compared with other auction types in the context of this session.

Thus, apart from `auctioneer_agent` (see Auction Session 1), there are four sets of agents involved in this session:

- **Advertise_Agent:** Working for 4PLP to redistribute the EoL product collection order confirmed by remanufacturer.

- **3PRLP_Agent:** On behalf of 3PRLP to bid EoL products collection.
- **Dismantler_Agent:** Representing EoL products dismantler to bid the collection order.
- **Retailer_Agent:** It stands for retailer to participate in auction.
- **Auctioneer_Agent:** It has the capability to carry out the buying task in session 2.

In principle, this session works as follows: considering a 4PLP, who manages multiple wanted EoL goods to be collected and transported to different destinations, using auctions to make short-term contracts with carriers. When 4PLP has a list transportation orders from remanufacturer, all the carriers that would like to participate in the auction observe the orders and submit their bids for various packages of orders.

3) *Auction Session 3_First-Price, Sealed-Bid Auction:* In this session, EoL product owners will publish the EoL product information, and the collectors will bid for them. In order to increase the customers' willingness-to-return, the first-price, sealed-bid auction is chosen in this session. Generally, in this auction, each bidder places his bid in a sealed envelope. These bids are given to the auctioneer who then picks the highest bid. The winner must pay his bid amount.

In addition to the same agents as described in Auction Session 2 like `3PRLP_agent`, `dismantler_agent`, `retailer_agent`, and `auctioneer_agent`, Auction Session 3 has a new agent set, that is, `mobile_agent` and its characteristics are described as below:

- **Mobile_Agent:** `mobile_agent` acts as the EoL product owner. Basically a `mobile_agent` is capable of moving over a network, such as the Internet, while interacting with foreign hosts, gathering information on behalf of the user and returning to the user after performing its assigned duties.

In general, Auction Session 3 works in this way: when a `mobile_agent` publish a piece of EoL product information, different bidder agents such as `retailer_agent`, `dismantler_agent` and `3PRLP_agent` will submit their bids privately and the highest bidder will eventually win the auction and pays his own bidder.

Meanwhile, in order that multiple sellers and multiple buyers can trade simultaneously, CDA (Continuous Double Auction) mechanism is selected as a connection between different auction sessions. In CDA setting, buyers and sellers can continuously update their bids and asks at any time throughout the trading period.

B. Agent Architecture

In a trading framework agents must interact with each other, selling and buying resources. Trading agents should sell for the higher proposal and vice-versa for buying. Each negotiation consists of a sequence of interactions [6] and thus each committee of agents can be viewed as a player in a game [7]. Basically, every agent consists of knowledge and capability packages, each of which is tailored according to the agent's specific role. An auction-based e-RL mainly recognizes eight sets of agents, namely, `auctioneer_agent`,

core_request_agent, search_agent, advertise_agent, 3PRLP_agent, dismantler_agent, retailer_agent, and mobile_agent. They belong to two types of agents respectively: auctioneer-type agent and bidder-type agent.

In an auctioneer-type agent, the knowledge package contains the information in the agent’s memory about the environment and the expected world. This includes the agent self-model, other agents’ model, goals that need to be satisfied, possible solutions generated to satisfy each goal, and the local history of the world that consists of all possible local views for an agent at any given time. The agent’s knowledge also includes the agent’s desires, commitments and intentions toward achieving each goal [8]. The capability package includes the reasoning component, the domain actions component which contains the possible set of domain actions that when executed the state of the world will be changed, and the communication component where the agent sends and receives messages to and from other agents and the outside world. The architecture of auctioneer-type agent is show in Figure 4 (a).

When bids via agents’ communication messages coming from the bidder-type agents through the communication component assigns a goal-state. The problem-solver component is responsible for the collection of bids and the winner determination process to allocate the item or bundles and their respective prices. Through the interaction component, the auctioneer-type agent sends out an “accept” message at the beginning of auction session informing the supplier-agent that the auction will be carried on its behalf. Once the auction ends, the outcome must be reported to all participants. This information is encapsulated in outgoing messages and sent to the outside world through the communication component.

The role of the bidder-type agent is to express its preferences in bid format and sends “bid” messages out to the auctioneer-type agent. The problem solver component contains a bid class that implements a cyclic behavior in order to respond to incoming messages from the auctioneer-type agent that requests bids. This class implements all the bidder-type agents’ tasks such as registrations and bid valuations. Similarly, as shown in Figure 4 (b), the bidder-type agent is designed using the same agent architectural principles.

The bidder-type agent interacts with the consumer via a user-interface designed to express the atomic bids as well as constraints that signify which item/bundle is mutually exclusive. The formulated bid represents the body of the agent’s communication message that will be sent to the auctioneer-type agent via the communication module.

V. CONCLUSION AND FUTURE WORK

This paper presented an ongoing research on developing a robust architecture of e-RL for meeting RMTO requirements. By integrating advanced information technologies, this framework provides the users with a more convenient option of discarding EoL products, and in the meantime, the framework also allows RMTO practitioners to take decisions on accepting or rejecting customers’ demand in a more real-time manner.

During the development of the online auction based and MAS supported solution for proposed e-RL model, several difficulties were faced by the authors. The first one was related with a lack of commercial available module that would be able to translate end users’ RFID data into appropriate events (i.e., pieces of EoL products information). But according to the literatures, an open source solution called LogicAlloy [9] has been selected as an RFID application level event middleware. LogicAlloy collects and filters the raw data from RFID readers, generates electronic code report and then subscribes the report into other applications. Furthermore, in order to assess the EoL products (i.e., core remanufacturability evaluation), remanufacturers have to access EPCIS (EPC Information Service) database. Currently, such EPCIS repository does not exist. Therefore, another choice of open source software compliant with the EPCIS standard called Fosstrak EPCIS Repository [10] was made in this research. The third difficulty was to model and simulate the proposed solution for e-RL model. By balancing the usability and functionality, NetLogo [11] was chosen in this work. As an agent-based

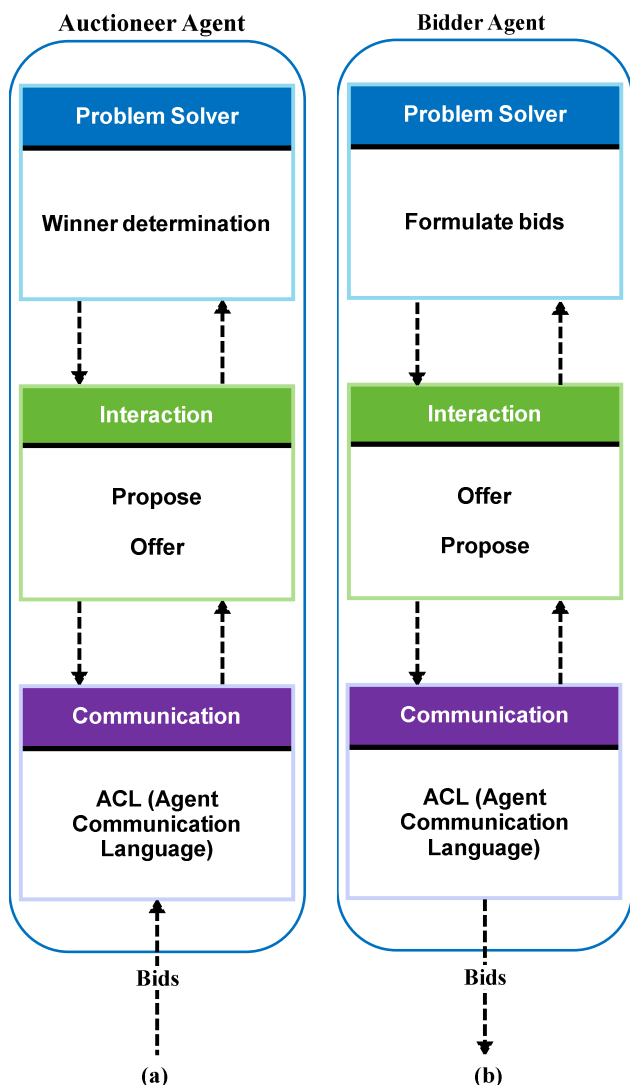


Figure 4. Architecture of auctioneer-type and bidder-type agent.

complex system modeling software environment, NetLogo offers modelers the opportunity to give hundreds or thousands of “agents” all operating independently. This makes it possible to explore the connection between the micro-level behavior of individuals and the macro-level patterns that emerge from the interaction of many individuals. It is believed that through the aforementioned steps, the feasibility of e-RL can be demonstrated.

As a future work, the authors intend to develop a comparison scheme to validate the effectiveness of e-RL in contrary to TRL. The building blocks of this scheme must be able to show that: (1) the customers’ willingness-to-return rate has been improved which can be measured by consumer happiness index; (2) the response delay has been largely reduced in the context of RMTO and (3) the cooperation degree among various EoL product collectors has been increased.

The current version of this research services as an architectural description and main features’ explanation of proposed e-RL. It is believed by combining aforementioned future work plan, the e-RL could permit a robust use of RMTO strategy which, in turn, will make a better contribution to EoL product recovery management.

ACKNOWLEDGMENT

This work is partially supported by National Research Foundation (NRF), South Africa and Dean’s Office, Faculty of Engineering and the Built Environment, University of Johannesburg, South Africa. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

- [1] B. Xing, W. -J. Gao, K. Battle, F. V. Nelvamento, and T. Marwala, “Artificial intelligence in reverse supply chain management: the state of the art,” Proc. of the Twenty-First Annual Symposium of the Pattern Recognition Association of South Africa (PRASA 2010), Stellenbosch, South Africa, November 2010, pp. 305-310.
- [2] B. Xing, W. -J. Gao, K. Battle, F. V. Nelvamento, and T. Marwala, “e-Reverse logistics for remanufacture-to-order: architecture design,” Faculty of Engineering and the Built Environment (FEBE), University of Johannesburg, South Africa, Report: FEBE-2011-June-01, 2011, unpublished.
- [3] A. Nandi, “Input control strategies for make-to-order manufacturing systems via order acceptance/rejection,” PhD Thesis, Department of Mechanical and Manufacturing Engineering, University of Calgary, Calgary, Alberta, Canada, 2000.
- [4] F. Pères and D. Noyes, “Envisioning e-logistics developments: making spare parts in situ and on demand State of the art and guidelines for future developments,” Computers in Industry, vol. 57, 2006, pp. 490-503.
- [5] S. Dowlatshahi, “Developing a theory of reverse logistics,” Interfaces, vol. 30, May-June 2000, pp. 143-155.
- [6] P. Mariano, *et al.*, “Simulation of a trading multi-agent system,” Proc. of IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC 2001), Tucson, Arzon, USA, October 2001, pp. 3378-3384.
- [7] T. Marwala, *et al.*, “Scalability and optimisation of a committee of agents using genetic algorithm,” Proc. Fourth Int. ICSC Symposium Soft computing and Intelligent Systems for Industry (SOCO/ISFI 2001), Paisley, Scotland, United Kingdom, June 2001, pp. 1-7.
- [8] H. Ghenniwa, J. Dang, M. Huhns, and W. Shen, “eMarketplace model: an architecture for collaborative supply chain management and integration,” in Multiagent based supply chain management. vol. 28, B. Chaib-draa and J. P. Müller, Eds. Berlin Heidelberg: Springer-Verlag, 2006, pp. 29-62.
- [9] LogicAlloy, <<http://www.logicalloy.com>> 25.09.2011
- [10] Fosstrak EPCIS Respository, <<http://www.fosstrak.org>> 25.09.2011
- [11] U. Wilensky, “NetLogo (4.1 ed),” Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL., <<http://ccl.northwestern.edu/netlogo/>> 25.09.2011

e-RL: The Internet of Things Supported Reverse Logistics for Remanufacture-to-Order

Bo Xing, Wen-Jing Gao, Kimberly Battle, Fulufhelo Vincent Nelwamondo, and Tshilidzi Marwala

Faculty of Engineering and the Built Environment (FEBE)

University of Johannesburg

Johannesburg, South Africa

bxing2009@gmail.com, wgao2011@gmail.com, kbattle@uj.ac.za, vnwamondo@gmail.com, tmawala@uj.ac.za

Abstract—With the increasing globalization, there are many sources of uncertainty across the entire reverse logistics for end-of-life product recovery. Information sharing is a key ingredient in this respect because it helps to eradicate potential uncertainties related to various corporate behaviors, especially in remanufacture-to-order aspect. However traditional reverse logistics does not fully address itself to this issue. As a result, the progressive loss of product information as it moves through its lifecycle becomes inevitable which leads to a difficulty in implementing remanufacture-to-order strategy. Motivated by this fact, this study presents an ongoing design of novel reverse logistics (i.e., e-reverse logistics). The Internet of things technology is introduced into this framework for the purpose of keeping product life cycle information integrity. Furthermore, the cooperation and collaboration between actors involved in proposed architecture are treated in multi-agent system philosophy. The design characteristics and working principles of e-reverse logistics are detailed in this paper and it serves as a start point for further implementation.

Keywords—*RL (Reverse Logistic); IoT (Internet of Things); MAS (Multi-Agent System)*

I. INTRODUCTION

The focus of this study will be on issues pertaining to IoT (Internet of Things) and RL (Reverse Logistics). Both fields are intended to be joined into research which will investigate the following key research question:

How IoT facilitated e-RL (e-Reverse Logistics) can reduce the blind spots during the EoL (End-of-Life) product recovery process?

End users, collectors, remanufacturers, redistributors and retailers are typically interconnected within networks and it is for this reason that the relationships among all of them are represented as reverse logistic networks. Having enough information about EoL products in which they are involved can be useful in RMTO (Remanufacture-to-Order) planning strategies or in assuring EoL products quality.

Built on intelligent products, the IoT, as a stimulating idea, is fast emerging in the wireless scenario. One of the main challenges faced in this area was the integrity of data as well as using a standard data format for sharing information among different RL participants. Nowadays, several approaches are used to implement information

management systems architectures, such as the EPCglobal Network [1] developed by the Auto-ID consortium, the Dialog System [2] developed at the Helsinki University of Technology, the WWAI (World Wide Article Information) system proposed by Trackway, and the ID@URI approach [3]. The EPCglobal Network stands out among the rest because in 2003 it was authorized as a GS1 (Global Standards I) [4]. The GS1 system of standards is the most widely-used supply-network standards system in the world.

Motivated by these facts, in this research, e-RL is introduced, standing for the traditional RL plan with the addition of “e” element. It aims at improving the bridge between information networks and EoL product flows to form a seamless, synchronous network functioning, so that to support RL process.

Briefly, the rest of this article is structured as follows: In Section II, the fundamental of RMTO is briefly introduced; the background of the study is outlined in Section III; Section IV identifies several key backbone of e-RL proposal; Section V details the proposed e-RL strategy; research expectations are concluded in Section VI; finally, the conclusion and future work are drawn in Section VII.

II. REMANUFACTURE-TO-ORDER

The RL/remanufacture has many similarities to its traditional forward logistics/manufacturing counterpart. At the most basic level, both involve supply, production and distribution. The major difference between the two involves the supply side [5], especially in used products (refer to as core) acquisition process [6].

To avoid the uncontrolled accumulation of core inventory, the issue for the RMTO strategy would put remanufacturing industry in the spotlight. This strategy provides an interface of high variety products in relatively low volumes. Often the volumes are low even at the component production stage as there is little scope for common components because cores are remanufactured to customer design and specification. Meanwhile, basis on RMTO strategy can reduce lead time achieved when implementing new manufacturing principles such as lean production [7].

III. BACKGROUND OF THE STUDY

A. Traditional Reverse Logistics

There are many aspects covered by TRL (Traditional Reverse Logistics) research such as network design, vehicle routing problem, and inventory management. Interested readers please refer to [8][9][10] (a three parts comprehensive survey dedicated to TRL) for more details. However, Rogers, *et al.* [11] argued that in practice the lack of information systems infrastructure was one of the largest barriers confronting the RL executives. For example, Ranasinghe, *et al.* [12] reported that it was evident that acceptability and widespread use of product unique identity (such as electronic product code) was visible in many aspects of RL in general and in the field of remanufacture in particular. In the TRL literatures, little research work has been carried out in the context of information technology and more studies are needed to conduct in this direction so that sharing information can become a possible during RL process. Thus a major contribution of this paper lies in that introducing advanced IoT technology into RL area.

B. Remanufacture-to-Order

One of the challenge associated with RMTO is the EoL product information to which the remanufacturer has embraced core acquisition appears equivocal. In a remanufacture system, core acquisition is largely exogenous, and the timing, quantity, and quality of core are much more uncertain than in traditional production distribution systems. Parlikad, *et al.* [13] argue that a fundamental obstacle in achieving more acceptable cores acquisition levels lies in that information related to the cores is often irrecoverably lost after the point of sale. Moreover, a great variety in offered products and the complexities involved in RMTO environment makes it a difficult task to collect a wide range of information. Thus the information sharing, collaboration, and coordination in an effort to improve channel efficiency is a necessity in RMTO area [14][15]. Therefore, by introducing a novel RL structure, this research can be treated as an initial attempt to solve these issues.

IV. BACKBONE OF E-REVERSE LOGISTICS PROPOSAL

e-RL is concerned with the applications of the IoT in RL area. Triggered by advanced computing techniques, one can witness that IoT has far-reaching impacts on citizens, businesses, and society as a whole. Therefore the purpose of this section is to give reader a quick overview of this disruptive idea.

A. What is Internet of Things?

In general, the IoT can be defined as: “A *global network infrastructure, linking physical and virtual objects through the exploitation of data capture and communication capabilities. This infrastructure includes*

existing and evolving Internet and network developments. It will offer specific object-identification, sensor, actuator and connection capability as the basis for the development of independent federated services and applications. These will be characterized by a high degree of autonomous data capture, event transfer, network connectivity and interoperability [15].”

Under the term of “IoT”, electronic systems can be, in principle, integrated into all conceivable objects around us. This ubiquity of information technology can thus be applied in many fields ranging from industrial production up to individual, daily life. In a word, no human endeavor or thought would be unchanged by IoT.

B. Enabling Technologies

Currently many techniques are driving the progress of IoT at different levels and they can be classified into the following categories.

1) *Context Information Identification Techniques:* The context information identification is the essence of IoT. Currently the technique like PEID (Product Embedded Information Device) [16] is a potential way of providing real-world entities with certain degree of “intelligence” so that the required level of context awareness can be achieved.

2) *Context Information Resolution Techniques:* Apart from the representation object identification technology, spontaneous interoperation is also a necessary which means after obtaining the complex context information of a physical object provided by a PEID, a database-like infrastructure is required to solve any potential queries. The most important use of context information resolution is to find an information service associated with objects. In practice, EPCglobal network [1] is an industry driven, product centric data management architecture to provide product’s traceability based on PEID technology.

V. DESCRIPTION OF PROPOSED E-REVERSE LOGISTICS STRATEGY

A. Key Players of e-Reverse Logistics Strategy

Several actors are involved in proposed e-RL strategy and their characteristics are laid out as follows:

1) *End User:* The roles of end user are twofold: on one hand, anyone who owns an EoL product can be referred to as an end user; on the other hand, end user also means a customer who purchases a remanufactured product directly from remanufacturer’s online ordering system. Both groups of end users are coexisted in the marketplace and keep switching roles with each other.

2) *EoL Product Collector:* In the proposed framework, an EoL product collector refers to any business entity that has been authorized for participating EoL product collection activities.

3) *RMTO Firm:* A RMTO firm means a remanufacturer who actually carries out EoL product

remanufacture activities and in the meantime employs RMTO policy. There are several types of RMTO firms in practice such as OEM (Original Equipment Manufacturer) and independent remanufacturers. But in this research they are treated as same since e-RL strategy is engineered to fit them all.

B. Brief Summary of e-Reverse Logistics Strategy Working Principles

The working principles of e-RL strategy is summarized and mapped in Figure 1. The red dashed lines represent the interacting information flows between different players, while the green solid lines show the physical product transportation process.

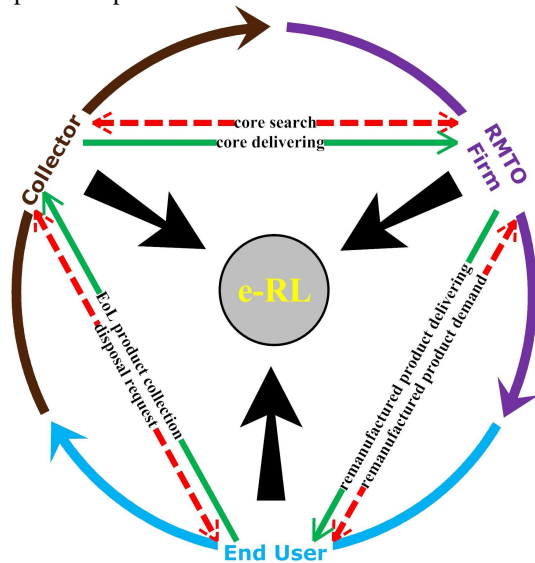


Figure 1. Processes mapping of e-RL strategy.

Normally, in each round of iteration, the e-RL strategy will perform as follows:

- First, the sales department receives an order from a customer through the company’s online ordering system.
- In order to perform an evaluation of whether the company should make a bid or not for such order, decision makers must take into account various factors. Among them, the existence of cores is the most influential one. Therefore, the sales department will immediately forward the core existence enquiry to the supply department.
- After receiving such enquiry from the sales department, the supply department will at once set out to search cores through the EPCglobal network.
- The outcome of this lookup stage will be the required cores cannot or can be found. If the answer is no, the supply department will notify the sales department and the latter will reject this particular customer’s order; otherwise if the answer is yes, the supply department will receive a

list of candidate cores’ specifications (i.e., EPC information, it is required to send a query to the EPC information system of all sites until a whole lifecycle data is located.) and such information will be promptly forwarded to production department.

- When candidate cores’ EPC information reach the production department, several sub-departments within it (e.g., disassembly, remanufacturing, and inventory departments) will be further consulted to help with the issues like remanufacturability evaluation, remanufacturing cost pre-calculation, and lead time estimation.
- After this stage, the sales department will finally receive a report-like feedback from production department in which all the useful results are outlined. Based on this feedback, the sales department will negotiate with the customer about the price, due date, payment methods, and so on.
- Once a purchasing order has been actually made, the sales department will send a copy to supply department and the latter will start buying cores from collectors.
- As soon as the production department receives physical cores, a designated remanufacturing process will be carried out.
- When a remanufactured product passes the prescribed testing procedure at the production department, it will be finally delivered to the corresponding customer.
- The product will stay in the marketplace until it reaches the end of its current life cycle. By that time, the end user will send a disposal request to EoL product collectors via his mobile device which embedded the NFC (Near Field Communication) tags and later on the allocated collector will sort out the collection issue.

Both the upstream and downstream flows (and the interactions between them) are considered within the concept of e-RL.

VI. RESEARCH EXPECTATIONS

A. Expected Outcomes

- Establishing an architecture of e-RL, which is built on MAS (Multi-Agent System).
- e-RL strategy provides RL participants with a self-developed EPCglobal network to support EoL products’ lifecycle information sharing and more effective communication amongst the e-RL participants. With such feature, each RL participant can track and trace the EoL products anywhere and anytime in a timely manner.
- Modeling and simulation of the proposed idea in NetLogo [17] and open source environment.

B. Expected Contributions

- Improving the collaborative efficiency and effectiveness in RL.
- Obtaining a better understanding of the value of information in the field of EoL product remanufacture.
- With improved visibility of the e-RL process, companies can respond to customer demand more promptly and efficiently by identifying the scattered cores in the marketplace that need to be acquired for RMTO purpose.
- It should be noted that the system is not limited to RMTO process, it can also integrate such information with other applications to support decision making and communication.

VII. CONCLUSION AND FUTURE WORK

The paper presented an idea of e-RL, which is built on IoT and MAS approaches. Main features and working principles were discussed as well. This architecture design of e-RL is the first step towards a successful RMTO policy implementation.

As a future work, validating the feasibility and effectiveness of proposed e-RL will be the focus. On one hand, the following work will be carried out to prove the new idea's feasibility: (1) open source solution for RFID event generation and searching and (2) agent-based modeling and simulation. On the other hand, in order to demonstrate the e-RL's effectiveness, the comparison with TRL will be made from the following perspectives: (1) customer's willingness-to-return; (2) processing speed of EoL products collection and (3) robustness of RMTO practitioners responding to market demand.

This position paper treats RL in the context of RMTO from a novel viewpoint. Its architecture characteristics and design features are different with TRL from many aspects. It is expected, by completing the future work plan, the e-RL would serve as a solid basis for employing RMTO strategy in practice which, on a higher level, will make the sustainable development become possible.

ACKNOWLEDGMENT

This work is partially supported by National Research Foundation (NRF), South Africa and Dean's Office, Faculty of Engineering and the Built Environment, University of Johannesburg, South Africa. The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

- [1] K. S. Leong, M. L. Ng, and D. W. Engels, "EPC network architecture," Auto-ID Center, Institute for Manufacturing, University of Cambridge, Cambridge, United Kingdom, Report: AUTOIDLABS-WP-SWNET-012, September 2005, unpublished.
- [2] K. Främling, T. Ala-Risku, M. Kärkkäinen, and J. Holmström, "Agent-based model for managing composite product information," *Computers in Industry*, vol. 57, 2006, pp. 72-81.
- [3] D. Kiritsis, A. Bufardi, and P. Xirouchakis, "Research issues on product lifecycle management and information tracking using smart embedded systems," *Advanced Engineering Informatics*, vol. 17, 2003, pp. 189-202.
- [4] Global Standards I, <<http://www.gs1.org>> 25.09.2011
- [5] M. Fleischmann, *et al.*, "Quantitative models for reverse logistics: a review," *European Journal of Operational Research*, vol. 103, 1997, pp. 1-17.
- [6] V. D. R. Guide and V. Jayaraman, "Product acquisition management: current industry practice and a proposed framework," *International Journal of Production Research*, vol. 38, 2000, pp. 3779-3800.
- [7] L. C. Hendry, "Applying world class manufacturing to make-to-order companies," *International Journal of Operations & Production Management*, vol. 18, 1998, pp. 1086-1100.
- [8] P. Sasikumar and G. Kannan, "Issues in reverse supply chains, part I: end-of-life product recovery and inventory management – an overview," *International Journal of Sustainable Engineering*, vol. 1, Sep. 2008, pp. 154-172.
- [9] P. Sasikumar and G. Kannan, "Issues in reverse supply chains, part II: reverse distribution issues – an overview," *International Journal of Sustainable Engineering*, vol. 1, Dec. 2008, pp. 234-249.
- [10] P. Sasikumar and G. Kannan, "Issues in reverse supply chain, part III: classification and simple analysis," *International Journal of Sustainable Engineering*, vol. 2, Mar. 2009, pp. 2-27.
- [11] D. S. Rogers and R. S. Tibben-Lembke, *Going backwards: reverse logistics trends and practices*. Pittsburgh, PA: Reverse Logistics Executive Council, 1999.
- [12] D. C. Ranasinghe, M. Harrison, K. Främling, and D. McFarlane, "Enabling through life product-instance management: solutions and challenges," *Journal of Network and Computer Applications*, vol. 34, 2011, pp. 1015-1031.
- [13] A. K. Parlikad, D. McFarlane, E. Fleisch, and Sandra Gross, "The role of product identity in end-of-life decision making," Auto-ID Center, Institute for Manufacturing, University of Cambridge, Cambridge, United Kingdom, Report: CAM-AUTOID-WH-017, Jun. 2003, unpublished.
- [14] F. Sahin and E. P. Robinson, "Flow coordination and information sharing in supply chains: review, implications and directions for future research," *Decision Sciences*, vol. 33, 2002, pp. 505-536.
- [15] M. A. Moisescu, I. Ş. Sacală, and A. M. Stănescu, "Towards the development of Internet of things oriented intelligent systems," *Ū.P.B. Science Bulletin, Series C*, vol. 72, 2010, pp. 115-124.
- [16] D. Kiritsis, "Closed-loop PLM for intelligent products in the era of the Internet of things," *Computer-Aided Design*, vol. 43, 2011, pp. 479-501.
- [17] U. Wilensky, "NetLogo (4.1 ed)," Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL., <<http://ccl.northwestern.edu/netlogo/>> 25.09.2011

Particle Swarm Optimization for Nonlinear Model Predictive Control

Julian Mercieca and Simon G. Fabri
 Department of Systems and Control Engineering
 University of Malta
 Msida, Malta

Email: julianmercieca@gmail.com, simon.fabri@um.edu.mt

Abstract—The paper proposes two Nonlinear Model Predictive Control schemes that uncover a synergistic relationship between on-line receding horizon style computation and Particle Swarm Optimization, thus benefiting from both the performance advantages of on-line computation and the desirable properties of Particle Swarm Optimization. After developing these techniques for the unconstrained nonlinear optimal control problem, the entire design methodology is illustrated by a simulated inverted pendulum on a cart, and compared with a particular numerical linearization technique exploiting conventional convex optimization methods. This is then extended to input constrained nonlinear systems, offering a promising new paradigm for nonlinear optimal control design.

Index Terms—particle swarm optimization; model predictive control; optimal control

I. INTRODUCTION

Nonlinear Model Predictive Control (NMPC) is an attractive control scheme for manipulating the behaviour of complex systems [1], exhibiting excellent dynamic performance in both industrial applications and theoretical studies [2]–[4]. However its application to nonlinear control is complicated, largely due to the optimization method that has come to be used in these controllers. A fundamental difficulty of the NMPC approach is the requirement to solve nonconvex constrained optimization problems. Most existing works are based on nonlinear programming methods [5] which only yield local optimum values, with the latter depending on the selection of the starting point. For this purpose, in [6] a particular numerical linearization technique has been developed to obtain a convex constrained optimization problem, albeit at the cost of performance deterioration. Other attempts to solve the nonconvex optimization problems exploit Genetic Algorithm (GA) optimizers [7]. However these face many challenges, including enormous computational effort due to its natural genetic operations [8], [9]. Although this may be reduced by using a real-value representation in the GA [8], [10], some deficiencies in GA performance have been highlighted in recent research. Applications governed by highly epistatic objective functions [11], [12] reveal shortfalls in performance, which is further worsened by the GA's premature convergence [11].

This paper presents two novel NMPC controllers based on a very powerful optimizer: Particle Swarm Optimization (PSO). First developed by Kennedy and Eberhart in 1995 [13], this

modern metaheuristic algorithm has been found to be robust in solving continuous nonlinear optimization problems [10], [12]–[14] and capable of generating high quality solutions with more stable convergence characteristics and shorter calculation times than other stochastic methods [10], [12], [15]. One of the novel controllers presented in this paper is based on a numerical linearization technique first proposed by Alaniz in [6], which is based on conventional convex optimization methods. By contrast, the proposed controller exploits PSO techniques for optimization. The second novel controller proposed in this paper does away with any form of numerical linearization to achieve optimization of the cost function. Both controllers are simulated for an inverted pendulum on cart problem and compared with the NMPC controller in [6].

The rest of the paper is organized as follows. Section II is a brief explanation of the implemented PSO algorithm, while Section III outlines the design of the three NMPC controllers evaluated in this paper. Section IV then presents the simulation setup, results and analysis, followed by a brief conclusion in Section V.

II. PARTICLE SWARM OPTIMIZATION

The particle swarm optimization (PSO) algorithm [13] is a population-based search algorithm inspired by the social behaviour of birds within a flock. The very simple behaviour followed by individuals emulates their own successes and the success of neighbouring individuals. The emergent collective behaviour is that of discovering optimal regions of a high dimensional search space.

In a PSO algorithm, each particle representing a potential solution is maintained within a swarm. In simple terms, the particles are therefore “flown” through a multidimensional search space where the position of each particle is adjusted according to the experience of itself and its neighbours. Let $\mathbf{x}_i(t)$ denote the position of particle i in the search space at time step t , which denotes discrete time steps unless otherwise stated. The position of the particle is changed by adding a velocity vector, $\mathbf{v}_i(t)$, to the current position *i.e.*

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1) \quad (1)$$

with $\mathbf{x}_i(0) \sim U(\mathbf{x}_{min}, \mathbf{x}_{max})$, where $U(\mathbf{x}_{min}, \mathbf{x}_{max})$ denotes the continuous uniform probability distribution within the real-valued space $(\mathbf{x}_{min}, \mathbf{x}_{max})$. The optimization process is driven

by the velocity vector, reflecting both the experiential knowledge of the particle (*cognitive component*) and socially exchanged information from the particle's neighbourhood (*social component*). In this paper we implement a particular PSO algorithm known as global best PSO, which exhibits very fast convergence rates [16] much needed for our predictive control application. For the global best PSO, or *gbest* PSO, the neighbourhood for each particle is the entire swarm, thus employing the social network of the star topology type. In this situation, the social information is the best position found by the swarm, referred to as $\hat{\mathbf{y}}(t)$.

For *gbest* PSO, the velocity of particle i is calculated as

$$v_{ij}(t+1) = v_{ij}(t) + c_1 r_{1j}(t)[y_{ij}(t) - x_{ij}(t)] + c_2 r_{2j}(t)[\hat{y}_j(t) - x_{ij}(t)] \quad (2)$$

where $x_{ij}(t)$, $y_{ij}(t)$ and $v_{ij}(t)$ are the position, personal best position and velocity of particle i in dimension $j = 1, \dots, n_x$ at time step t respectively, $\hat{y}_j(t)$ is the global best position in dimension j , c_1 and c_2 are positive acceleration constants used to scale the contribution of the cognitive and social components respectively, and $r_{1j}(t), r_{2j}(t) \sim U(0,1)$ are random values in the range $[0, 1]$, sampled from a continuous uniform distribution. These random values introduce a random element to the algorithm. Algorithm 1 summarizes the *gbest* PSO, where \mathbf{y}_i denotes the personal best position associated with particle i and $\hat{\mathbf{y}}$ denotes the global best position.

Algorithm 1 *gbest* PSO

Create and initialize an n_x -dimensional swarm

repeat

for each particle $i = 1, \dots, n_s$ **do**

//set the personal best position

if $f(\mathbf{x}_i) < f(\mathbf{y}_i)$ **then**

$\mathbf{y}_i = \mathbf{x}_i$;

end

//set the global best position

if $f(\mathbf{y}_i) < f(\hat{\mathbf{y}})$ **then**

$\hat{\mathbf{y}} = \mathbf{y}_i$;

end

end

for each particle $i = 1, \dots, n_s$ **do**

update the velocity using equation (2);

update the position using equation (1);

end

until stopping condition is true;

III. NONLINEAR MODEL PREDICTIVE CONTROL

A nonlinear dynamic system may be represented by a set of nonlinear differential equations [17], which may be discretized for computational purposes using Euler's method, where T_s is the sampling period and k is the sample index in discrete-time, as follows:

$$\mathbf{x}(k+1) = \mathbf{x}(k) + T_s f(\mathbf{x}(k), \mathbf{u}(k), \mathbf{v}(k), k) \quad (3)$$

$$\mathbf{y}(k) = g(\mathbf{x}(k), \mathbf{u}(k), \mathbf{v}(k), k) \quad (4)$$

Arguments of the nonlinear function f include a state vector $\mathbf{x}(k)$, a control input $\mathbf{u}(k)$, and a disturbance input $\mathbf{v}(k)$. The set of physical quantities that can be measured from the system constitute the output, $\mathbf{y}(k)$, which is also a nonlinear function g of the same arguments. More accurate discretization approximations, such as the Runge-Kutta methods, can be used if the system dynamics are highly nonlinear or the desired time step is large.

The Model Predictive Control (MPC) design methodology is characterized by three main features: an explicit model of the plant, computation of control signals by optimizing predicted plant behaviour, and a receding horizon [18]. An internal model is used to predict how the plant will react, starting at the current time k , over a discretized prediction interval. The objective is to select the control history that results in the best predicted behaviour with respect to a reference trajectory and optimization parameters.

The cost function used in this paper is given by equation (5) which has a quadratic structure comprising two terms. The first term, weighted by a symmetric weighting matrix $\mathbf{Q}(k)$, penalizes the deviations from a reference trajectory that occur throughout the prediction interval. The second term, weighted by a symmetric weighting matrix $\mathbf{R}(k)$, penalizes the magnitude of each control value in the control history. We will now describe the three nonlinear model predictive controllers considered in this paper, two of which represent the novel contributions of this work.

$$J = \sum_{i=0}^{l-1} \|(\mathbf{y}(k+i) - \tilde{\mathbf{y}}(k+i))\|_{\mathbf{Q}(k+i)}^2 + \sum_{i=0}^{m-1} \|\mathbf{u}(k+i)\|_{\mathbf{R}(k+i)}^2 \quad (5)$$

A. A numerical linearization method

This method, proposed by Alaniz in [6], centres around a particular numerical linearization technique for generating the predicted output trajectory \mathbf{y} . A nominal control history $\bar{\mathbf{u}}$ is first chosen, then the corresponding nominal output trajectory $\bar{\mathbf{y}}$ is computed through numerical integration. Typically $\bar{\mathbf{u}}$ is the previous optimal solution, but it can be set equal to zero if none exist. The predicted output is then based on linearizing the control perturbation $\Delta\mathbf{u}$ about the nominal trajectory as follows:

$$\begin{aligned} \mathbf{y}(k) &= \bar{\mathbf{y}}(k) + \alpha_0 \Delta\mathbf{u}(k) \\ \mathbf{y}(k+1) &= \bar{\mathbf{y}}(k+1) + \alpha_1 \Delta\mathbf{u}(k) + \beta_0 \Delta\mathbf{u}(k+1) \\ \mathbf{y}(k+2) &= \bar{\mathbf{y}}(k+2) + \alpha_2 \Delta\mathbf{u}(k) + \beta_1 \Delta\mathbf{u}(k+1) + \gamma_0 \Delta\mathbf{u}(k+2) \\ &\vdots \end{aligned} \quad (6)$$

The coefficients $\alpha_i, \beta_i, \gamma_i, \dots$ are produced by computing a perturbed trajectory for each $\Delta\mathbf{u}(k+i)$ and finding the subsequent deviation from the nominal trajectory. Perturbed trajectories are the result of adding a pulse of magnitude one to the nominal control history at time $(k+i)$. Each trajectory is formed by propagating the present state $\mathbf{x}(k)$ over a fixed interval of time while applying an associated control history. The prediction interval and control history are divided into l and m discrete steps, respectively, of length T_s , where T_s is the

time step, and $m \leq l$. After the control history has ended, the control is held constant for the final $(l - m)$ time steps.

The MPC problem is to solve for the optimal control perturbation $\Delta \mathbf{u}^*$ by minimizing a cost function with respect to a reference trajectory and optimization parameters. The optimal control history is then the sum of the nominal control history and the optimal control perturbation [6]. By rearranging and simplifying the form of equation (5), a set of matrices is obtained which leads to the unconstrained and constrained optimization problems. For the unconstrained case, Alaniz [6] presents a solution by using an equivalent least squares technique, while for the constrained case, the problem is reinterpreted so as to obtain the standard form handled by quadratic programming solvers.

Once the optimal control history is chosen, the first N time steps of the solution are applied to the plant. The cycle of forming predicted behaviours and solving for the optimal control perturbations is then repeated using the most recent feedback from the plant. The interested reader is referred to [6] for further detail about this technique.

B. A novel numerical linearization technique using PSO

A novel application of PSO proposed here exploits the aforementioned numerical linearization technique used in conjunction with the PSO algorithm, where the convex least squares or quadratic programming optimization methods are now replaced by the global best PSO algorithm. The evaluation function is the cost given by equation (5), so that PSO searches for the optimal perturbed control history of equation (6), denoted by $\Delta \mathbf{u}(k)^*$, in order to obtain the optimal control history $\mathbf{u}(k)^*$ that minimizes J . For this purpose, we require an m -dimensional PSO, with each particle's position defined by \mathbf{K} , an m -dimension column vector equal to $\Delta \mathbf{U}(k)^*$, which is a column vector having $\Delta \mathbf{u}(k+i)^*$ as its elements.

C. A novel PSO-based nonlinear MPC strategy

The second novel controller makes use of the PSO search algorithm for obtaining the optimal control history that minimizes directly the cost function J given by equation (5) without resorting to numerical linearization as represented by equation (6). In this manner we simply use equation (5) as the evaluation function to be minimized using global best PSO, thereby avoiding any linearization technique or mathematical result for minimization, albeit at an increased computational complexity. Each particle's position in the swarm represents the m -dimension column vector defining the optimal control history, $\mathbf{U}(k)^*$.

As we shall see, this remarkably simple method produces the best results for the controllers studied in this paper in terms of the performance index obtained. The block diagram in Figure 1 illustrates the structure of the proposed predictive control loop. A particle swarm optimizer uses the reference input and predicted output trajectories to minimize the quadratic cost function given by equation (5) and compute the optimal control history which is then applied to the plant. The proposed controller is further enhanced by actively

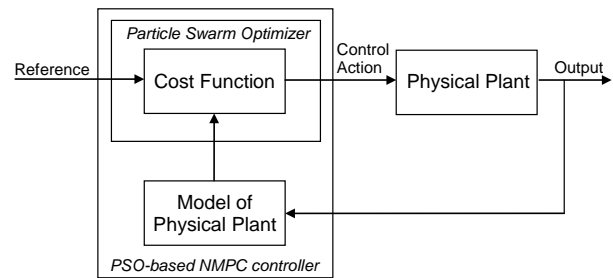


Fig. 1. PSO-based nonlinear MPC loop

correcting the weighting matrix \mathbf{R} in an adaptive manner, so that the chattering effect of the control input observed about the equilibrium point is reduced.

IV. PERFORMANCE EVALUATION: INVERTED PENDULUM ON CART

The performance of the proposed controllers is evaluated by analyzing the results from simulation experiments. The plant chosen for simulation is an inverted pendulum on a cart and two types of controllers are generated for the three methods presented in this paper; an unconstrained and constrained version. The latter problem shall only consider single constraints and therefore no penalty functions are required. The pendulum is initially at the stable equilibrium point and the purpose of each controller is to invert the pendulum. Since the dynamics at the stable and unstable equilibrium points are very different, this is a good problem to demonstrate the effectiveness of our nonlinear MPC controllers.

A. Plant Model

The nonlinear model of the plant is derived by applying Newton's Laws of Motion to the free body diagrams in Figure 2. The resulting equations of motion are given by equations (7) and (8). A complete derivation is given in [6].

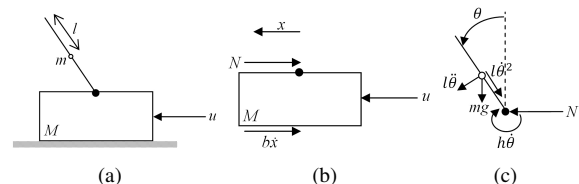


Fig. 2. (a) Inverted Pendulum on a Cart; (b) Free body diagram 1; (c) Free body diagram 2.

$$\ddot{x} = \frac{1}{M+m} [u - b\dot{x} - ml\ddot{\theta} \cos(\theta) + ml\dot{\theta}^2 \sin(\theta)] \quad (7)$$

$$\ddot{\theta} = \frac{3}{4ml^2} [mgl \sin(\theta) - ml\dot{x} \cos(\theta) - h\dot{\theta}] \quad (8)$$

M is the mass of the block that slides along a surface, m is the uniformly distributed mass of an ideal pendulum, $2l$ is the length of the ideal pendulum, b is the surface friction damping constant, h is the rotational friction damping coefficient, u is the force applied to the block, θ is the clockwise angle between the normal and the pendulum (as shown in Figure 2(c)), and x is the cart's horizontal displacement from its

equilibrium position. To allow the model to be numerically integrated, equations (7) and (8) are expressed in terms of the state variables x , \dot{x} , θ , and $\dot{\theta}$. The second-order differential equations are then discretized using the fourth-order Runge-Kutta method [6].

B. Controller Layout

The simulation experiments were run on the Simulink software package [19]. The layout shown in Figure 3 is the simulated realization of the control loop given in Figure 1. It makes use of S-Functions that implement constrained or unconstrained versions of the PSO-based NMPC controller.

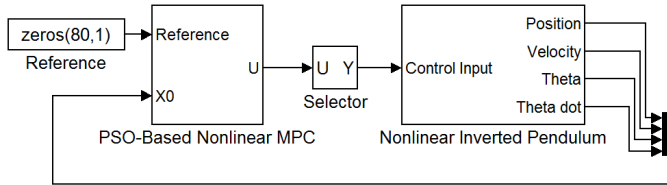


Fig. 3. Nonlinear MPC Simulink layout

C. Controller Parameters

The MPC controller rate is $\frac{1}{NT_s}$, where N is the number of controls in the control history that are applied to the plant. $N = 1$ is used in the controller since this is the typical value selected [18]. The computational load of MPC can be reduced if N is increased, but a disadvantage to having $N > 1$ is that some of the controls applied to the plant are based on old feedback. The fourth-order Runge-Kutta method is tested using different values for T_s , and it is established that the response with $T_s = 0.1s$ is almost indistinguishable from the actual response, thus using this value for the controller.

Since this controller is very computationally intensive, it is not feasible to have a long prediction length or control history. A value of $l = m = 20$ is chosen as a balance between performance and computation time. This results in a controller capable of predicting for 2 seconds.

The two novel PSO-based controllers use the following PSO parameters, which were derived empirically through successive simulations:

- Each particle consists of 20 members, corresponding to the 20 elements that make up the optimal control perturbation history column vector, $\Delta U(k)^*$, for the PSO-based numerical linearization method, or the optimal control history column vector, $U(k)^*$, for the PSO-based NMPC controller.
- Swarm size, $n_s = 30$.
- Inertia weight starting with $w(0) = 0.9$ and linearly decreasing to $w(n_t) = 0.4$.
- Velocity clamped to within half the particles' positional constraints.
- Search space is limited to real-valued variables between $-300N$ ($x_{min,j}$) and $300N$ ($x_{max,j}$) for the unconstrained case.
- Acceleration coefficients $c_1 = 2$ and $c_2 = 2$.
- Number of iterations = 30.

Both weighting matrices Q and R given in equation (5) are set equal to the values shown in equation (9). Deviations are measured from the reference trajectory which is set equal to a zero column vector. There is a zero for each state variable at each time step in the prediction interval. This reference remains constant for the duration of the simulation.

$$Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, R = 1 \quad (9)$$

D. Simulation Results

The response of the unconstrained controller, using the three control schemes described in this paper, is shown in Figure 4 with the pendulum initially at the stable equilibrium point (180 degrees), hanging straight down. The running performance index plot describes the minimized value of the cost function for each time step. The cart's position moves back and forth so that the pendulum gains momentum. This continues until there is enough momentum to swing up and invert the pendulum in the 0 degree position. Table I shows the superior performance obtained for the proposed PSO-based NMPC controller for the unconstrained case. The performance index values quoted here are obtained using equation (5), this time using the actual output trajectory and the applied control inputs, summed over a sufficiently large amount of time (10s). This reveals the true performance index for the whole control action. When PSO is used in conjunction with the numerical linearization technique, no significant advantage is obtained over the least squares method (an improvement in J of only 1.46%), as expected for the convex optimization problem being solved. On the other hand, the second proposed nonlinear PSO controller gives an improvement in J of 8.04% over the numerical linearization (least squares) counterpart.

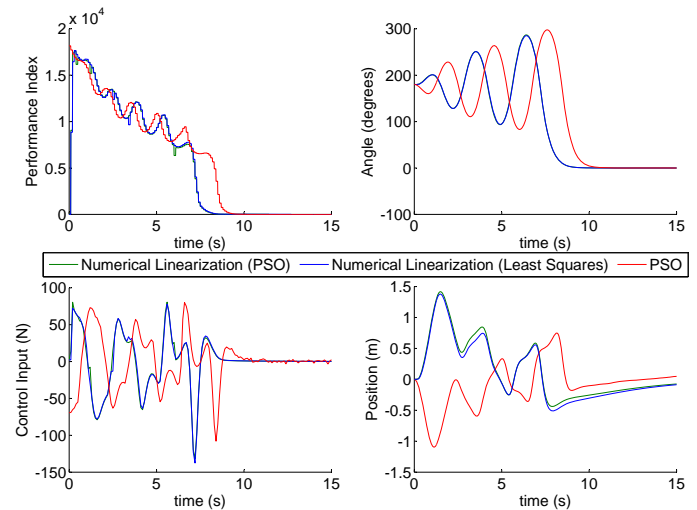


Fig. 4. Unconstrained nonlinear MPC: A comparison

Figure 5 plots the response of the constrained controller when a single constraint, restricting the control input of the

TABLE I
UNCONSTRAINED NMPC: PERFORMANCE INDEX VALUES

Method	Numerical Linearization (Least Squares) [6]	Numerical Linearization (PSO)	PSO
Performance Index $J (\times 10^6)$	1.4538	1.4326	1.3369

cart to be within $-45N$ and $45N$, is active. In Figure 5, the final angular deflection is either 0° or 360° . Note that both these angles correspond to the same inverted position of the pendulum. For the novel PSO-based NMPC controller, the cart is noted to move a much smaller distance to achieve swing-up. In real-world terms, this translates to a more efficient process, with less work being done by the cart to achieve swing-up and equilibrium. This is further evidenced by Table II, which indicates that the novel PSO-based nonlinear MPC controller has the edge over the numerical linearization technique which uses quadratic programming, a method known to have the problem of getting stuck in local minima [20]. We record a 14.78% improvement in J , accompanied by a very low standard deviation when the experiment is repeated over 10 trials indicating PSO's repeatable nature despite being a metaheuristic optimization method. Note that for the constrained case, using the numerical linearization technique in conjunction with PSO is computationally inefficient since every particle must be checked for its corresponding optimal control history, doubling the workload of its unconstrained counterpart, rendering it practically useless to investigate for this purpose. The advantage of the novel PSO-based NMPC

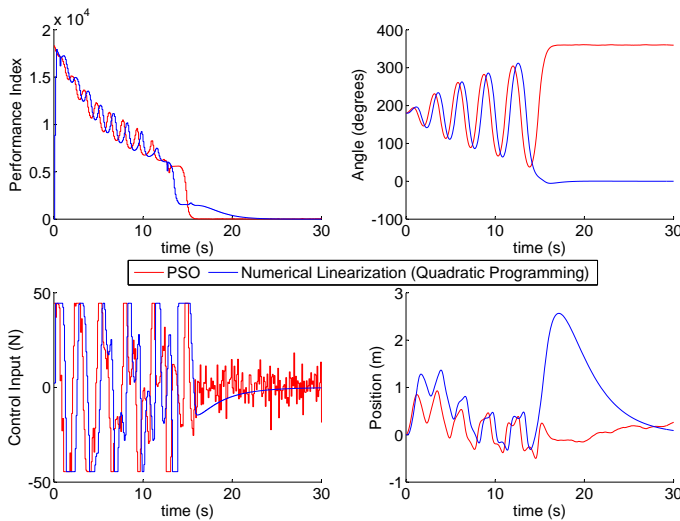


Fig. 5. Constrained nonlinear MPC: Restricting control input to within $-45N$ and $45N$ (10 independent trials)

TABLE II
CONSTRAINED NONLINEAR MPC: PERFORMANCE INDEX VALUES

Method	Numerical Linearization (Quadratic Programming) [6]	PSO (mean J)	PSO (standard deviation) ($\times 10^6$)
Performance Index $J (\times 10^6)$	2.8534	2.4316	0.02396

controller is even more evident in Figure 6, where the proposed active correction for the chattering effect of the control input is implemented for the same constrained NMPC problem by changing R dynamically. The control input is being more heavily penalized when the angle approaches the equilibrium point by increasing R from 1 to 30. In other words, we are telling the system that in the close neighbourhood of the equilibrium point, minimal control effort is required, mitigating the effect of metaheuristic stochasticity. This reduces the performance index even further, giving an improvement in J of 16.65% (see Table III), making the process even more efficient. The system's robustness to model uncertainty is best

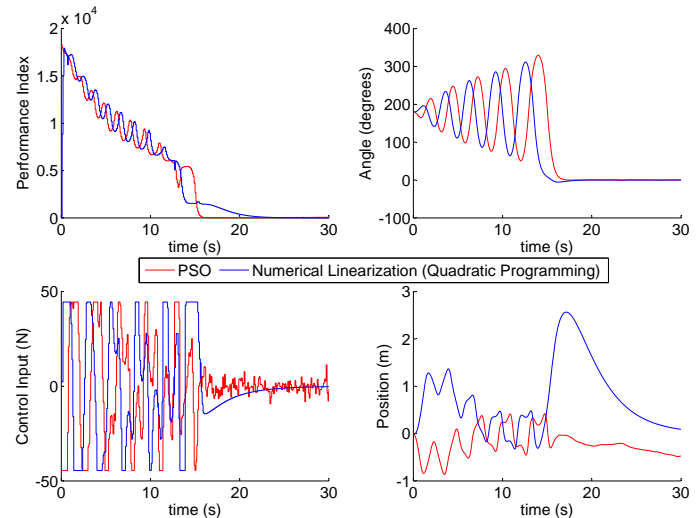


Fig. 6. Actively controlling control input weight R for reduced chattering.

TABLE III
PERFORMANCE INDEX VALUE COMPARISON FOR ACTIVE R CORRECTION

Method	Numerical Linearization (Quadratic Programming) [6]	PSO
Performance Index $J (\times 10^6)$	2.8534	2.3810

illustrated by the simulation results of Figure 7. This is tested by randomly increasing or decreasing each of the plant model's parameters by 5% (all parameters are changed for every trial), so despite the fact that the constrained NMPC controller is using a severely inaccurate model for its predictions, and we are not actively controlling the control input weight R (to consider the worst case), excellent performance is noted, and the pendulum swings up normally, except for a larger distance now required. Table IV shows the corresponding changes implemented in the model's parameters for the simulation results of Figure 7. The corresponding performance index values are given in Table V, where although both controllers manage swing-up and equilibrium similarly as for the results shown in Figure 5, the novel PSO-based NMPC controller exhibits an improvement in J of 12.07%. Repeatability is tested by performing several trials with different constraints, as shown in Table VI. The novel PSO nonlinear controller shows consistently better performance, with a mean improvement in J of 9.17%.

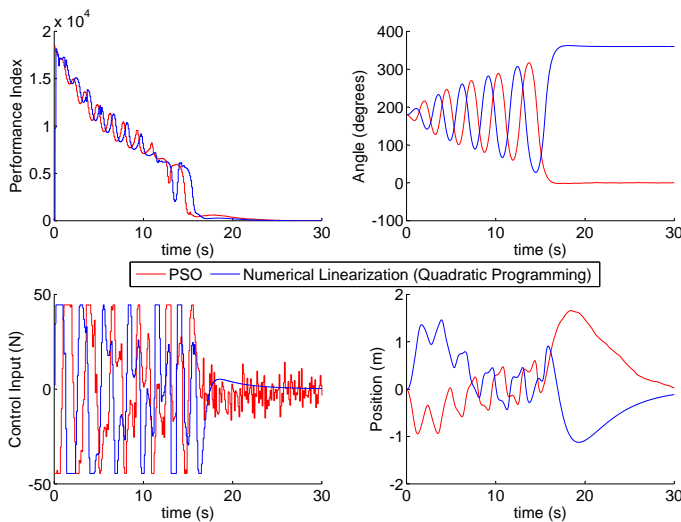


Fig. 7. Robustness Test: Model’s parameters are significantly different from the actual plant parameters (constrained NMPC problem results shown).

TABLE IV

ACTUAL PLANT AND MODEL PARAMETERS (FOR A PARTICULAR TRIAL)

Parameter	Units	Actual Plant	Model
M	Kg	14.6	15.33
m	Kg	7.3	6.935
$2l$	m	2.4	2.52
b	Kg/s	14.6	15.33
h	Kgm^2/s	0.0136	0.0129

TABLE V

PERFORMANCE INDEX VALUES OBTAINED FOR THE ROBUSTNESS TEST

Method	Numerical Linearization (Quadratic Programming) [6]	PSO
Performance Index J ($\times 10^6$)	2.7369	2.4065

TABLE VI

SIMULATION RESULTS FOR DIFFERENT CONSTRAINTS (10 INDEPENDENT TRIALS WITH CONSTANT R)

Constraint	Numerical Linearization (Quadratic Programming) [6]	PSO
$-30N \leq U \leq 30N$	2.7182	2.4026
$-35N \leq U \leq 35N$	2.5374	2.3947
$-40N \leq U \leq 40N$	2.4590	2.3880
$-45N \leq U \leq 45N$	2.8534	2.3810

V. CONCLUSION AND FUTURE WORK

In this paper two novel controllers were proposed for the receding horizon strategy, both exploiting the desirable optimization properties of PSO. One makes use of the numerical linearization technique where instead of convex optimization methods, we employed a PSO strategy. The latter yielded a minor improvement in performance index over its convex optimization counterpart for a simulated inverted pendulum on cart problem. However the second novel PSO-based controller proved superior to both, approaching up to 16% less performance cost at best. In addition, we proposed a further enhancement for this novel controller by actively controlling the control input weight R to reduce the chattering effect of the control input observed for the nonlinear model predictive

controller. Having shown that this framework extends to input constrained systems, we have provided the foundation to include other advances in control theory as they become available. Further work may include investigation of the use of PSO to obtain the much needed connection between the selection of weighting matrices Q and R , and performance specifications, possibly through some time-domain performance criterion. A similar investigation may be carried out for other control schemes, including linear quadratic optimal control strategies.

REFERENCES

- [1] C. V. R. D. Q. Mayne, J. B. Rawlings and P. O. M. Scokaert, “Constrained model predictive control: Stability and optimality,” *Automatica*, vol. 36, no. 6, pp. 789 – 814, 2000.
- [2] E. F. Camacho and C. A. Bordons, *Model Predictive Control in the Process Industry*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1997.
- [3] D. W. Clarke, *Advances in Model-Based Predictive Control*. Oxford University Press, 1994.
- [4] K. R. Muske and J. B. Rawlings, “Model predictive control with linear models,” *AIChE Journal*, vol. 39, no. 2, pp. 262–287, 1993.
- [5] S. Shin and S. Park, “Ga-based predictive control for nonlinear processes,” *Electronics Letters*, vol. 34, no. 20, pp. 1980–1981, oct 1998.
- [6] A. Alaniz, “Model predictive control with application to real-time hardware and a guided parafoil,” Master’s thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA, USA, 2004.
- [7] X. Blasco, M. Martinez, J. Senent, and J. Sanchis, “Generalized predictive control using genetic algorithms (gagpc). an application to control of a non-linear process with model uncertainty,” in *Methodology and Tools in Knowledge-Based Systems*, ser. Lecture Notes in Computer Science.
- [8] T. Kawabe and T. Tagami, “A real coded genetic algorithm for matrix inequality design approach of robust pid controller with two degrees of freedom,” in *Intelligent Control, 1997. Proceedings of the 1997 IEEE International Symposium on*, jul 1997, pp. 119 –124.
- [9] R. Krohling, H. Jaschek, and J. Rey, “Designing pi/pid controllers for a motion control system based on genetic algorithms,” in *Intelligent Control, 1997. Proceedings of the 1997 IEEE International Symposium on*, jul 1997, pp. 125 –130.
- [10] P. Angeline, “Using selection to improve particle swarm optimization,” in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, may 1998, pp. 84 –89.
- [11] D. B. Fogel, *Evolutionary computation: toward a new philosophy of machine intelligence*. Piscataway, NJ, USA: IEEE Press, 1995.
- [12] R. C. Eberhart and Y. Shi, “Comparison between genetic algorithms and particle swarm optimization,” in *Proceedings of the 7th International Conference on Evolutionary Programming VII*, ser. EP ’98. London, UK: Springer-Verlag, 1998, pp. 611–616.
- [13] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, nov/dec 1995, pp. 1942 –1948 vol.4.
- [14] Y. Shi and R. Eberhart, “A modified particle swarm optimizer,” in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, may 1998, pp. 69 –73.
- [15] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Nakanishi, “A particle swarm optimization for reactive power and voltage control considering voltage security assessment,” *Power Systems, IEEE Transactions on*, vol. 15, no. 4, pp. 1232–1239, nov 2000.
- [16] A. P. Engelbrecht, *Computational Intelligence: An Introduction*, 2nd ed. Wiley Publishing, 2007.
- [17] J. J. E. Slotine and W. Li, *Applied Nonlinear Control*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1991.
- [18] J. Maciejowski, *Predictive control: with constraints*. Prentice-Hall, Harlow, UK, 2002.
- [19] The MathWorks, Inc. (2011) Simulink - simulation and model-based design. [Online]. Available: <http://www.mathworks.com/products/simulink/>
- [20] S. Boyd and L. Vandenberghe, *Convex Optimization*. New York, NY, USA: Cambridge University Press, 2004.

Extending Microsoft® Project for Real-World Job-Shop Scheduling with Genetic Algorithm

Peter Steininger

Steinbuch Center for Computing (SCC)
 Karlsruhe Institute of Technology (KIT)
 Kaiserstrasse 12
 D-76128 Karlsruhe, Germany
 steininger@KIT.edu

Abstract — Although production scheduling has attracted the research interest of production economics communities for decades, there still remains a gap between academic research and real-world problems. Genetic Algorithms (GA) constitute a technique that has already been applied to a variety of combinatorial problems. We will explain the application of a GA approach to bridge this gap for job-shop scheduling problems, for example to minimize makespan of a production program or to increase the due-date reliability of jobs. The presented approach focuses on integrating a scheduling algorithm, based on GA, into a commercial software product, namely Microsoft Project 2003. We extended Microsoft Project in a range of aspects: A new graphical user interface is introduced to support users by a guided wizard describing the problem for which an optimal schedule is sought. The GA was developed to search for the solution with the maximum results for a given set of production logistical objectives. The developed GA algorithm and operators are tested on real-world data from a one-of-a-kind manufacturing department of a major company. It includes different aggregation operators for combining objectives. Furthermore, the efficiency of the algorithm was compared to benchmark tests available in literature.

Keywords: Job-Shop Scheduling, Genetic Algorithms, Job-Shop Benchmarks, Real-World Scheduling Problems

I. PROBLEM STATEMENT

A. Characteristics of job-shop scheduling problems

Many jobs in industry and elsewhere require a collection of tasks or activities to be completing while satisfying temporal, resource and precedence constraints. Temporal constraints impose that some activities, or a set of them have to be started or finished before or only after a certain point in time. Resource constraints dictate that two tasks requiring the same resource cannot be carried out simultaneously. Precedence constraints refer to the technological winding-up of a job. The objective is to create a schedule specifying when each task is to begin (or finish) and what resources it shall use in order to satisfy all the constraints while pursuing an objective, e.g., taking as little overall time as possible, minimizing mean delay, minimizing maximum delay, minimizing the number of late jobs and so on. This is known as the job-shop scheduling problem (JSP).

The JSP is a very important and well-studied scheduling problem. It is a basic model, which may be extended by

additional characteristics like buffers, transportation, setup time, time lags, etc., allowing practical scheduling problems in practice are to be modeled more precisely. In its general form, it is \mathcal{NP} -complete, meaning that there is probably no efficient procedure for exactly finding the shortest schedule for arbitrary instances of a problem.

Bagchi [1, p. 109] references the JSP as follows: "Within the great variety of production scheduling problems that exist, the job shop scheduling problem (JSP) is one that has generated the largest number of studies. It has also earned a reputation for being notoriously difficult to solve. Nevertheless, the JSP illustrates at least some of the demands imposed by a wide array of real world scheduling problems... Attempts to tackle the multi objective job shop are still relatively few." A JSP is usually solved using a heuristic algorithm that takes advantage of special properties of a specific class of instances. This can be regarded as a backdoor to reducing the complexity of a given problem.

B. Formal problem description

An instance of the JSP consists of a set of NOJ jobs i and NOM machines j . Each job consists of a number of activities so that we can count the total number of activities NOA as follows:

$$NOA = \sum_{i=1}^{NOJ} NOJ_i \quad (1)$$

Each job has fixed number and sequence of activities. Each activity has certain duration and requires a single machine for its entire duration. The activity following a preceding one within a job requires a different machine. An activity must be finished before each activity following it in its job. Two activities cannot be scheduled at the same time if they both require the same machine. The problem is to find a feasible schedule which minimizes some objective function, e.g., minimizing makespan, in other words the overall completion time of all activities, see Steininger [15, p. 26 f.]. These results in a complexity function for the JSP expressed as

$$O\left((NOJ!)^{NOM}\right) \quad (2)$$

which means in order to find the best schedule for a problem instance, we have to enumerate and evaluate all possible

schedules and the number of possible schedules to enumerate is the result of function (2).

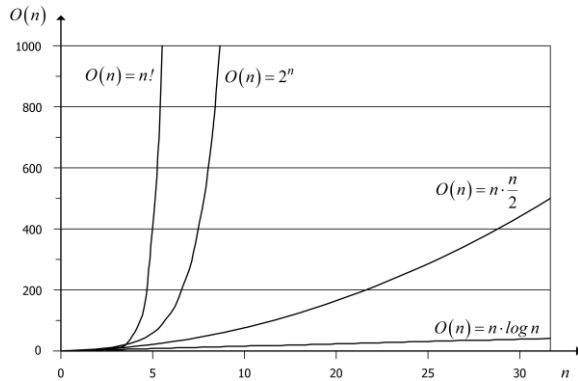


Figure 1. Selected complexity functions

Figure 1 illustrates the dimension of selected complexity functions, where n is the number of problem elements, here the number of activities. The graph illustrates that the complexity of JSP is much bigger than some other well-known problems, such as "Permutation problem" which is $O(n) = n!$, "Towers of Hanoi" which is $O(n) = 2^n$, "Quicksort" which is $O(n) = n \cdot \log n$, and so on.

C. 1.3 Classification of scheduling problems

Classes of scheduling problems can be specified in terms of the three-field classification approach initial introduced by Conway, Maxwell and Miller [5] and extended by Graham [12] and Brucker [3], which is under a continuous reconsideration.

The three-field classification is described as $\alpha|\beta|\gamma$, where α specifies the machine environment, β specifies the job characteristics and γ describes the objective function or a combination of objective functions. Using the three-field classification to specify the problem instance of the JSP we are examining, the following taxonomy is noted:

$$J, NOM | NOJ, \text{intree}, t_{ij} | C_{\max} \quad (3)$$

Formula (3) describes a class of scheduling problems as JSP (J) with a fixed machine count of NOM and a predefined and fixed number of NOJ jobs. The order of activities in each job is predefined and fixed as a directed graph with operation times (t_{ij}), expresses as integer values, for each task.

The three-field classification notes γ as the objective function or a combination of objective functions. In formula (3) γ specifies the "traditional" objective function (C_{\max}), which de-scribes our goal as taking as little makespan as possible for the schedule of all NOJ jobs using NOM machines.

II. MODELLING OF JSP SCHEDULING PROBLEMS

A. Formal problem representation

Even slightly different job-shop problems require completely different encodings in order to find a good solution.

Thus, choosing a good representation is a very important component of solving a JSP. However, choosing a good representation for a scheduling problem is as difficult as choosing a good search algorithm for a search problem. Not all algorithms work equally efficient on a specific problem representation. To describe the representation technique developed for our solution we use a simple job-shop scheduling problem as shown in Table I.

Job j	Machine S_j	t_{ij} [TU]		
		1 _{i}	2	3
1	(3,2,1)	3	5	1
2	(1,2,3)	3	2	1
3	(2,1,3)	1	2	5

Table I. Example of a production schedule problem with 3 jobs, routing information S_j for jobs, 3 machines and task operation times.

The scheduling problem can be represented by a graph as shown in Table I. In addition to the activity nodes (j, i), it contains a source node a with no duration (operation time), a sink node e , also with no duration (operation time), and two nodes called r_2 and r_3 which describe an imposed later start of job 2 and 3 relative to job 1.

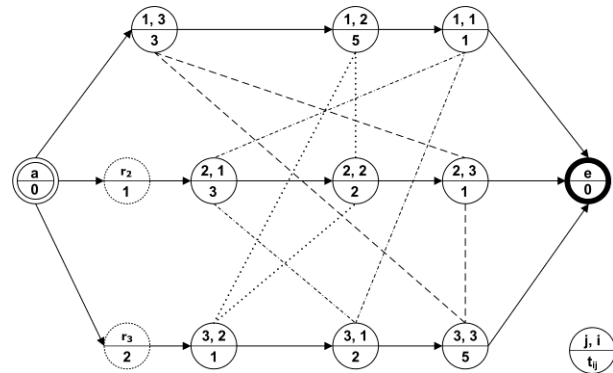


Figure 2. Network representation of a JSP based on Table I

The directed arcs running from the source node, through each activity node (j, i) to the sink node describes the technological sequence of activities based on S_j in Table I. Each node shows the job id, the machine needed and the operation time t_{ij} . There are also undirected arcs in the network, which references all possible sequences of an activity of a given job on a specific machine. Such a representation is called a disjunctive network.

B. Data representation and problem reduction

Care must be taken when adopting such a graphical representation into a data structure, especially for the JSP in an area with hundreds of jobs, thousands of sequences and millions of possible activity orders at specific machines.

A data structure which is very efficient in the use of storage (because of the size of a practical problem) as well as in

time has to be found to represent the introduced network. Gallo and Pallottino [10] introduced the so-called "Forward Star" data structure, which is the most efficient representation of all existing network data structures for representing a network [4]. The "Forward Star" data structure uses three arrays to describe a (directed) network. First we have an array called *from*, whose index represents all nodes of the network. The value of an index field references the index of the second array called *succ*, which is the index of a node to connect, referencing *from*. The third array is called *cost* and reports the cost of a specific arc connecting two nodes.

The "Forward Star" data structure allows for a perfect implementation of the activity order of any JSP, an efficient implementation in storage and time and a reduction of the initial problem described by the $\alpha | \beta | \gamma$ three-field classification. Using the "Forward Star" data structure our problem is reduced to the following taxonomy ():

$$J, NOM | NOJ, chains, t_{ij} | C_{max} \quad (4)$$

As mentioned above, the selection of a good representation is very important for the solution of a JSP. Care must also be taken to adopt both representational schemes and the associated operators for an effective algorithm. When using the traditional way of solving a problem with GAs, the chromosomes are implemented as binary vectors. This simple representation is an excellent choice for problems in which a point naturally maps into a chromosome of zeros and ones. Unfortunately, this approach of zeros and ones cannot be used for real-word engineering problems such as JSP, because of the number of information needed to represent coding of the JSP. Therefore, we have to find a way to integrate the "Forward Star" data structure into a GA.

III. GENETIC ALGORITHMS

A. General principle

The term Genetic Algorithm describes a set of methods, which can be used to optimize complex problems. As the name suggests, the processes employed by GAs are inspired by natural selection and genetic variation. This GA uses a population of possible solutions to a problem and applies a cycle of processes to modify them. These processes mimic those in nature in such a way that subsequent populations are fitter and more adapted to their environment. As generations progress over time, they become better suited to their environment and provide an advantageous solution in a given time.

Since their development in the late 1980's GAs [11] have been used to find solutions too many types of problems. A unique characteristic of a GA is that the fundamental algorithm is unaware of the problem it is optimizing. All that is required is that the parameters entered into a model or system can be efficiently transformed to and from a suitable GA chromosome format. This means GA optimization can be applied to many types of complex problems. Detailed introductions are given by Goldberg [11] and Davis [7].

The flowchart for the GA is given in Figure 3. First, an initial population of randomly generated sequences of the

tasks in the schedule is created. These individual schedules form chromosomes which are subject to a form of evolution. Once an initial population has been formed, "selection", "crossover" and "mutation" operations are performed repeatedly until the fittest member of the evolving population converges to an optimal fitness value. Alternatively, the GA may run for a user-defined number of iterations [11].

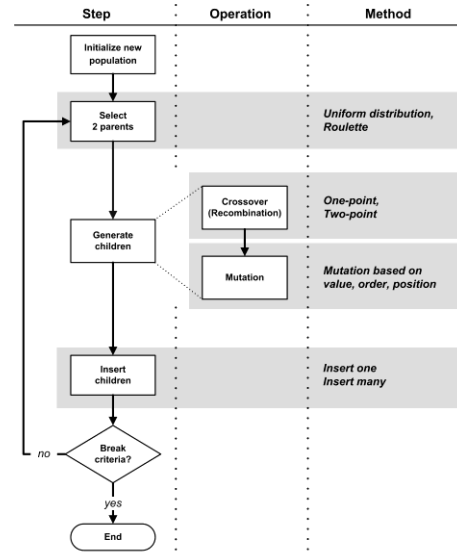


Figure 3. Principal flow of Genetic Algorithms.

The size of the population is user-defined and the fitness of each individual, in this case a schedule, is calculated according to a function, in our case the makespan or an additive combination of different goals. It is also possible to use a fitness function on other calculated values like mean or maximum delay, number of delayed jobs and so on. Also combinations of different functions are possible. The schedules are then ranked according to the value of their fitness function and, after that, selected for reproduction.

B. Schedule encoding and decoding

GAs were derived by examining biological systems. In biological systems evolution takes place on chromosomes which are organic devices for programming the structure of living beings. In this sense, a living being is a decoded structure of all chromosomes. Natural selection is the link between chromosomes and the performance of the decoded structures. When implementing the GA, the variables that characterize an individual are represented in arrays (by index ordered lists). Each variable corresponds to a gene and the array corresponds to a chromosome in biological systems.

Decision was made [15] to use the encoding schema introduced by Bean [2] to build the chromosomes. He calls his schema "Operation Based Representation". Encoding starts by enumerating jobs and corresponding activities in a list. Each activity in a job is encoded with the numerical id of the job in which it resides. All jobs and activities are encoded following that description in one potential schedule for the problem. The result is a chromosome which represents a potential schedule.

C. Crossover

The GA uses crossover, where mating chromosomes are cut once. Crossover is the most delicate operation of GA because of the problem that it can produce irregular activity sequences for a job. We use a corrected 2-point crossover to avoid non-regular activity sequences of orders, which Goldberg [11] refers to as a PMX-crossover operator. The following figures will illustrate a crossover operation of a sample JSP with 4 jobs, each with 3 activities, and 4 machines (see Figure 4 to Figure 8), based on an example in Steininger [15, p. 146 f.].

Step 1: Select two individuals randomly from the population (Figure 4).

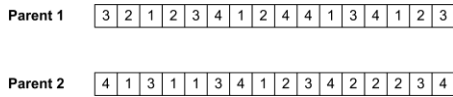


Figure 4. Crossover operator, step 1: Parent selection.

Step 2: Select a segment of the first chromosome which starts and ends with the same job number (Figure 5). Selected segment: 4124.

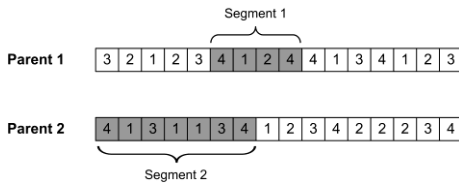


Figure 5. Crossover operator, step 2: Segment selection.

Step 3: Select a segment of the second chromosome which starts and ends with the same job number as the selected segment of the first chromosome (Figure 6). Selected segment: 4131134.

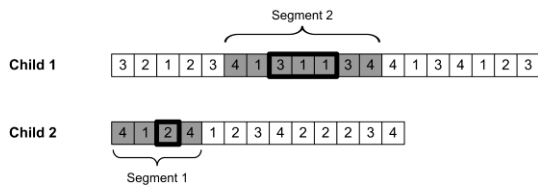


Figure 6. Crossover operator, step 4: Segment exchange.

Step 4: Exchange the selected segments between the two chromosomes to get the "child". The result is two chromosomes with non-regular activity sequences of jobs. Child 1 has too many activities and child 2 lacks some genes/activities (Figure 7). This result necessitates a correction step.

Step 5: The following process, called "normalization", initializes this correction step. It examines the original segment of a child with the exchanged segment of the same child (Figure 7). The result of that examination for child 1 is: 2 are missing; 3,

1, 1 and 3 are added. For child 2 we get: 3, 1, 1 and 3 are missing and 2 is added.

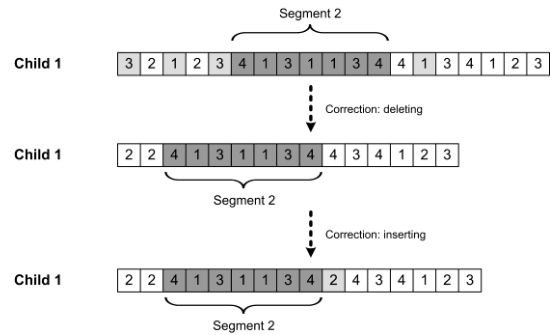


Figure 7. Crossover operator, step 5: Correction of child 1.

Step 6: Having detected the added and missed genes, we can start the correction step: For child 1 we delete 3, 1, 1 and 3 at first occurrence in child 1 without inspecting the exchanged segment; for child 1 we add 2 at the end of exchanged segment (Figure 8). The same operations are executed on child 2 and, at the end of all steps; we have two new and correct chromosomes.

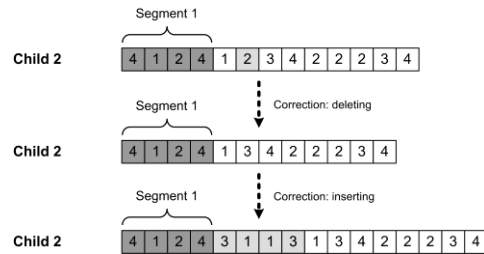


Figure 8. Crossover operator, step 6: Correction of child 2.

D. Mutation

Mutation is the process of random dissimilarity of the value of a gene with small probability. It is not a primary operator, but it ensures that the possibility of searching any section in the problem space is never zero and prevents complete loss of genetic material through reproduction and crossover. We execute the mutation operator as a permutation by first picking (and deleting) a gene before reinserting it at a randomly chosen position of the permutation (Figure 9).

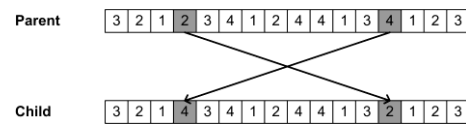


Figure 9. Mutation operator: Gene flipping.

E. Fitness

The fitness function is used to evaluate the fitness of each individual in the population and depends on the specific application. Since a GA proceeds toward more fit individuals and the fitness value is the only information available to the GA, the performance of the algorithm is highly sensitive

to the fitness function. In case of optimization routines, the fitness is the value of the objective function to be optimized.

F. Selection

To selectively reproduce the population and to determine the next generation we use a hit and miss selection procedure based on the fitness function. This could be implemented using a roulette wheel method. An imaginary roulette wheel is constructed with a segment for each individual in the population. An individual's section size is based on the fitness value of the particular individual. A fit individual will occupy a larger slice of the roulette wheel than a weaker one. Selection is made by rotating the roulette wheel a number of times equal to the population size. When the roulette wheel stops, the individual it points to is selected. This means that fitter individuals will have a propensity to be selected more frequently than weaker ones.

The GA needs a few additional parameters to work. These parameters specify the size of the population, use of operators and so on.

G. Population size

The population size depends on the nature of the problem [13]. Typically, it contains several hundreds or thousands of possible solutions. The population is generated randomly, so it is possible to cover the entire range of possible solutions. We use a population size of 500 individuals, which represents 500 possible schedules.

H. Probability of crossover

The parameter probability of crossover affects the rate at which the crossover operator is applied [11]. A higher crossover rate introduces new chromosomes more quickly into the population. If the crossover rate is too high, good individuals are eliminated faster than selection can produce improvements. A low crossover rate may cause stagnation due to the lower exploration rate. We use probability of crossover with a value of 0.6.

I. Probability of mutation

Probability of mutation is the likelihood with which each gene of each individual in the new population undergoes a random change after a selection process. A low mutation rate helps to prevent any gene positions from getting stuck to single values, whereas a high mutation rate results in essentially random search [11]. We use a value of 0.05 for mutation probability.

J. Final result

It is a characteristic of the GA that once fairly good solutions have been found their features will be carried forward into even better solutions, which will ultimately lead to a near-optimal solution. Therefore, GAs are particularly attractive for scheduling.

Compared with other optimization methods, GAs are suitable for traversing large search spaces since they can do this relatively rapidly and because the mutation operator diverts the method away from local minima. Being suitable for large search spaces is a useful advantage when dealing

with schedules of increasing size since the solution space will grow very rapidly. It is important that these large search spaces are scanned as fast as possible to enable the practical and useful implementation of schedule optimization.

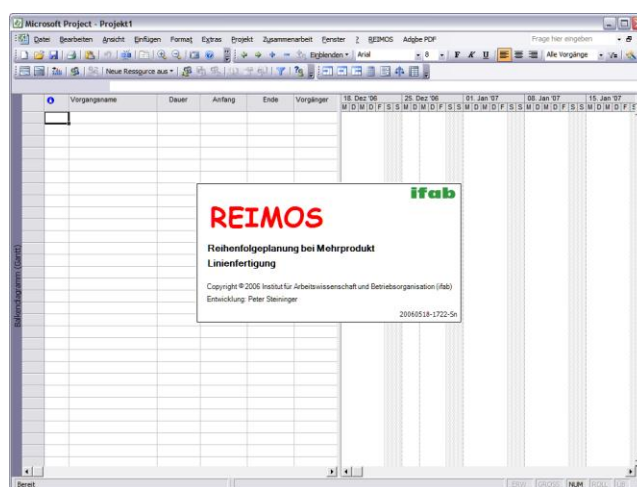


Figure 10. Microsoft Project 2003 with integrated REIMOS.

IV. APPLICATION EXAMPLES

A. Scheduling problem in a one-of-a-kind manufacture

We have tested our approach, *REIMOS* (German abbreviation for "Sequence planning for multi-product manufacturing systems") [15], in a one-of-a-kind manufacture of a major German company. For confidentiality reasons, the model mix, job and operation data are under a non-disclosure agreement and we are not allowed to publish the data. But we have also tested our approach with a few publicly available benchmark sets for JSP, which can be acquired by any researcher from a public website (Taillard [16], e.g. Figure 12).

B. Benchmark tests

Early on, research of scheduling problems started a competition on the "best schedule" of a specific problem. For that reason some researchers designed (calculated) very hard to solve scheduling problems as so called "benchmark instances" [9]. Some modern benchmark instances are distributed by Taillard [16] and called JSP-15-15, JSP-20-15 and JSP-50-20. The name of the benchmark is derived from scheduling a problem; in the case of JSP-15-15 it stands for 15 jobs on 15 machines and so on (Figure 11). Taillard [16] also published the list of best results for each scheduling problem instance, allowing our own results to be compared with those of other researchers.

We used three benchmark instances to model Microsoft Project 2003 files as input for *REIMOS* and started a GA run with the parameter values mentioned above (3.8, Figure 11 & 12). Our algorithm for benchmarking was modified to write a so-called "debugging output", so we know specific values for the GA at every step. For example, these values are: number of iterations, best calculated schedule and so on. We compared the best schedule found thus far with the best known schedule of Taillard [16] and calculated a quality

level, which is the percentage reached of so far "best" known solution by Taillard [16]. This is shown in Figure 12 for JSP-15-15.

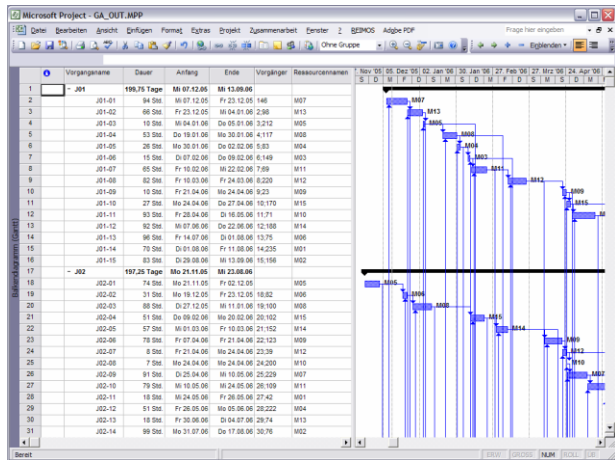


Figure 11. Benchmark instance JSP-15-15 in Microsoft Project 2003.

For all benchmark instances we attained a quality level around 95 % of the best known solution calculated over 1000 generations GA runtime. On a so-called standard PC suitable for the use of Microsoft Office, one calculation run around 10 to 20 minutes, which were our time goal for planning a schedule. We could have gone even further and let it run for hours or days, but the effect would not bring a realistic benefit. The problem with the best known solution by Taillard [16] is that we did not know its runtime, implementation of algorithm, computer etc. So it was hard to say "how good was best" from an economical point of view.

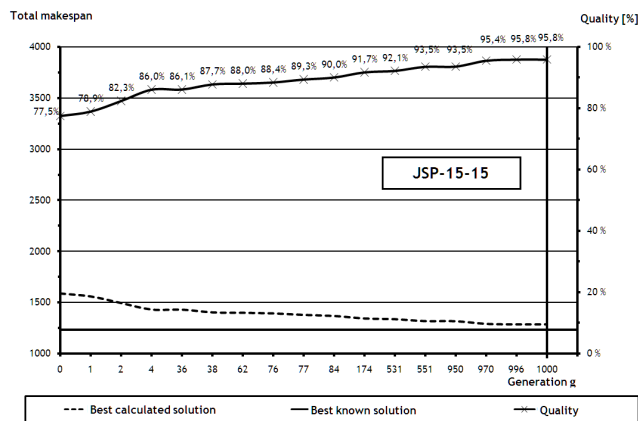


Figure 12. Results of benchmark instance JSP-15-15 with REIMOS.

V. CONCLUSION AND OUTLOOK

A computer algorithm being based on the evolution of living beings may be surprising, but the extensiveness with which this approach is applied in so many areas is even more surprising. Genetic algorithms have already proven their efficiency in many application areas, commercial, educational and scientific. Their usefulness in solving various kinds of problems have made them a preferable choice compared to traditional, mainly heuristic approaches.

The adaptation of a GA to schedule jobs in manufacturing shops with time, resource and precedence constraints has been demonstrated here [15]. The simplicity of the methods used supports the assumption that GA can provide a highly flexible and user-friendly solution to the JSP. The use of standard software and an implemented "add-in" for Microsoft Project 2003 to realize the GA has shown that this approach can be used for solving real industrial scheduling problems [15].

VI. REFERENCES

- Bagchi, Tapan P., 1999. Multiobjective Scheduling by Genetic Algorithms. Boston, Dordrecht, London: Kluwer Academic Publishers.
- Bean, James C., 1994. Genetic Algorithms and Random Keys for Sequencing and Optimizations. ORSA Journal of Computing. 6 (2), 154-160.
- Brucker, Peter, 2004. Scheduling Algorithms. Berlin, Heidelberg, New York et al.: Springer Verlag.
- Cherkassky, B. V., Goldberg, A. V., Radzik, T., 1993. Shortest Paths Algorithms: Theory and Experimental Evaluation. Technical Report 93-1480, Computer Science Department, Stanford University.
- Conway, Richard W., Maxwell, William L., Miller, Louis W., 2003. Theory of Scheduling. Mineola (NY): Dover Publications.
- Darwin, Charles, 1859. On the Origin of Species by Means of Natural Selection. London (UK): John Murry.
- Davis, Lawrence, 1996. Handbook of Genetic Algorithms. Florence (KY): International Thomson Computer Press.
- Domschke, Wolfgang, Scholl, Armin, Voß, Stefan, 1997. Produktionsplanung. Berlin, Heidelberg, New York et al.: Springer Verlag.
- Fisher, Howard, Thompson, Gerald L., 1963. Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules. In: Muth, John F.; Thompson, Gerald L., (Eds.). Industrial Scheduling. Englewood Cliffs (NJ): Prentice-Hall, 225-251.
- Gallo, Giorgio, Pallottino, Stefano, 1982. A new Algorithm to find the Shortest Paths between all Pairs of Nodes. Discrete Applied Mathematics. Amsterdam, 3 (4), 23-25.
- Goldberg, David E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. München: Addison Wesley.
- Graham, Ronald L., Lawler, Eugene L., Lenstra, Jan Karel et al., 1979. Optimization and approximation in deterministic sequencing and scheduling. Annals of Discrete Mathematics, 16 (5), 287-326.
- Haupt, R. L., 2000. Optimum population size and mutation rate for a simple real genetic algorithm that optimizes array factors. IEEE Antennas and Propagation Society International Symposium, 2, 1034-1037.
- Roy, Bernard, Sussmann, B., 1964. Les problèmes d'ordonnancement avec contraintes disjonctive. Montrouge (F): SEMA Groupe.
- Steininger, Peter, 2007. Eine Methode zur Reihenfolgeplanung bei Mehrprodukt-Fertigungssystemen. Dissertation, Universität Karlsruhe. Aachen (D): Shaker Verlag.
- Taillard, Éric D., 2006. Scheduling instances. <http://ina2.eivd.ch/Collaborateurs/etd/problemes.dir/ordonnancement.dir/ordonnancement.html>.

An Analysis of MOSIX Load Balancing Capabilities

Siavash Ghiasvand, Ehsan Mousavi Khaneghah, Sina Mahmoodi Khorandi, Seyedeh Leili Mirtaheri, Najmeh Osouli Nezhad, Meisam Mohammadkhani, and Mohsen Sharifi

School of Computer Engineering
Iran University of Science and Technology
Tehran, Iran

ghiasvand@comp.iust.ac.ir, emousav@iust.ac.ir, sina_mahmoodi@comp.iust.ac.ir, Mirtaheri@iust.ac.ir, {n_osouli, m_mohammadkhani}@comp.iust.ac.ir, and msharifi@iust.ac.ir

Abstract—*Mosix has long been recognized as a distributed operating system leader in the high performance computing community. In this paper, we analyze the load-balancing capabilities of a Mosix cluster in handling requests for different types of resources through real experiments on a Mosix cluster comprising of heterogeneous machines.*

Keywords—*Mosix Load balancing; CPU-intensive process; I/O-intensive process; IPC-intensive process; Memory-intensive process.*

I. INTRODUCTION

Mosix used to be a pioneer distributed operating system for cluster computing. It was built as an extension to the UNIX operating system kernel and provided a single system image to applications. Using Mosix, developers could build SMP machines by clustering a number of dispersed homogeneous machines running under UNIX.

Load balancing the computational power of clustered machines is a well-known mandatory requirement to the provision of single system image and performance (i.e., response time) in homogeneous clusters [1]. Mosix has used a decentralized and probabilistic approach to balance the computational loads on clustered off-the-shelf machines. It has used a special mechanism for scalable dissemination of load information too. To prevent process or system thrashing, it has also used a process reassignment policy through sharing of memory.

In this paper, we intend to show experimentally the capabilities of Mosix in balancing the loads on a Mosix cluster comprised of heterogeneous machines. We have designed and ran a number of tests to investigate if Mosix achieves its acclaimed capabilities in balancing the load on the cluster when requests for different cluster resources are called.

The rest of paper is organized as follows. Section 2 presents a brief description of the Mosix load-balancing mechanism. Section 3 presents our test programs and the results of running them on a 5-node Mosix cluster, and Section 4 concludes the paper.

II. MOSIX LOAD BALANCING MECHANISM

The load balancing mechanism of Mosix works in a decentralized manner using load information of the clustered machines. Process is the basic unit of load balancing in Mosix. It includes appropriate mechanisms such as process

migration and remote system call for adaptive load balancing. It reassigns processes when it decides to balance the load between two nodes [2].

Mosix load balancing mechanism is based on three components, load information dissemination, process migration, and memory information dissemination. Mosix can assign a new process to the best machine as well as balancing the load on entire cluster [3]. In the remaining parts, we describe load dissemination and balancing mechanisms and our experiments written in C language based on process behaviors.

Mosix employs CPU load, memory usage, and IPC information to calculate machine's load and process needs. Indeed these are load indices in Mosix. Mosix monitors these resources, gathers their information and status, and packs them into a message [3], [4]. It then sends the built message to some other machines. Mosix uses depreciation to calculate load information. When a new index is calculated, Mosix changes it with respect to the depreciation rate. With this idea, Mosix employs a history-based approach to balance the load. Mosix monitors resources many times in a second. At the end of each second, Mosix normalizes indices and sends them into two randomly selected machines in the cluster. Mosix stores information about a few numbers of machines due to scalability reasons. This limited number is called information window. When Mosix gathers information about local resources, it selects two machines, one from its window and the other from non-window machines. This mechanism makes Mosix scalable. The main challenge with this mechanism is what size of window is suitable. Simulations show that if there is N machines in the cluster, a window size equal to $\log N$ is suitable [3], [5].

One of the major drawbacks of Mosix information dissemination is its periodic approach. The periodic information dissemination can result in waste of network bandwidth and CPU time. On the other hand, if there are many changes in indices during a period, it will result in unsuitable information. Event-driven information dissemination solves these problems [6].

As mentioned earlier, Mosix employs CPU load, memory usage and IPC information to balance and distribute load among machines belonging to a cluster [3]. In the following, we first describe the Mosix basic load balancing mechanism and then discuss its memory sharing and IPC optimization. In general, a load-balancing algorithm must provide four steps: (1) indices computation (2) information dissemination

(3) load balancing trigger, and (4) load balancing operation. In step four, load balancer must decide on migration destination and a candidate process.

CPU load balancing in Mosix operates based on the amount of available CPU for each process. Mosix computes the amount of CPU time taken by each process. It counts the number of executable processes in a period (one second) and normalizes it with CPU speed in favor of maximum CPU speed in the cluster. Then it changes the index in favor of depreciation to realize actual load based on load history. Gathered information is disseminated between some machines. The Mosix load balancing mechanism is triggered when the difference between a local machine's load and the windows machines' loads exceed a threshold. Load balancing pair contains a machine with lower load index that has enough memory space to host a migrant process. Selecting a migrant process is an important stage of load balancing operation. CPU-intensive processes have priority for migration. Mosix limits the number of times a process is allowed to migrate in order to prevent starvation. In this paper we have designed three types of processes, CPU-intensive, memory-intensive and IO-intensive [1], [3], [5], [7].

Thrashing is a phenomenon in operating systems that occurs by the growth of page faults. When free memory in the system is decreased, processes requiring memory to bring in their pages into memory, page fault. Mosix migrates a process if free memory size drops below a threshold. Indeed, this algorithm distributes processes in the memory of the entire cluster. Although, this algorithm results in unbalanced CPU load, but it can increase performance when system into thrashes [3], [4], [8].

When free memory size drops below a threshold, memory sharing algorithm is triggered. When free memory size drops below this threshold, Mosix expects increases in page faults. Therefore, determination of this threshold is a crucial decision. Target machine at least must satisfy two conditions. First, target machine must have enough space to host a new process. Second, target machine must be in window. Mosix usually selects a target machine with the least CPU load. It selects a process with the least migration cost that has caused memory overload. Migration of the selected process must increase free memory size up to threshold and migration must not overload the target machine . If there is no such process, Mosix selects a larger process that can be placed at the target machine. After this replacement, if there is still memory overload, Mosix repeats the above two steps. We have designed a program for this algorithm [3], [8].

The main goal of balancing load based on IPC is to reduce the communication overhead between processes while keeping the load balanced as much as possible. CPU load balancing tries to utilize all processors equally while keeping communicating processes together results in lower communication cost. Therefore, it would reduce response time in the presence of load balancing if processes have low communication. However, in real science applications, communication between processes is high. Therefore, balancing the load based on IPC information can lower the

response time. Mosix employs a producer consumer model to optimize its load-balancing algorithm. In this model, each consumer tries to find a producer that presents its product with lower cost. In the cluster environment, products are resources such as CPU cycles, memory and communication channels. Consumers are processes residing on machines. The cost of a resource is the amount of time that a process spends to use one unit of that resource. Whereas a process can run on another machine with lower cost, it is migrated to that machine [3], [9].

Mosix computes the cost of each process based on CPU load, free memory space, and IPC information. It uses a heuristic to compute an approximate response time. The algorithm is initiated when the cost difference between running the process on the current machine and running it on one of the window machines exceeds the migration cost. Mosix selects a process with maximum difference and a machine with the least cost as the target machine [3].

Mosix measures performance in terms of speed of CPU in GHz unit. It also measures the load of each node by counting the number of processes in each scheduling time and computes the primary load as the average of counted processes. It then normalizes the load relative to CPU speedup. A load unit represents 100% utilization of a 'standard' processor [2], [3]. We use these measurements in reporting the results of our experiments in Section III.

III. EVALUATION

We have deployed a 5-node cluster of openMosix to analyze the behavior of Mosix in balancing the load on the nodes of the cluster. Each node of cluster had direct access to other nodes through an 8-port 10/100 Ethernet switch. All nodes ran openSUSE 11.2 as their operating system and were equipped with LAM/MPI. Table 1 shows more details of the testbed.

TABLE 1 SPECIFICATION OF THE DEPLOYED OPENMOSIX CLUSTER

System ID	CPU	Memory	OS	MPI
Node1	Intel1.7GHz	256MB	openSUSE11.2	LamMPI
Node2	IntelCeleron 2.4GHz	256MB	openSUSE11.2	LamMPI
Node3	Intel1.8GHz	256MB	openSUSE11.2	LamMPI
Node4	IntelCeleron 2.4GHz	256MB	openSUSE11.2	LamMPI
Node5	IntelCeleron 2.4GHz	1GB	openSUSE11.2	LamMPI

We have designed and ran 7 test programs to examine the behavior of openMosix's load balancing mechanism with different types of node overloading including CPU, I/O, and IPC. We also tested its behavior on MPI directives.

A. Test No. 1, CPU-intensive

In this test, we check how openMosix reacts when cluster is overloaded with CPU-intensive processes. We use

a process (Figure 1) with an infinite loop that consumes as much CPU cycles as it could.

```
while (1){
    ;
}
```

Figure 1 - Pseudo code of a CPU-intensive process

We ran 20 instances of this process with *mosrun* command. Table 2 shows the results of these runs; the number in each cell shows the number of instances of the process on that node; we have ignored non-Mosix processes; the *mosrun* command started from Node1. As it is shown in Table 2, openMosix spreads the load on the entire cluster, with respect to nodes' abilities.

TABLE 2 CPU-INTENSIVE TEST RESULTS

	Node1	Node2	Node3	Node4	Node5
Startup	20	-	-	-	-
Balanced	4	4	4	4	4

The performance of each node depended on its processor power. Figure 2 shows the performance of cluster on each node.

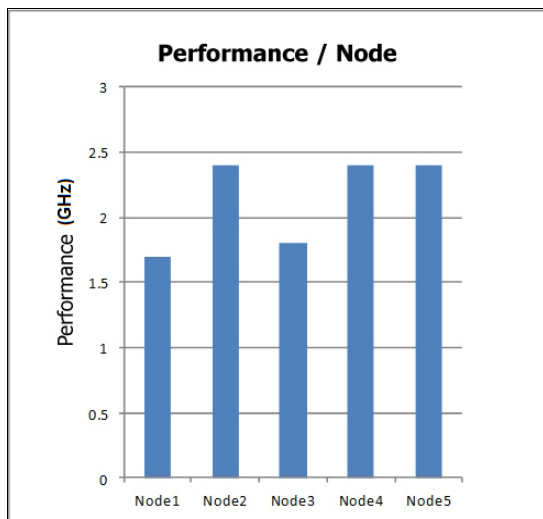


Figure 2 - Performance of nodes

Figure 3 shows the loads on the cluster nodes when 5 instances of a specific process ran on the cluster. Mosix calculates load for each node in the cluster with respect to relative node performance and number of processes in run queue. When numbers of processes on each machine are equal the output load diagram is look like Figure 3.

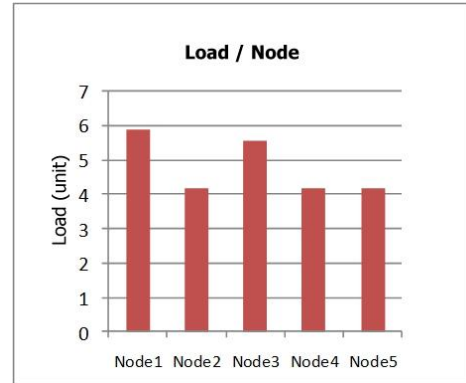


Figure 3 – Loads on nodes for CPU-intensive processes

Normalizing load with respect to node's relative performance, Mosix attempts to overcome the impact of performance heterogeneity. It calculates relative performance through dividing each node's performance by maximum performance in cluster.

B. Test No. 2, I/O-intensive

In this test, we check how openMosix deals with I/O-intensive processes. We use a process (Figure 4) with an infinite loop that sends an empty string to the standard output in each iteration.

```
while(1){
    printf("");
}
```

Figure 4 – Pseudo code of an I/O-intensive process

After running 20 instances of this I/O-intensive process on the cluster, Mosix migrated them to other nodes with respect to each node's performance. But before migrating each I/O-intensive process, it created a shadow process with a "." prefix in front of its name on the main host *mosrun* executed. While the process was migrated, the shadow process remained to handle future references of the migrated process to its local host. Due to the existence of these shadow processes and remote references from the migrated process to the local host, Mosix was reluctant to migrate I/O-intensive processes like CPU-intensive processes. Table 3 shows the results of this test, wherein the number in parentheses shows how many shadow processes were located on the local host.

TABLE 3 I/O-INTENSIVE TEST RESULTS

	Node1	Node2	Node3	Node4	Node5
Startup	20	-	-	-	-
Balanced	4 (20)	4	4	4	4

C. Test No. 3, Memory-intensive

In this test, we checked how openMosix dealt with memory-intensive processes. We used a process (Figure 5) containing CPU-intensive parts because memory-only-intensive processes did not trigger the Mosix load balancer mechanism.

```

for(i=1;i<1000000;i++){
    malloc(10000);
}
while (1){
    ;
}
    
```

Figure 5 - Pseudo code of a memory-intensive process

By running the code shown in Figure 5, a large amount of RAM was occupied at the beginning and then a CPU-intensive phase started. In regular situations, Mosix tried to reduce the load on a specific node by migrating some of the processes in that node to other nodes. Nevertheless, in this case it refused to do any migration although the nodes were overloaded. Migration of a large amount of memory is costly. Therefore, Mosix keeps memory-intensive processes in their places as long as the administrator does not force a migration.

D. Test No. 4, IPC-intensive – Direct

A process is IPC-intensive when it repeatedly sends messages to other processes. When a process sends too many messages, it becomes a candidate for migration by Mosix. However, as in the case of memory-intensive processes, the migration is costly and Mosix does not migrate them automatically. This reaction is a part of Mosix’s policy in dealing with communicating processes.

Figure 6 shows the results of running our test on 17 instances of two IPC-intensive sender and receiver processes.

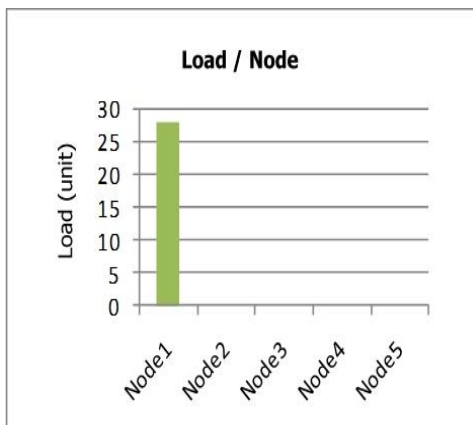


Figure 6 – Loads for IPC-intensive processes before migration

Mosix does not migrate any IPC-intensive processes in this experiment due to their communication cost. Migrating each IPC-intensive process may result in heavy communication cost. So, Mosix attempt to extract process’ IPC behaviors and make decision based on it. But when a process passes its IPC age, Mosix does not find its transition soon.

In the next experiment, processes on the first node were migrated to another node manually. After moving all processes to Node 5, the load of the first machine remained almost unchanged (Figure 7), implying that all IPC messages were redirected to the home node.

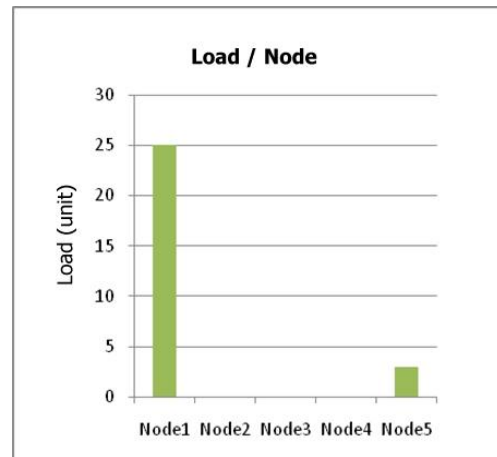


Figure 7 – Loads for IPC-intensive processes after migration

Whereas IPC-intensive processes have heavy communication cost, migrating them does not change their home node’s load. Migrated processes communicate with their deputy on home node and communicate via their home node. Therefore, home node’s load increases and processes response time falls down.

E. Test No. 5, IPC-intensive – Shared memory

Shared memory is another popular IPC mechanism that we had to investigate its support in Mosix. When the “mosrun” command was executed, Mosix returned the error message “MOSRUN: Attaching SYSV shared-memory not supported under MOSIX (try ‘mosrun -e’ or ‘mosrun -E’)”. This error means that MOSIX does not support shared-memory-based communication between processes.

F. Test No. 6, Forked processes

A child process inherits the features of its parents. In Mosix, a parent can fork a child that can in turn fork its own child. The hierarchy of parent-child can grow until the number of processes reaches a threshold.

We tested the inheritability of parent features in their children in Mosix and found out that whenever a parent process created a child process, Mosix passed the features of the parent to the child process (Table 4).

TABLE 4 FEATURE INHERITANCE OF FORKED PROCESSES

	Node1	Node2	Node3	Node4	Node5
Startup	1	-	-	-	-
Balanced	*	*	*	*	*

G. Test No. 7, Pipe-based processes

To investigate the behavior of Mosix in pipe-based IPC communications, we tested a pair of producer consumer processes. We initialized the consumer process manually that later created the producer process to provide input for the consumer process.

When the “mosrun” command was executed, Mosix returned the error message “MOSRUN: Attaching SYSV shared-memory not supported under MOSIX (try ‘mosrun -e’ or ‘mosrun -E’)”. This error means that MOSIX does not support pipe-based IPC communication between processes.

We thus used the “mosrun -e” command instead. The results were interesting. After running the test program with “mosrun -e” command, 10 processes were initialized on the first node but after a short time Mosix migrated some of them. Figure 8 shows the cluster status immediately after initializing processes on the first node while Figure 9 shows the cluster status some time after migrations happened and cluster became stable.

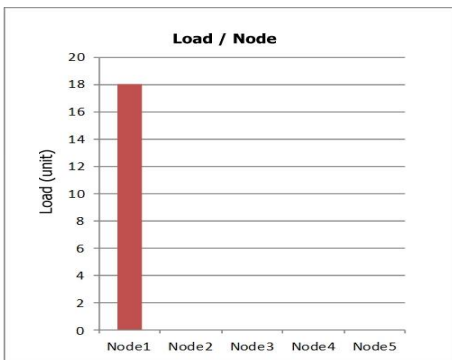


Figure 8 – Loads for Pipe-based processes before migration

Mosix treats piped processes as like as IPC-intensive processes but there is some difference. When processes communicate via pipe, their waiting time is more than the time they uses messages. Since Mosix counts number of processes in ready queue at each scheduling period, it calculates fewer loads.

The interesting point is that although Mosix migrated some processes to other nodes than their home (first node), the load on the home node remained unchanged (Figure 9). This was because copies of migrated processes remained on the home node to communicate with their producer counterparts. We can thus conclude that this migration had been redundant and only had increased the network traffic with undesirable effect on overall performance.

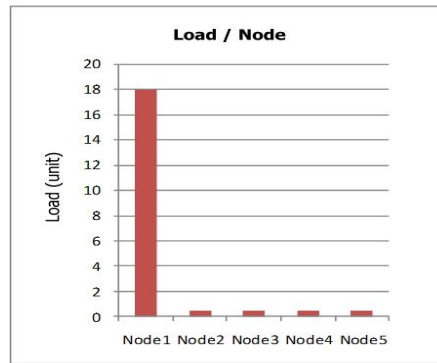


Figure 9 – Loads for Pipe-based processes after migration

By migrating some piped processes to other cluster nodes, communications must take place through communication infrastructures and file system. So, the home node’s load does not changed after migrating piped processes.

H. Test No. 8, MPI-based processes

MPI uses sockets and shared memory [10], while Mosix does not efficiently support these two communication mechanisms. Therefore, MPI processes could not be migrated by Mosix. A new “direct communication” feature has been recently added to Mosix that provides migratable sockets for MOSIX processes, but there is still no support for shared memory in Mosix [11], [12].

Therefore, it is impossible to run default MPI-based applications on a Mosix cluster yet. However, there are some short ways. LamMPI, configured with the “--with-rpi=tcp” option can bypass this limitation of Mosix.

In fact, “--with-rpi=tcp” option ensures that no shared memory is used in communications between processes. Therefore, when there is no shared memory in use, Mosix handles an MPI process like any other process.

I. Test No. 9, Priority in migration

To investigate how Mosix prioritizes processes for migration, we ran a number of tests. We tried to identify what processes become candidates for migration by Mosix. We compared two types of processes in each test, but compared all four types of CPU-intensive, I/O-intensive, IPC-intensive, and memory-intensive processes in our final experiment.

In our experiments, Mosix migrated CPU-intensive processes with low allocated memory first. It then migrated I/O-intensive processes and at last equally migrated the IPC-intensive and memory-intensive processes. However, this order was not fixed on all Mosix clusters because of Mosix decision making function. For example, if the power of machines and the amount of available physical memory installed on each machine in a Mosix cluster were widely different, the pattern of Mosix migration priority might be diverse.

IV. CONCLUSION AND FUTURE WORKS

We ran a number of test programs on a 5-node Mosix cluster to check the real abilities of Mosix in providing a reasonably high performance in handling different requests. We found that Mosix did not guarantee the performance improvement in all cases and that it even reduced the performance by making wrong decisions.

We showed that the Mosix cluster handled CPU-intensive, memory-intensive, and I/O-intensive processes effectively although it was slow and sometimes inaccurate when the cluster was overloaded with large memory-intensive processes. It also properly supported feature inheritance by inheriting all features of parents in the forked children.

We also showed that Mosix did not support shared-memory-based communications between processes and as a result did not support MPI-based processes too unless processes used a different mechanism for their communications than the shared memory. Worst of all, Mosix misbehaved in dealing with pipes and made decided wrongly in migrating tightly-connected processes to other machines, lowering the performance and increasing the network traffic rather than improving the performance.

REFERENCES

- [1] A. Barak, S. Guday, and R. G. Wheeler, *The MOSIX Distributed Operating System: Load Balancing for UNIX*, Secaucus, Ed. New York, USA: Springer, 1993.
- [2] A. Barak and O. La'adan, "The Mosix Multicomputer Operating System for High Performance Cluster Computing," *Future Generation Computer Systems*, vol. 13, no. 4-5, pp. 361-372, Mar. 1998.
- [3] A. Barak, A. Braverman, I. Gilderman, and O. Laadan, "The MOSIX Multicomputer Operating System for Scalable NOW and its Dynamic Resource Sharing Algorithms," *The Hebrew University Technical* 96-11, 1996.
- [4] A. Barak, O. La'adan, and A. Shiloh, "Scalable Cluster Computing with Mosix for Linux," in the 5th Annual Linux Expos, Raleigh, 1999, pp. 95-100.
- [5] J. M. Meehan and A. Ritter, "Machine Learning Approach to Tuning Distributed Operating System Load Balancing Algorithms," in *Proceedings of the ISCA 19th International Conference on Parallel and Distributed Computing Systems (ISCA PDCS)*, San Francisco, 2006, pp. 122-127.
- [6] M. Beltrán and A. Guzmán, "How to Balance the Load on Heterogeneous Clusters," *International Journal of High Performance Computing Applications*, vol. 23, no. 1, pp. 99-118, Feb. 2009.
- [7] R. Lavi and A. Barak, "The Home Model and Competitive Algorithms for Load Balancing in a Computing Cluster," in the 21st International Conference on Distributed Computing Systems, Washington DC, Apr, 2001, pp. 127-136.
- [8] A. Barak and A. Braverman, "Memory ushering in a scalable computing cluster," *Microprocessors and Microsystems*, vol. 22, no. 3-4, pp. 175-182, Aug. 1998.
- [9] A. Keren and A. Barak, "Opportunity Cost Algorithms for Reduction of I/O and Interprocess Communication Overhead in a Computing Cluster," *IEEE Transactions on Parallel and Distributed Systems*, vol. 14, no. 1, pp. 39-50, Jan. 2003.
- [10] "MPI: A Message-Passing Interface Standard," *Message Passing Interface Forum Standard 2.2*, 2009.
- [11] S. D. S. Pty/Ltd. Mosix updates. <http://www.mosix.com.au/updates.html>. Jul, 2011, [retrieved: Sep, 2011].
- [12] A. S. Amnon Barak. The MOSIX Management System for Linux Clusters, Multi-Clusters, GPU Clusters and Clouds. <http://www.mosix.org>. Apr, 2010, [retrieved: Sep, 2011].

Characterizing Energy Efficiency in I/O System for Scientific Applications

Javier Panadero, Sandra Méndez, Dolores Rexachs and Emilio Luque
 Computer Architecture and Operating Systems Department (CAOS)
 Universitat Autònoma de Barcelona
 Barcelona, Spain
 javier.panadero@campus.uab.es
 {sandra.mendez, dolores.rexachs, emilio.luque}@uab.es

Abstract—The increasing complexity of scientific applications and the increase of scalability in high performance computing systems demand a more powerful Input/Output system. This requirement is present in both performance and power consumption. For this reason, performance, power consumption, energy and energy efficiency have become critical measures in Input/Output systems. Nowadays, when a High Performance Computing center buys a system of storage not only does it take into account the price, but also the cost of its useful life cycle as well as the energy cost. This paper proposes a methodology to characterize the energy efficiency of the Input/Output system, considering the Input/Output system at a device level, I/O library and file system. The methodology provides a wide range of I/O system parameters that have an impact on the energy efficiency. Furthermore, we evaluate the impact of Input/Output intensive applications in energy efficiency.

Keywords-Energy Efficiency; Consumption; I/O System; HPC; Methodology.

I. INTRODUCTION

Energy efficiency has become an extremely important consideration for the storage system, due to several factors, among which the most important is the scalability of the system, because we will not be able to expand the system if we have consumed all the available energy [1]. Another important factor to consider is the cost of the Input/Output (I/O) system, due to the Kilowatt/hour rate imposed by the electricity company. It is because of this reason that nowadays, when a High Performance Computing center buys a system of storage, not only does it take into account the price, but also the cost of its useful life cycle.

These days, we can find similar rankings to the Top500 [2] such as the Green500 [3], where we can obtain a list of the supercomputers with the highest energy efficiency in computing. The Green500 updates its ranking 3 times a year in order to increase awareness about power consumption and energy. Furthermore, they promote alternative total costs and ensure that supercomputers do not only simulate the climate change but they do not help to its degradation. For I/O systems, it is not easy to find analysis for comparing energy efficiency.

In considering power consumption and energy efficiency, the scientific community has an extremely important challenge to overcome [4]. When we take into account both the energy

efficiency and a determinate performance of the I/O system, it is important to have common sense for the configuration of the I/O system. First of all, we are going to consider, how to do an energy diagnosis; What can characterize of the I/O system; How can analyze power consumption and energy efficiency; What metric should we use. All these questions are necessary to be able to plan and to propose improvements of energy efficiency in the configurations of the I/O system. In this paper, we offer a methodology for characterizing energy efficiency in the I/O system. The proposed methodology relates application phases to power consumption phases throughout the execution time. This methodology considers the I/O system at device level, I/O library and global file system. On the other hand, it extracts information about throughput, power consumption, energy and energy efficiency in a system with different device access patterns. Considering this information, we analyze the impact of energy efficiency for the different configurations of the I/O system. The methodology allows us to characterize I/O scientific applications such as EarthScience, NuclearPhysics, CombustionPhysics, etc. This methodology serves as a starting point to be able to decide on the dimensioning of the I/O system that improves its energy efficiency. We also analyze the impact of the executed benchmarks in the characterization of the I/O system.

In this paper, we use a Watts UP pro ES digital power meter to take measurements and analysis for the I/O system. This meter provides a sampling each second.

This paper is organized as follows: in the next section we briefly review related work. In Section III, we introduce our methodology for characterizing energy efficiency in the I/O system. Then, in Section IV, we expose and analyze the experimental results. Finally, in Section V, we present the conclusions and future work.

II. RELATED WORK

The work that we introduce in this paper is related to the analysis and characterization of the I/O system.

Ge [4] proposes a methodology to profile the performance, energy and energy efficiency considering the parallel I/O access patterns and the CPU frequency. This study differs from our work since we do not just consider the I/O access patterns.

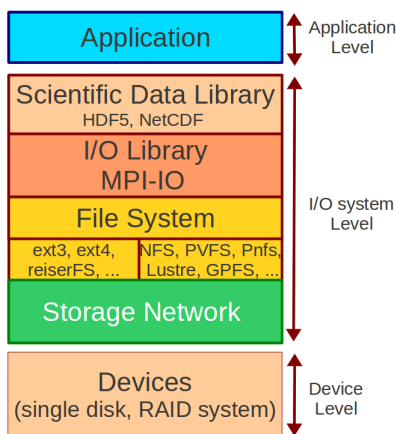


Fig. 1. I/O system

On the other hand, Hylick [5] proposes an analysis of power consumption at device level, considering the dependence of the locality of the data and the block size of access. Our study differs from this one since we do not consider just the I/O at the device level but also the I/O at the system level.

Another study proposed by Sehgal [6] analyzes the energy and energy efficiency by considering several Linux local file systems modifying the default options. Our work differs from his study because we do not consider modifying the default options but instead we consider different file system levels (local, distributed and parallel).

In terms of massive storage, Dong [7] proposes an analysis about power characteristics of read/write operations compared with the power efficiency of different RAID levels. Our work considers RAID levels as a part of the characterization of device level.

III. PROPOSED METHODOLOGY

In this section, we detail our methodology for the characterization of the energy efficiency in terms of performance, power, energy and energy efficiency.

We characterize at device level and at I/O system level. Fig. 1 illustrates the I/O system; we characterize the elements that include the device level and the I/O system level. The methodology is divided into four parts:

- Metrics selection used during the characterization,
- Characterization at device level,
- Characterization at I/O system level and
- Characterization of the benchmark parameters.

Fig. 2 illustrates these parts and, in addition, the information obtained in each part of the characterization.

A. Metrics used in the methodology

In this section, we detail the metrics used in the methodology for: performance, power, energy and energy efficiency.

The performance of the I/O operations is normally quantified using throughput (number of megabytes transferred per second) and/or IOPs (Input/Output Operations Per Seconds). The throughput (MB/sec) is obtained directly by the I/O

Where?	Benchmark	System characterization	Devices characterization
What?	<ul style="list-style-type: none"> -Access Patterns -Type of operations (write and read) -Size request -Number of processes -I/O library -Phases of the benchmark 	<ul style="list-style-type: none"> -Bandwidth and power on: <ul style="list-style-type: none"> -I/O library -Filesystem Local and Global (parallel, distributed and network) -Connection Storage (NAS, DAS, SAN, NASD) -Buffer/Cache -Interconnection Network 	<ul style="list-style-type: none"> -Bandwidth and power on: <ul style="list-style-type: none"> •Devices •RAID level •Disk Cache •Rotation Disk
How?	<ul style="list-style-type: none"> -Benchmarks Configuration -Benchmark Documentation -Trace of benchmark 	<ul style="list-style-type: none"> -Filesystem Benchmark (Iozone, Bonnie++) -I/O library benchmark (IOR, IOBench) 	<ul style="list-style-type: none"> -Disk benchmark (Iozone, Bonnie++) -Tools system linux (hdparm, vmstat, iostat)
Measurements?		<ul style="list-style-type: none"> -Performance: Bandwidth (MB/sec) -Power: watts (W) -Energy: Joules (J) -Energy efficiency (MB/J) 	<ul style="list-style-type: none"> -Performance: Bandwidth (MB/sec) -Power: watts (W) -Energy: Joules (J) -Energy efficiency (MB/J)

Fig. 2. Characterization of the I/O system

benchmarks or Linux I/O tools and IOPs are obtained indirectly analyzing the bandwidth and the duration time of I/O operations.

We use the watt (W) as metrics for power and the Joule (J) for the energy, both included in the international unit system. The power meter obtains Watts directly and energy is derived indirectly from the total execution time of the benchmark per average consumption power of the benchmark execution.

There is not a standard measurement for energy efficiency. Due to the interrelation between performance and energy, Liu [8] proposes two new metrics for energy efficiency: IOPS/Watt and MBPS/Kilowatt. For this study we have chosen MBPS/watt as efficiency metrics. We use the equation mentioned above introducing the energy. For this reason, we use the equation (1) and we obtain MB/J as the final equation for the energy efficiency.

$$1 \text{ Joule} = 1 \text{ Watt} * 1 \text{ Second} \tag{1}$$

B. Characterization at the device level

This phase consists of the characterization of the devices and RAID system using I/O benchmark as Iozone [9] or/and Bonnie++ [10]. These benchmarks generate and measure a wide variety of file disk operations. During the characterization, we consider the access patterns (sequential, random, stride), the request size of operations and the type of access (block or character). For these operations, we characterize the bandwidth, the power consumption, the energy and the energy efficiency. We also consider the device's different states of power consumption.

C. Characterization of the I/O system level

This phase consists of the characterization of the I/O library, the file system (local, distributed and parallel), storage connection (NAS, SAN, DAS, NASD) and the system buffer cache. We obtain the bandwidth, the power consumption, the energy and the energy efficiency. In order to characterize the I/O system level, one could use the IOzone or/and Bonnie++

file system benchmarks. IOR [11] or/and PIO-bench [12] could be used as I/O library benchmarks. These benchmarks leverage the scalability of MPI to accurately calculate the throughput of a given number of client machines. As we have already illustrated in figure 1, this characterization is linked with the characterization at the device level because the data follows a process until it is written or read in the final device (single device or RAID system). For this reason, at the same time we carried it out at the I/O system level and at the device level.

D. Benchmark parameters characterization

To characterize the different levels, we use I/O benchmarks. These benchmarks have many configuration parameters, the access patterns (sequential, random, stride), the request size of operations, the type of access (block or character), the number of processors, the type of I/O library, amongst others.

The objective is to tune the specific parameters of the benchmarks, according to the characterization that we are doing.

E. Characterization of the I/O system

We carried out our characterization on two different systems. The first system characterized was a Pentium 4 single core, with a 512 MB RAM memory and a single device Seagate Barracuda ATA ST340016A. It also has a capacity of 80GB and a cache disk of 2MB. Fig. 3 illustrates the power consumption specifications of each state and also the transitions of the device. The local file system used was Ext4 with a DAS store network. The I/O library used was MPICH. The second system characterized was the cluster Aohyper. This cluster has 8 dual Core nodes AMD Athlon(tm) 64 X2, 2 GB RAM memory, 150 GB local disk. The local file system used was Ext4. Also has a 1 NFS server with RAID 1 (2 disks) with 230GB capacity and RAID5 (3 disks) with stripe=256KB and 917GB capacity, both with write-cache enabled (write back). The networks used were two Gigabit Ethernet network, one for communication and one for data.

We utilized the Watts UP pro digital power meter to measure. It was connected to the output AC power source of the computer. This meter provides a sampling each real-time second.

These two systems have been characterized. Now, we present how we characterized the I/O system. Although we expose the workstation characterization, this characterization is extensible for the cluster characterization or another system. We describe the workstation characterization, however it could be extended for the cluster characterization or any other system.

1) *The device characterization:* What we did, first of all, was to characterize the effects of power-saving using state controls. In order to do that, we used the benchmark IOR, which was executed twice with an interval of 60 seconds between each execution. Fig. 4 illustrates the result of the execution with power-saving using state controls and without power-saving. Fig. 5 shows the power consumption and the energy required for the two executions. The result of execution

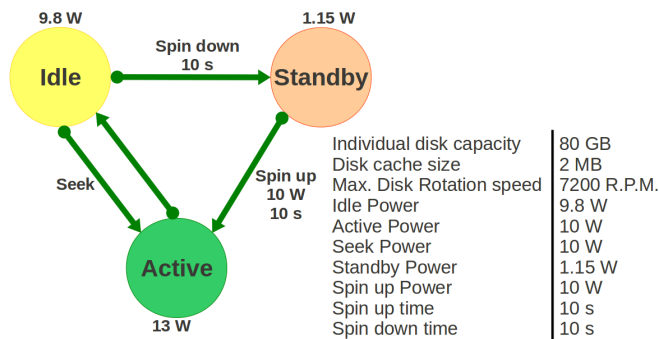


Fig. 3. Specifications of power consumption

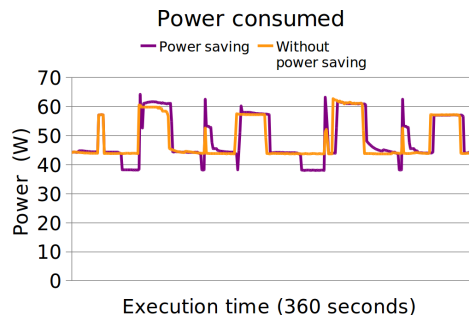


Fig. 4. Execution both power saving and without power saving

is better without power-saving states than with power-saving states. This is due to several factors such as: short Standby periods, the cost of spin-up transactions (time and power) and the peaks obtained during the spin-up transactions. Applications with short standby periods do not take advantage of power saving states.

After that, we characterized different access patterns (sequential and random). In order to do that, we used the benchmark IOzone with different requests sizes and a file size of 1GB. Fig. 6 illustrates the result of the characterization for sequential access patterns. Finally, Fig. 7 illustrates the characterization for random access patterns. We observe the following trends if we execute the sequential characterization and the set of request sizes tested in read operations. In the

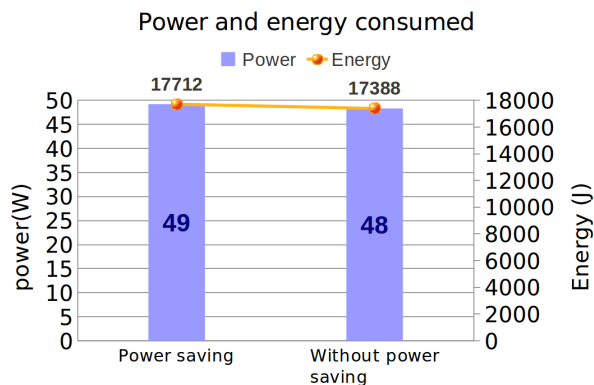
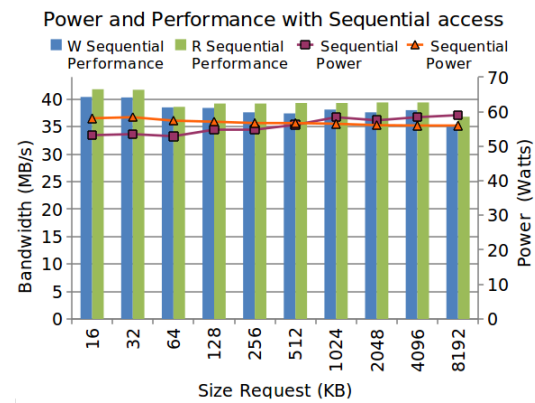
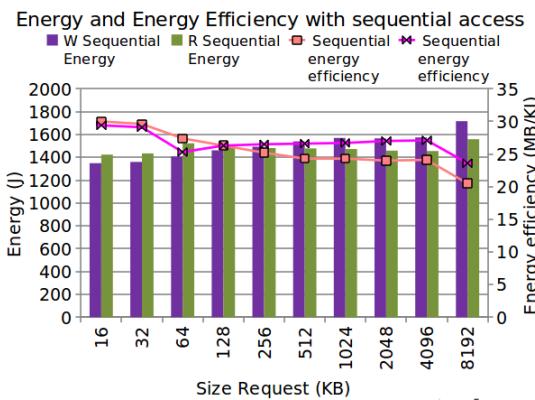


Fig. 5. Power and energy for execution with power saving



(a) Bandwidth and Power

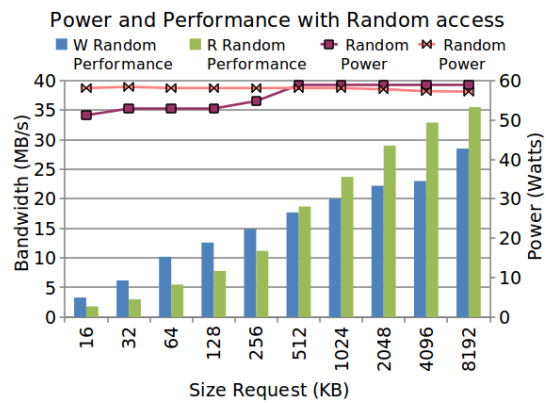


(b) Energy and energy efficiency

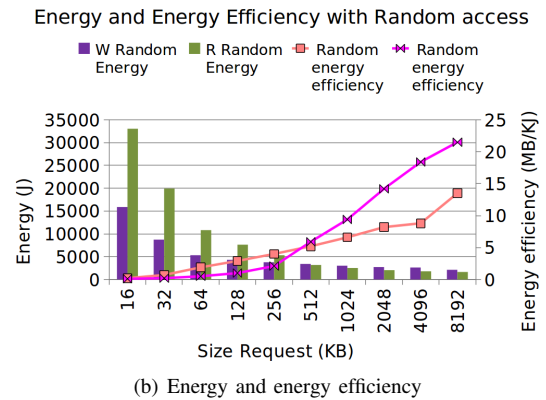
Fig. 6. Characterization for sequential access patterns

case of a very small-sized request (16 KB - 32 KB), we obtain a larger bandwidth, more power, less energy and an increase in energy efficiency. On the other hand, in the case of a request size of 8192KB, we obtain the worst bandwidth and the lowest power of the whole set of request size. The result is larger energy consumption and the worst energy efficiency. In write operations, in terms of bandwidth, energy and energy efficiency show the same trends. However, in terms of power, by increasing the request size, the power increases. Despite the small power variation that we observe, it has an influence in the metrics of energy and energy efficiency. We observe the following trends if we execute the random characterization and the set of request size tested in read and write operations. In the case of a larger request size, we obtain a larger bandwidth, less energy and more energy efficiency. Because of the data locality, as we increase the request size, data transferring time is longer than data seeking time. Power consumption has two different trends depending on the operation type. In write operations, if we increase the request size we obtain more power consumption. On the other hand, in read operations, if we increase the request size the lowest power consumption is obtained.

2) *I/O system characterization:* At I/O system level, we characterized the influence of different cache levels on the system. It is worth mentioning that we have included the



(a) Bandwidth and Power

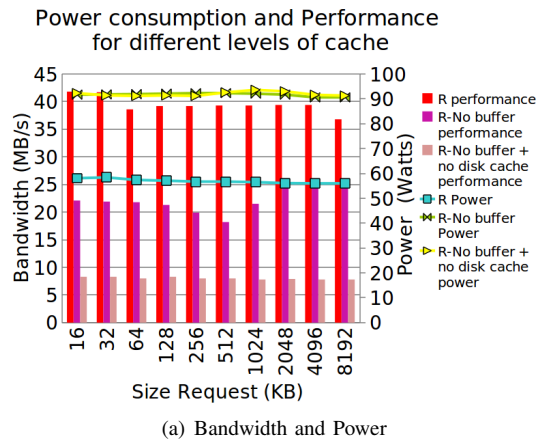


(b) Energy and energy efficiency

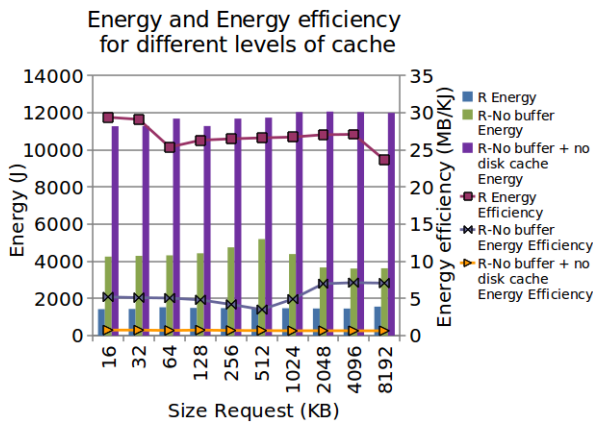
Fig. 7. Characterization for random access patterns

cache disk as part of the system cache hierarchy. We call the buffer memory of the operating system buffer cache. For that reason, we used the benchmark IOzone with different request sizes and size file of 1GB. Fig. 8 illustrates the result of execution with different levels of cache disabled. In this figure, we observe the following trends: bandwidth and energy efficiency decrease whereas power and energy increase as we disable levels of cache. There are no substantial differences between the power consumption without buffer cache disabled and disk cache enabled; and there is no difference either between power consumption without buffer cache disabled and disk cache disabled. On the contrary, there are differences in energy and energy efficiency for those configurations, because of the influence of the bandwidth.

Moreover, we also characterized the influence of the I/O library. To achieve these study objectives, we characterized the MPICH library. In order to do that, we used the benchmark IOR for 1 core, with different request sizes and a file size of 1GB. The goal was to observe the influence in energy efficiency with the insertion of the new layer in the I/O stack. In Fig. 9, we observe the following trends: bandwidth and energy efficiency decrease whereas energy increases for a larger request size. Power consumption in write operations increases until a request size of 512 KB and then, it begins to decrease. In read operations, in the case of request size (16KB - 512 KB), we obtain the same trend in relation to power. It



(a) Bandwidth and Power



(b) Energy and energy efficiency

Fig. 8. Execution with different levels of cache disabled

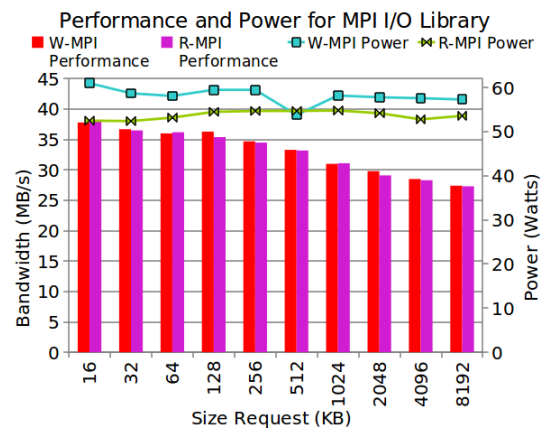
begins to decrease from a 512 KB request size on. It also influences energy efficiency.

IV. EXPERIMENTATION

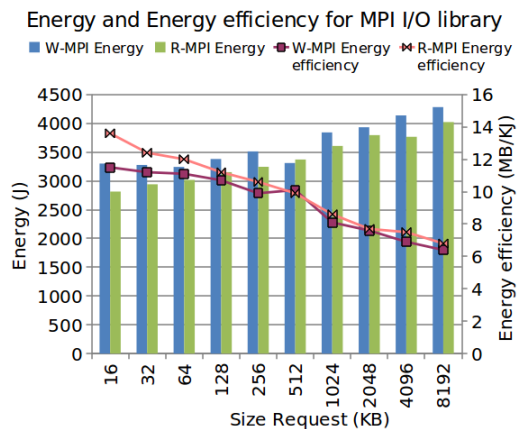
In order to evaluate the methodology for the characterization, we selected the Block Tridiagonal (BT) application of NAS Parallel Benchmark (NPB) suite. We executed this application in the systems that were previously characterized. On the workstation, the application was executed in its class B and subtype full. That configuration writes a file size of 1,5 GB. Whereas for the execution on the cluster, we executed the application in its class C and subtype full for 16 processors. That configuration writes a file size of 6,5 GB.

Fig. 10 shows the trace of the application for the workstation. The violet color represents write operations and the green color represents read operations. We observe 3 different phases; involves the computing and discontinuing write operations of 128 KB request size. The second phase is I/O intensive in write operations, whereas the last phase is I/O intensive in read operations.

Fig. 11 illustrates power consumption during the application execution. After an initial time when the state of the device was idle, we executed the application. We observed 2 distinct phases; in the first phase, we observed more consumption than in the second phase. This is due to application compute



(a) Bandwidth and Power



(b) Energy and energy efficiency

Fig. 9. Characterizing the influence of the I/O library

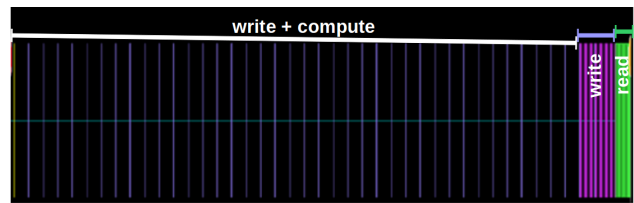


Fig. 10. Trace of the application NAS BT class B subtype FULL

and write discontinuous operations that took place during the first phase. Whereas in the second phase only intensive I/O operations without compute were made. During the first phase, we observed peaks of consumption, which are caused by the addition in write operations of power to compute power.

Fig. 12 shows the trace of the application for the cluster. The application was executed with two different RAID configura-

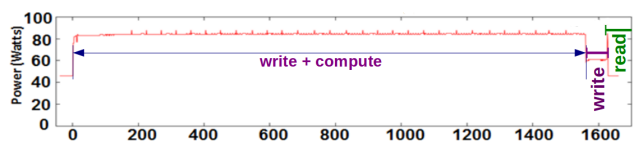


Fig. 11. Power of the execution application BT class C subtype FULL

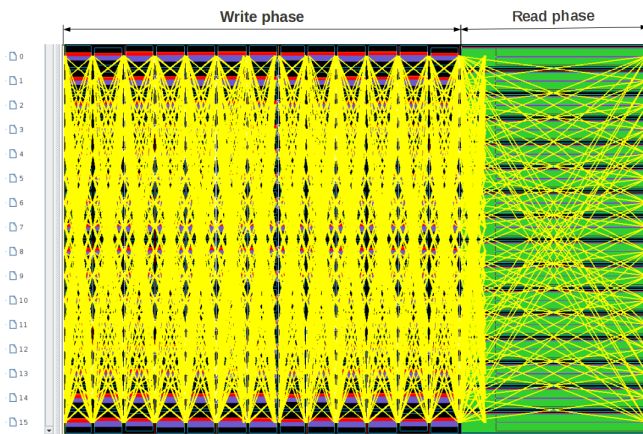
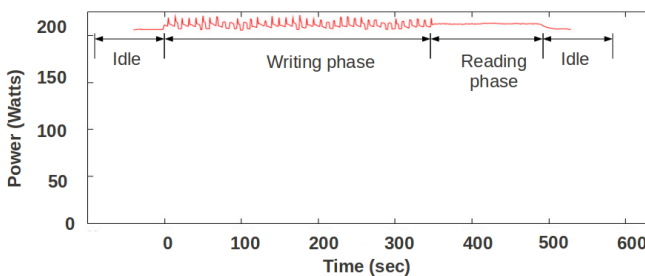
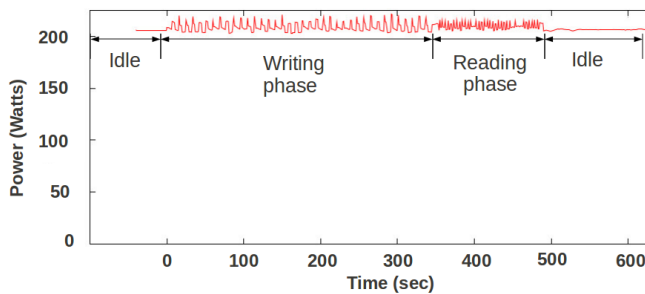


Fig. 12. I/O Phases of NAS BT-IO by 16 processes FULL subtype



(a) RAID 1



(b) RAID 5

Fig. 13. Power consumed for NAS BT-IO by 16 processes

tions (RAID 1 and RAID 5). The yellow color represents write operations and the green color represents read operations. We observe 2 different phases; the first phase is I/O intensive in write operations of 128 KB request size, whereas the second phase is I/O intensive in read operations.

Fig. 13(a) illustrates the power consumption during the application execution on the cluster for a RAID 1 configuration, whereas the Fig. 13(b) illustrates the power consumption during the application execution on the cluster for a RAID 5 configuration.

After an initial time where the state of the devices was idle, we observed 2 distinct phases for the two configurations. The first configuration phase is the power consumption for I/O intensive write operations; whereas the second phase is the power consumption for I/O intensive read operations. We observed that because the I/O system is apart from the compute

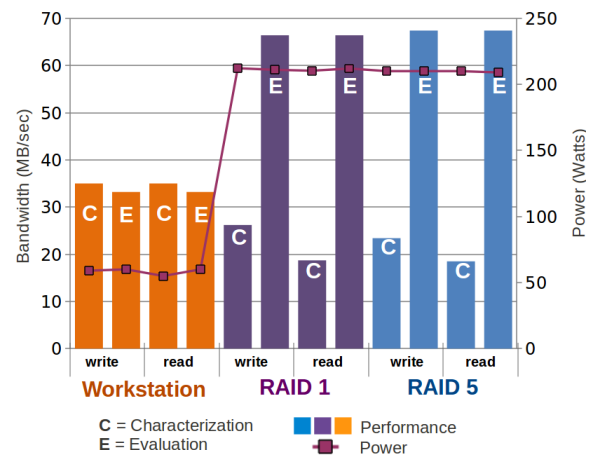


Fig. 14. Real and reference values obtained

TABLE I
PERCENTAGE (%) DEVIATION OBTAINED

Operation type	write	read
WorkStation	3%	7%
RAID 1	0.6%	0.8%
RAID 5	0%	0.5%

nodes, the power consumption values obtained are only of I/O operations. Because of this, the intensive computation does not have in the results.

Fig. 14 illustrates real values obtained during the application execution and reference values obtained during the characterization for all systems characterized. We show the performance and the power consumption. In terms of workstation, we selected real values of performance and power consumption obtained during the execution of the application's second phase because it is the I/O intensive phase. We selected the reference values results of a 128Kb size request obtained in Fig. 9, because the BT application is a parallel MPI application that uses a 128 KB request size in I/O operations.

In terms of cluster values, because the I/O system is apart from the compute nodes, we selected the real values obtained in each phase. We selected the reference values from the cluster characterization in the same way as we did before for the workstation architecture. We observe close results in power consumption in all characterized systems. However, in bandwidth, we observe significant differences for cluster configurations. These differences are because the application did not manage to stress the system.

Table I, shows the power consumption deviation of reference values with real values. We observe close results between real and reference values.

V. CONCLUSION

In this paper, we propose a new methodology to characterize the energy efficiency in the I/O system. The methodology takes into account performance and energy. Moreover, it extracts information about the bandwidth, the power consumption, the energy and the energy efficiency from different I/O benchmarks. We evaluated the methodology with real applications

and we observed that the reference values of characterization were close to the real values obtained with the application's execution. We also observed that in intensive operations of the I/O system, power consumption changed to a small extent. However, that change did modify energy and energy efficiency.

This paper is just a small part of our research and will serve to find new ways of investigating. Our final goal will be to propose a methodology for dimensioning the I/O system in terms of energy and energy efficiency. That methodology will be able to characterize, analyze and evaluate the I/O system for dimensioning. We are also looking for a new method to identify the significant phases in terms of power consumption at device level considering the writing and reading blocks. In order to carry this out, we are also working in new metrics for energy efficiency.

ACKNOWLEDGMENT

This research has been supported by MICINN-Spain under contract TIN2007-64974.

REFERENCES

- [1] T. Minartz, J. Kunkel, and T. Ludwig, "Simulation of power consumption of energy efficient cluster hardware," *Computer Science - Research and Development*, pp. 165–175, 2010. [Online]. Available: <http://www.springerlink.com/content/r21r2376730p7161/fulltext.pdf>
- [2] T. 500, "Top 500 supercomputing list," Tech. Rep., [Retrieved: September, 2011]. [Online]. Available: <http://www.top500.org>
- [3] G. 500, "Green computing list," Tech. Rep., [Retrieved: September, 2011]. [Online]. Available: <http://www.green500.org>
- [4] R. Ge, X. Feng, S. Subramanya, and X. he Sun, "Characterizing energy efficiency of i/o intensive parallel applications on power-aware clusters," in *Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on*, april 2010, pp. 1–8.
- [5] A. Hylick, R. Sohan, A. Rice, and B. Jones, "An analysis of hard drive energy consumption," in *Modeling, Analysis and Simulation of Computers and Telecommunication Systems, 2008. MASCOTS 2008. IEEE International Symposium on*, sept. 2008, pp. 1–10.
- [6] P. Sehgal, V. Tarasov, and E. Zadok, "Optimizing energy and performance for server-class file system workloads," *Trans. Storage*, vol. 6, pp. 10:1–10:31, September 2010. [Online]. Available: <http://doi.acm.org/10.1145/1837915.1837918>
- [7] Y. Dong, J. Chen, and T. Tang, "Power measurements and analyses of massive object storage system," in *Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on*, 29 2010-july 1 2010, pp. 1317–1322.
- [8] Z. Liu, F. Wu, X. Qin, C. Xie, J. Zhou, and J. Wang, "Tracer: A trace replay tool to evaluate energy-efficiency of mass storage systems," in *Proceedings of the 2010 IEEE International Conference on Cluster Computing*, ser. CLUSTER '10, 2010, pp. 68–77, [Retrieved: June, 2011]. [Online]. Available: <http://dx.doi.org/10.1109/CLUSTER.2010.40>
- [9] W. D. Norcott, "Iozone filesystem benchmark," Tech. Rep., [Retrieved: September, 2011]. [Online]. Available: <http://www.iozone.org/>
- [10] R. Coker, "Bonnie++ filesystem benchmark," Tech. Rep., [Retrieved: September, 2011]. [Online]. Available: <http://www.coker.com.au/bonnie++/>
- [11] . S. J. Shan, Hongzhang, "Using ior to analyze the i/o performance for hpc platforms," LBNL Paper LBNL-62647, Tech. Rep., [Retrieved: September, 2011]. [Online]. Available: www.osti.gov/bridge/servlets/purl/923356-15FxFxGK/
- [12] F. Shorter, *Design and analysis of a performance evaluation standard for parallel file systems*, [Retrieved: September, 2011]. [Online]. Available: <http://books.google.com/books?id=g7sEOAAACAAJ>

Supervised Hybrid SOM-NG Algorithm

Mario J. Crespo-Ramos, Iván Machón-González, Hilario López-García
 Área de Ingeniería de Sistemas y Automática (DIEECS)
 Universidad de Oviedo
 Gijón (Asturias), Spain
 E-mail: crespomario@uniovi.es, machonivan@uniovi.es,
 hilario@uniovi.es

José Luis Calvo-Rolle
 Escuela Universitaria Politécnica de Ferrol.
 Departamento de Ingeniería Industrial
 Universidad de A Coruña
 A Coruña, Spain
 E-mail: jcalvo@cdf.udc.es

Abstract—The hybrid SOM-NG algorithm was formulated to improve the quantization precision in Self Organizing Maps by the means of combine both SOM and Neural Gas properties using a parameter γ to tune the topology preservation. A supervised learning algorithm is proposed to take advantage of the balanced hybrid algorithm. The proposed algorithm makes a linear approximation of the goal function for every Voronoi region. The algorithm gives good estimations and well balanced prototype positions combining the benefits of the original algorithms.

Index Terms—hybrid algorithm, supervised learning, neural networks, self-organizing mapping, neural gas.

I. INTRODUCTION

The Self-Organizing Map (SOM) [1] is a very popular algorithm because of its many features, specially the topological preservation between input and output spaces. Most of the uses of SOM were related with dimensionality reduction and unsupervised training, but SOM has been used in a supervised way in different applications. Supervised SOM was developed by different authors in different ways specially for classification purposes, but in this work numeric modeling will be studied.

Supervised modeling was approached in different ways. Additive composition of several supervised SOM networks [2] was developed from the theory that an additive composition of linear functions can be estimated with an additive composition of neural networks. Continuous Interpolation Self-Organizing Maps (CI-SOM) [3] are created using interpolation methods to approximate a continuous function using discrete prototype vectors. For time-series regression, special networks were created, Temporal Kohonen Map (TKM) [4] and Recurrent SOM [5] are examples of the use of leaky integrators for temporal sequence processing, they are compared in [6].

The main drawback of SOM training is the lack of quantization precision, specially if compared to Neural Gas (NG) [7] algorithm that does not take into account topological order aiming to minimize the quantization error. In [7], a regression method is also presented, using a reference value for the the prototype vector's position and a gradient vector for its Voronoi region.

Using these algorithms has a main advantage of using local approximations for each prototype vector, i.e., a model in every Voronoi region, so a closer model can be obtained if compared with using a single model for the whole data set. Using local

linear models also require less computational resources, as they use simple mathematical techniques.

A hybrid algorithm was proposed based on both SOM and NG in [8]. This algorithm combines their properties and obtains a more balanced result achieving a trade-off between topology preservation and quantization error, i.e., between data projection and accuracy. This hybrid algorithm is briefly explained in Section II. Using this algorithm, a new estimation tool will be formulated in Section III and Section IV will explain the training procedure. The algorithm is tested in Section V and discussed in Section VI.

II. SOM-NG ALGORITHM

The SOM-NG algorithm is based on two different kernels:

$$h_{NG}(v, w_i) = \exp\left(-\frac{k(v, w_i)}{\gamma^2 \cdot \sigma(t)}\right) \quad (1)$$

$$h_{SOM}(v, w_i) = \exp\left(-\frac{s(r_{i*}, r_i)}{\sigma(t)}\right) \quad (2)$$

where $k(v, w_i)$ and $s(r_{i*}, r_i)$ are rank functions, γ is the topology preservation constant and $\sigma(t)$ is the neighborhood radius in training epoch t .

Each kernel determines the behaviour of each algorithm, i.e., h_{NG} is the influence over quantization error, as in NG, and h_{SOM} represents the topology preservation in a SOM-like way. For the sake of simplicity the parameters will be removed from expressions and h_{NG} and h_{SOM} will be used instead of $h_{NG}(v_j, w_i)$ and $h_{SOM}(v_j, w_i)$ in the whole paper.

The ranking function $k(v, w_i)$ is the position of prototype vector w_i in the distance ranking to data vector v in the input space. The best matching unit, i.e., the closest one, gets ranking 0, the next closer one gets 1 and continues until $m-1$, where m is the number of prototype vectors.

Ranking function $s(r_{i*}, r_i)$ is a bit more complex because it is a modified ranking in distance between the best matching unit r_{i*} and unit r_i considering both of them on the output space. The modified ranking imposes that map units at same distance from a map unit must have same ranking value, so ranking values in a square lattice will be like 0,1,1,1,1,5,5, and so on. Calculating this ranking is a bit more complex than a monotonically increasing one, but it is constant during

the training, so it is calculated once before training starts. This modified ranking function obtains more robust results than a monotonous one because of the lack of random tie-breakers. The modified ranking is similar to a sigmoid function of distance, and it produces a slimmer Gaussian bell in the prototypes close to the *bmu* with a wider base in the farther ones.

The neighborhood radius $\sigma(t)$ was chosen to decrease exponentially to improve the algorithm's steadiness, the recommended expression is:

$$\sigma(t) = \sigma_{t0} \cdot \left(\frac{\sigma_{tmax}}{\sigma_{t0}} \right)^{t/tmax} \quad (3)$$

where t is the training epoch, t_{max} is the number of training epochs, σ_{t0} is the initial value and σ_{tmax} is the final value. Appropriate values for them are: $\sigma_0 = m/2$ and $\sigma_{tmax} = 0.001$, since it should tend to zero in order to minimize the quantization error.

In the unsupervised algorithm, the energy cost function optimizes quantization in a cooperative way. The energy cost function according to the squared Euclidean distance is:

$$E = \sum_{i,j} h_{NG} \cdot h_{SOM} \cdot \|v_j - w_i\|^2 \quad (4)$$

The batch version of the algorithm is obtained using Newton's method (5).

$$\Delta w_i = -J_E(w_i) \cdot H_E^{-1}(w_i) \quad (5)$$

where J_E is the Jacobian matrix of the energy cost function E and H_E is the Hessian matrix. Thus they are calculated from (4) and the following expressions are obtained:

$$J_E(w_i) = -2 \cdot \sum_j h_{SOM} \cdot h_{NG} \cdot \vec{d}_{ji} \quad (6)$$

where $\vec{d}_{ji} = v_j - w_i$ is the distance vector for data point v_j and prototype vector w_i .

$$H_E(w_i) = 2 \cdot \sum_j h_{SOM} \cdot h_{NG} \quad (7)$$

The increment for each prototype vector w_i in every epoch is calculated using (6) and (7) by substitution in (5):

$$\Delta w_i = \frac{\sum_j h_{SOM} \cdot h_{NG} \cdot \vec{d}_{ji}}{\sum_j h_{SOM} \cdot h_{NG}} \quad (8)$$

And the updating rule is:

$$w_i = \frac{\sum_j h_{SOM} \cdot h_{NG} \cdot v_j}{\sum_j h_{SOM} \cdot h_{NG}} \quad (9)$$

This updating rule is very similar to batch SOM and batch NG [9] ones. If the h_{SOM} term is deleted, i.e., replacing it by the unity, the updating rule is the NG one and the opposite is obtained canceling the h_{NG} term, it results the batch SOM one, with the difference of the modified ranking kernel. In

fact, values of γ over 80 usually make the h_{NG} tend to the unity and that is the reason why the algorithm behaves in a SOM way for high values of γ .

This algorithm is tested and discussed in [8].

III. SUPERVISED SOM-NG ALGORITHM

Once the previously developed SOM-NG algorithm was presented, a supervised learning rule will be added. The aim is to approximate an unknown scalar field $f(v)$, defined in a multidimensional input space with the following expression:

$$\tilde{f}(v) = y_{i*} + a_{i*} \cdot (v - w_{i*}) \quad (10)$$

where y_{i*} is a reference value in the position of w_{i*} in the input space and a_{i*} is the gradient in the Voronoi region defined by w_{i*} , which is the best matching unit for input vector v .

The energy cost function for Supervised SOM-NG is:

$$E_S = \sum_{i,j} h_{SOM} \cdot h_{NG} \cdot \left(f(v_j) - \hat{f}_i(v_j) \right)^2 \quad (11)$$

where $\hat{f}_i(v_j) = y_i - a_i \cdot (v_j - w_i)$ is the approximation for data vector j using prototype vector i .

As it was done with w_i , Newton's method will be employed to calculate the learning rules for the estimation parameters:

$$\Delta y_i = -J_E(y_i) \cdot H_E^{-1}(y_i) \quad (12)$$

$$\Delta a_i = -J_E(a_i) \cdot H_E^{-1}(a_i) \quad (13)$$

Both Jacobian and Hessian matrices are calculated with respect of y_i :

$$J_E(y_i) = -2 \cdot \sum_j h_{SOM} \cdot h_{NG} \cdot \left(f(v_j) - \hat{f}_i(v_j) \right) \quad (14)$$

$$H_E(y_i) = 2 \cdot \sum_j h_{SOM} \cdot h_{NG} \quad (15)$$

As the variation term for the Voronoi region $a_i \cdot (v_j - w_i)$ is symmetrical around the region center w_i , the whole term can be dispensed and the increment becomes:

$$\Delta y_i = \frac{\sum_j h_{SOM} \cdot h_{NG} \cdot (f(v_j) - y_i)}{\sum_j h_{SOM} \cdot h_{NG}} \quad (16)$$

The following updating rule is obtained:

$$y_i = \frac{\sum_j h_{SOM} \cdot h_{NG} \cdot f(v_j)}{\sum_j h_{SOM} \cdot h_{NG}} \quad (17)$$

The same process is carried out with a_i :

$$J_E(a_i) = -2 \cdot h_{SOM} \cdot h_{NG} \cdot \vec{d}_{ji} \cdot \left(f(v_j) - \hat{f}_i(v_j) \right) \quad (18)$$

$$H_E(a_i) = -2 \cdot h_{SOM} \cdot h_{NG} \cdot \left(\vec{d}_{ji} \cdot \vec{d}_{ji} \right) \quad (19)$$

The increment for a_i is:

$$\Delta a_i = \frac{\sum_j h_{SOM} \cdot h_{NG} \cdot \vec{d}_{ji} \cdot (f(v_j) - \hat{f}_i(v_j))}{h_{SOM} \cdot h_{NG} \cdot (\vec{d}_{ji} \cdot \vec{d}_{ji})} \quad (20)$$

Since expression (20) cannot be simplified, an increment rule is obtained instead of an absolute updating rule.

IV. PROCEDURE OF THE ALGORITHM

The prototype vectors and the estimation parameters are randomly initialized with small random values within the interval (0, 0.1).

```

begin initialize randomly prototype vectors  $w_i$  and
estimation parameters  $y_i$  and  $a_i$ 
determine the ranks of the map units  $s(r_{i*}, r_i)$ 
calculate the neighborhood radius  $\sigma(t)$  for every epoch
according to (3)
do for each epoch
    determine the ranks of the prototypes  $k(v_j, w_i)$ 
    calculate both kernels using (1) and (2)
    update the prototypes  $w_i$  using (9)
    update the estimation values  $y_i$  by means of (17)
until the maximum number of epochs
determine the ranks of the prototypes  $k(v_j, w_i)$ 
do for each epoch
    calculate both kernels using (1) and (2)
    modify the estimation vectors  $a_i$  as in (20)
until the maximum number of epochs
end
    
```

During the first loop, prototype vectors w_i and estimation values y_i are distributed along the input space to make the best possible fit to the data. In the second loop, gradient vectors a_i are estimated keeping constant the previously calculated parameters w_i and y_i .

The first loop includes an inner loop that calculates the distance from every prototype to each data vector and sorts them into the distance ranking $k(v_j, w_i)$, and after it the distance ranking is calculated again. This result is constant as the prototype vectors w_i are not modified in the second loop. In the second loop the gradient vectors a_i are approximated using the corresponding neighborhood radius for each epoch, starting with σ_{t0} again.

The computational cost of the algorithm itself is approximately linear with the number of data vectors and the map size, but it also has to be taken in care the complexity of the operations, specially the matrix multiplication.

V. EXPERIMENTAL TESTING

The quality of the proposed algorithm was compared with the well known SOM and NG [10] algorithms in their batch versions. The comparison was done using different values of the topology preservation constant γ , each one of them with a representative range of square output maps. Experiments were

repeated 20 times with a previously generated set of values as initialization to ensure that all the differences are produced by the algorithms and not by external causes.

The first measure is the quantization error q_e defined as the mean distance from a data vector to its best matching unit according to

$$q_e = \frac{\sum_{j=1}^N \|v_j - w_{j*}\|}{N} \quad (21)$$

where N is the number of data vectors.

Topographic preservation is measured using the topographic error proposed in [11]. The topographic error is defined as the proportion of data whose two best matching units are not adjacent in the output map, mathematically defined as:

$$t_e = \frac{1}{N} \sum_{j=1}^N u(v_j) \quad (22)$$

where $u(v)$ is equal to 1 when the best and second best matching units are non-adjacent. Otherwise it is equal to zero. In this work 2-neighborhood measure is used, for rectangular maps it means each unit considers neighbors all the adjacent units, including the diagonal ones.

Finally, and most important, the estimation ability is measured by its root mean square error according to (23).

$$RMSE = \sqrt{\frac{1}{N} \sum_{j=1}^N (f(v_j) - \hat{f}(v_j))^2} \quad (23)$$

All the non-numerical attributes and the missing values were removed before training. Data was normalized to zero mean and unitary standard deviation for every variable. For each data set a training data collection was created and the full collection was used to calculate the quality measures q_e , t_e and $RMSE$.

Estimation error for SOM was calculated using a normal training including all input and output variables together. The final value for the output variable is considered to be the estimation for any input vector in the Voronoi region.

For comparison purposes, new values will be calculated for errors. Quantization and estimation errors will be normalized with the expressions:

$$q'_e = \frac{q_e|_{SOM-NG}}{q_e|_{NG}} \quad (24)$$

$$RMSE' = \frac{RMSE|_{SOM-NG}}{RMSE|_{NG}} \quad (25)$$

where q'_e and $RMSE'$ are the relative quantization error and the relative root mean square error respectively and $error|_{algorithm}$ specifies which error is calculated and the algorithm that produces it. A value of 1 for any of them represents that the error is the same for the hybrid algorithm, i.e., the tested one, and NG, i.e., the reference algorithm.

Topographic comparative error is calculated using:

$$t'_e = t_e|_{SOM-NG} - t_e|_{SOM} \quad (26)$$

where t'_e is the comparative topographic error, that represents how worse the topographic preservation is in the hybrid algorithm compared to SOM results. Values of t'_e are within the range $[-1, 1]$, where $t'_e > 0$ means the hybrid algorithm is not as ordered as SOM and $t'_e < 0$ means the opposite.

A. Concrete Passive Strength data set

This data set contains lab measurements of compressive strength for different concrete types used in [12] and it was obtained from UCI Machine Learning Repository [13]. This data set is used to measure the quality of the trained maps in non-projected data. In this experiment the dimensionality reduction and the map projection is not taken in care, but the influence of parameter γ over the quality measures is studied. Training data was created using a representative fraction of data vectors uniformly distributed along the whole data set.

Both q_e and t_e are represented in Figure 1 for three different map sizes. Values of γ lower than the unity have similar behaviour, with low quantization error and very high topographic error. On the opposite side, high values of γ have higher quantization errors and lower topographic ones. Intermediate values, like $\gamma = 10$, offer good equilibrium between both errors, being close as good in quantization as low values and also close to high values in topographic preservation.

The most interesting value is $\gamma = 10$ so it is going to be expanded to different map sizes. Results are shown in Figure 2.

This data set has a high dependence on initialization values because data points are one single cluster in the input space. This means that all the algorithms have a high deviation from their average value for such a big number of experiments. In Figure 2, both quantization and topographic errors have an acceptable balance as it was expected from that value of γ . It also can be seen that in average terms the estimation has a similar accuracy in comparison to the NG approach. Standard deviations were slightly lower in the proposed algorithm than in SOM for most of the calculated map sizes with $\gamma = 10$, so it can be considered robust enough.

In Figure 3, the estimation error is shown for three examples of map size and several values of γ . Low values of γ offer less accurate estimations and intermediate values reach good estimation capabilities while keeping a good trade-off with the other two quality measures.

B. Trigonometric function

To study the estimation capability of the proposed algorithm in topologically ordered data, a trigonometric function is going to be estimated with SOM, NG and hybrid SOM-NG algorithms. The selected function is:

$$z = f(x, y) = \sin(x + y) \cdot \cos(x \cdot y) \quad (27)$$

The training data set was created in $(x, y) \in [-4, 4] \times [-4, 4] \subset \mathbb{R}^2$ using a coarse mesh in the center of the region and a fine one close to the limits. The function is represented in Figure 4, just for illustrative purposes. The test data was

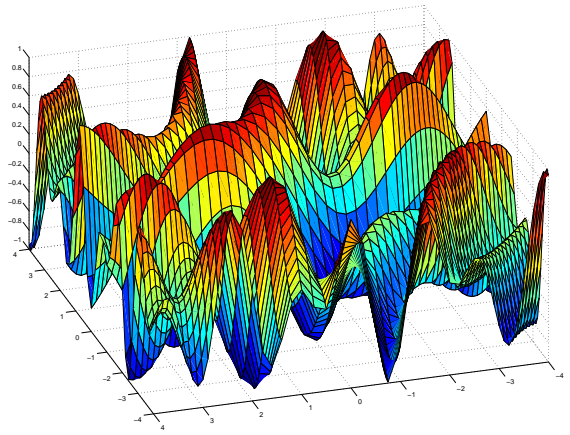


Figure 4. Trigonometric function to be estimated

uniformly arranged within the whole function domain, using a mesh with a step of 0.01 in both x and y .

Quantization error for this data set is different, as data is a discretized plane and SOM algorithms have better performance. If q_e is calculated for the training data NG still has the lowest quantization error, but for the test data set more ordered algorithms have better values, so SOM and high values of γ in the hybrid algorithm offer better quantization than NG.

Even in this circumstance, the hybrid algorithm has a good result, between the NG and SOM algorithms, as it can be seen in Figure 6. This special situation is good for the purpose of checking the algorithm, as it tends to get a good results because of the mixture of behaviours between SOM and NG.

In this data set, topographic preservation is similar to SOM, having values close to 0 in most of cases and bunch of outliers due to some different initializations that cause zonal disorders in one of the algorithms.

Estimation quality is good as it can be seen in the estimation error box-plot, looking for the quality of the best algorithm in estimation as it did in quantization. Estimation absolute results for three map sizes and different values of γ are shown in Fig 7. Best estimations are obtained for $\gamma = 0.2$, at the cost of having a very high topographic error, estimations obtained with values of γ greater than 5 are slightly worse, but keeping an acceptable topographic order.

VI. CONCLUSION

The hybrid SOM-NG algorithm has been tested with additional data sets to the previous work [8] and the results regarding data approximation and topographic preservation are confirmed.

The algorithm was improved with supervised learning using linear approximation for every Voronoi region. The value of γ has great influence over the final training.

Small values of γ , i.e., lower than 1, have a tendency towards NG behaviour, topographic preservation is very low and quantization is close to reach optimal results. The main

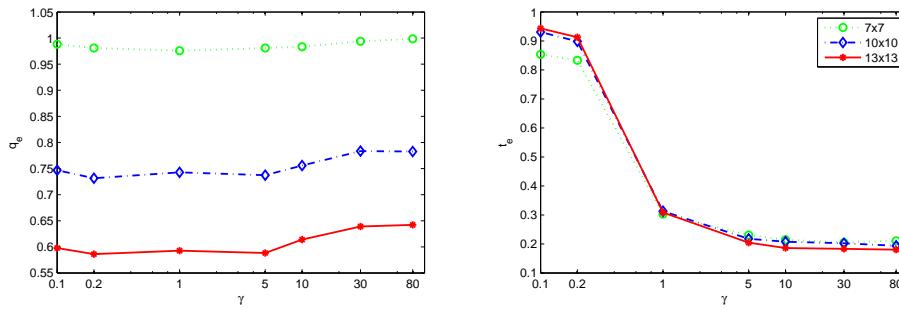


Figure 1. q_e and t_e versus γ for three different map sizes using Concrete data set

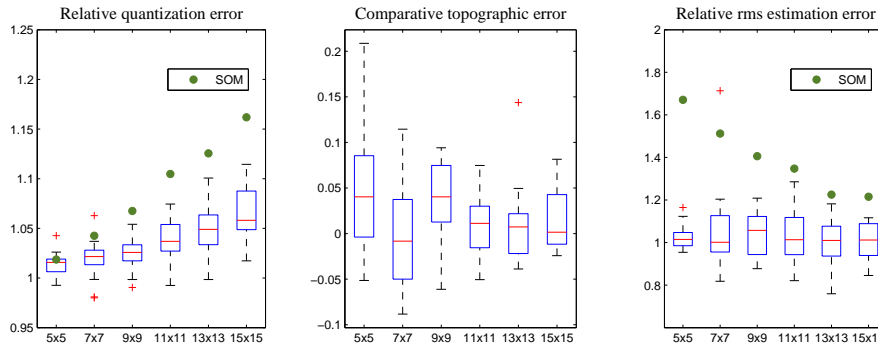


Figure 2. Concrete Passive Strength relative errors versus map size for $\gamma = 10$

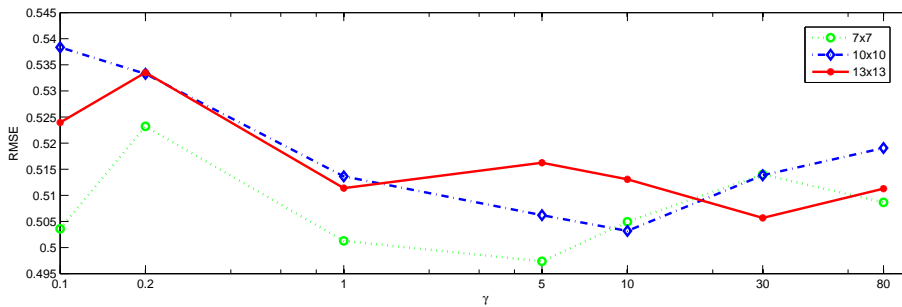


Figure 3. RMSE versus γ for three different map sizes for Concrete data set

drawback of using this learning condition is the lack of robustness. Low values of γ reduce the cooperation between neurons, as if the neighborhood radius had a smaller value, and initialization has a greater influence on the final results. If topographic preservation is not necessary, NG is a simpler and better choice.

The result is similar to a SOM training for high values of γ , having very small differences for any value over 80. Topographic preservation is very good, with some differences with SOM because of the ranking based kernel. The estimation done using the network as function approximator is satisfactory in comparison with the NG approach.

During the experiments the neighborhood function based on the modified ranking $s(r_{i*}, r_i)$ has different influence over the rest of the units than the usual squared distance. It is

important to realize that the neighborhood radius for the squared euclidean distance measures *how far* does the *bm* affect other units in the cooperative phase, while in the ranking it means *how many* units does it affect.

The most interesting feature of this algorithm is the use of intermediate γ values where a good trade-off solution is obtained. Relative measures, like the ones presented in Section V, demonstrate that close to optimum results can be achieved with a single algorithm.

REFERENCES

[1] T. Kohonen, *Self Organizing Maps*. Springer, 2001.
 [2] J. L. Buessler, J. P. Urban, and J. Gresser, "Additive composition of supervised self-organizing maps," *Neural Processing Letters*, vol. 15, no. 1, pp. 9–20, 2002.

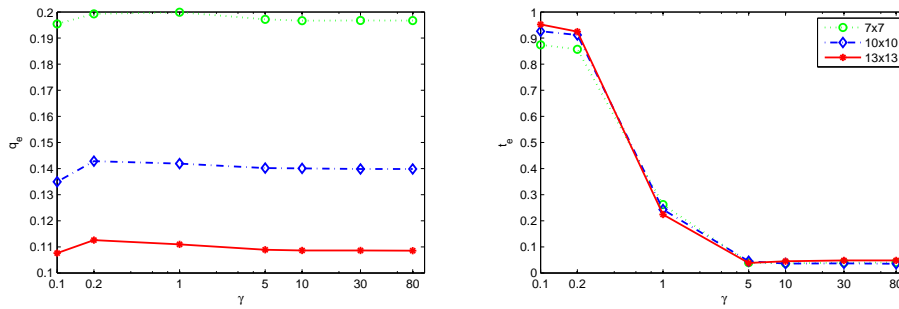


Figure 5. q_e and t_e versus γ for three different map sizes training with the trigonometric function

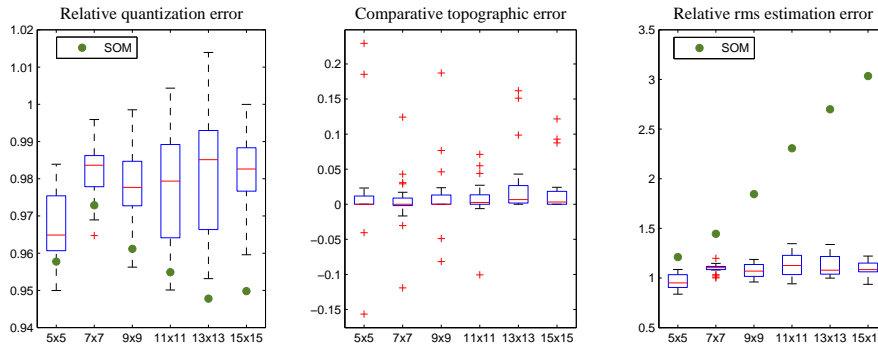


Figure 6. Trigonometric function relative errors versus map size with $\gamma = 10$

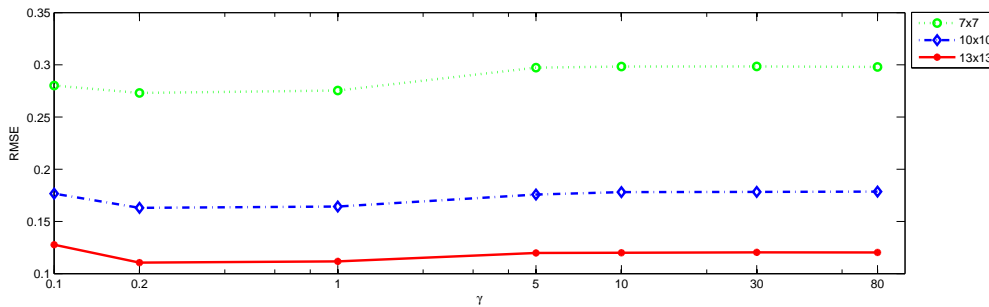


Figure 7. RMSE versus γ for three different map sizes approximating the trigonometric function

[3] J. Göppert and W. Rosenstiel, "The continuous interpolating self-organizing map," *Neural Processing Letters*, vol. 5, no. 3, pp. 185–192, 1997.

[4] G. J. Chappell and J. G. Taylor, "The temporal kohonen map," *Neural Networks*, vol. 6, no. 3, pp. 441 – 445, 1993.

[5] T. Koskela, M. Varsta, J. Heikkonen, and K. Kaski, "Recurrent SOM with local linear models in time series prediction," in *In 6th European Symposium on Artificial Neural Networks*, pp. 167–172, D-facto Publications, 1998.

[6] M. Varsta, J. Heikkonen, J. Lampinen, and J. Millán, "Temporal Kohonen map and the recurrent self-organizing map: analytical and experimental comparison," *Neural processing letters*, vol. 13, no. 3, pp. 237–251, 2001.

[7] T. Martinetz, S. Berkovich, and K. Schulten, "'neural-gas' network for vector quantization and its application to time-series prediction," *IEEE Transactions on Neural Networks*, vol. 4, pp. 558 –569, July 1993.

[8] I. Machón-González, H. López-García, and J. Calvo-Rolle, "A hybrid batch SOM-NG algorithm," in *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pp. 198–202, 2010.

[9] M. Cottrell, B. Hammer, A. Hasenfuß, and T. Villmann, "Batch neural gas," in *International Workshop on Self-Organizing Maps (WSOM 2005)*, pp. 275–282, 2005.

[10] M. Cottrell, B. Hammer, A. Hasenfuß, and T. Villmann, "Batch and median neural gas," *Neural Networks*, vol. 19, no. 6-7, pp. 762–771, 2006.

[11] K. Kiviluoto, "Topology preservation in self-organizing maps," in *IEEE International Conference on Neural Networks, 1996*, vol. 1, pp. 294 –299 vol.1, 1996.

[12] I.-C. Yeh, "Modeling of strength of high performance concrete using artificial neural networks," *Cement and Concrete Research*, vol. 28, no. 2, pp. 1797–1808, 1998.

[13] A. Frank and A. Asuncion, "UCI machine learning repository," <http://archive.ics.uci.edu/ml>, Sept. 2010.

Building Virtual Private Clouds with Network-aware Cloud

J. Soares^{1,2}, J. Carapinha¹, M. Melo^{1,2}, R. Monteiro^{1,2}

¹Department of Exploratory Innovation
Portugal Telecom Inovação
Aveiro, Portugal

{ joao-m-soares, jorgec, marcio-m-melo, romeu-r-monteiro }@ptinovacao.pt

Susana Sargento²

²Instituto de Telecomunicações
University of Aveiro
Aveiro, Portugal
susana@ua.pt

Abstract— Cloud computing presupposes on-demand network access to pool of computing resources. However, network access through the WAN is usually not compliant with any kind of service guarantees, including reliability, security and performance. In this work, two types of network services able to fulfill cloud requirements are presented. In addition, an extension of the Virtual Private Cloud concept is proposed by integrating these network services. Managing cloud and network resources in an integrated way is a need and an obvious challenge, thus resource management in such environment is a major focus. We identify the main inherent challenges in resource management and how they can be overcome. Further, an experimental platform is presented, along with a preliminary analysis of results.

Keywords— cloud computing; cloud networking; virtual private cloud; network-as-a-service; connectivity-as-a-service.

I. INTRODUCTION

Today, the proliferation of broadband access gives users the possibility to use services available directly through the Internet, which represents a change of the paradigm for using applications and communicating, thus popularizing the so-called Cloud Computing (CC).

CC brings with it requirements at two different levels: at the cloud level, i.e. data centers; and at the network level, where required levels of performance, reliability and security must be guaranteed.

So far, the cloud and the network have been seen as two separate entities in this picture, with the network playing a relatively minor role, mostly as provider of connectivity between the cloud resources and the user premises. We argue that, to provide assured levels of performance to cloud services, cloud and network resources need to be provisioned, managed, controlled and monitored in an integrated way.

There are several reasons for the integration of network and cloud. First, the establishment of Service Level Agreements (SLA) is essential to encourage customers, particularly enterprises, to adopt cloud services. Today, the lack of reliability and performance guarantees is one of the main obstacles against the widespread use of cloud services. It is clear that these SLAs can only be implemented through network integration. Just like the Wide Area Network (WAN) component of enterprise networks is usually based on reliable managed network services such as Virtual Private Networks (VPNs), rather than the public Internet, there is no

reason to believe that future enterprise cloud services will require a lesser degree of reliability and performance guarantees from the network.

Secondly, it is essential that cloud properties such as elasticity and self-provisioning be also extended to network resources. Quite often, expanding or reducing cloud resource capacity, or provisioning new cloud resources, requires a corresponding reconfiguration of network resources, e.g. bandwidth admitted into the network. Today, by contrast, reconfiguration of network services is supposed to be relatively infrequent and usually involves a significant amount of manual effort.

Thirdly, the dynamism of the cloud will often require live migration of resources (e.g. from a local enterprise data center to the cloud, or between two different sites of the cloud service provider) without interrupting the operating system and any noticeable impact on the running application. This requires IP addressing to remain unchanged after migration and all relevant QoS, security and traffic policies applied on network equipment (e.g. routers, switches, firewalls) to be adapted appropriately in real time.

For the reasons stated above, it is clear that next generation of cloud services must handle network and cloud resources in an integrated way. This paper presents the concept of Cloud Networking (CN) to achieve this integration in the context of virtual private environments, and its resource management aspects to develop an integrated view and allocation of both network and cloud resources.

This paper is organized as follows. Section II summarizes relevant work in the area and how this work intends to progress, and section III presents how the concept of CN can be applied in the context of virtual private environments. Further, section IV provides an overview of the resource management challenges that arise in a CN environment. The experimental platform that embraces this approach is described in section V. Section VI describes how a Virtual Private Cloud (VPC) request is instantiated and also presents the prototyping results achieved so far to demonstrate the concept of CN. Finally, section VII provides general conclusions and indicates directions for future work.

II. RELATED WORK

Based on recent trends and evolutions, it is clear that the network will play a key role in the provisioning of cloud

computing services, by giving the necessary guarantees to access the cloud. The importance of this role will be increasingly evident. In this area, some research works have been presented in the literature, such as [2] where the critical impact of network performance on the applications is shown and an extension of [3] is presented based on a platform for provisioning of virtual infrastructures, to extend the traditional cloud paradigm to network provisioning. [4] presents a software-based network resource management system for VPCs, able to handle heterogeneous network equipment and proposes a virtual network point for multipoint network provisioning.

The European Commission, through the Seventh Framework Programme (FP7), has also been supporting research in this area, FP7 Projects SAIL (Scalable & Adaptive Internet soLutions) [5] and GEYSERS (Generalised Architecture for Dynamic Infrastructure Services) [6] are two relevant examples.

From the industry side, both standardization bodies and enterprise efforts have highlighted the need for cloud and network resources to be handled together. Verizon has been working on the extension of VPNs for Private Clouds and an Internet Engineering Task Force (IETF) Internet-Draft has been released on this matter [7]. Meanwhile, IBM already offers enterprises a cloud data backup supported by Verizon’s VPN services.

Although there are works on management addressing VPCs and others addressing virtual networks, there are few addressing the management of both in an integrated way. This paper addresses this subject and proposes an extension of the VPC concept to also provide WAN guarantees. A platform able to provide this VPC model is also presented in the paper.

III. CLOUD NETWORKING FOR VIRTUAL PRIVATE CLOUDS

Virtualization has been the key enabler of agility in data centers, which in turn led to the emergence of CC. The fundamental breakthrough offered by virtualization is the separation of operating systems (OSs) and applications from the underlying physical infrastructure.

Applying the same concept to networks has been often advocated – by decoupling networks from infrastructure through virtualization, it should be possible to establish and reconfigure (virtual) networks with great flexibility, nearly on-demand. Network virtualization has been explored by different research initiatives in multiple contexts and application scenarios. The idea of on-demand provisioning of network services has been demonstrated in practice [9].

Providing the network infrastructure with the ability to match the dynamism of the cloud would be required to overcome the problems and limitations identified in the previous section. From this point of view, network virtualization would be the perfect companion for virtualization in the data center, in order to build seamless end-to-end elastic and agile offer of cloud services.

A virtual network (VN) is supposed to fully replicate the behavior of a physical network, from all points of view. While this replication may be useful in some cases (e.g.

when the customer is itself a service provider), in most cases the effort of managing a VN is a burden that customers would prefer to avoid.

Thus, just like the CC service model defines three basic services - Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) - we propose a similar approach for networks. From this perspective, we define two types of network services (Figure 1): *Network as a Service* (NaaS) and *Connectivity as a Service* (CaaS). NaaS allows the user to request a network by specifying precisely the network topology, link bandwidths, routers’ computing capacities, routing protocols, as well as possibly other features (e.g. physical location, security properties). As for CaaS, the user is provided with the ability to define, just like in VPNs, a set of customer edge equipments (CEs) (e.g. enterprise sites and possibility the cloud CE, however this latter does not necessarily needs to be defined) and certain characteristics such as bandwidth at ingress/egress points and routing protocols between the CE and the provider edge equipment (PE). Everything that runs inside the service provider network domain is not visible to the customer.

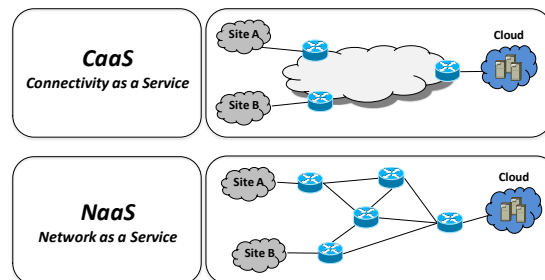


Figure 1. Network layering model.

Both network service options can be materialized in multiple ways. A fully virtualized network is the “natural” way to materialize NaaS, whereas managed VPNs (e.g. BGP/MPLS VPNs) are a typical example of CaaS.

In this paper, we propose a solution that is able to embrace both types of network service.

At this point it is useful to define the concept of VPC. In [8] a VPC is defined as “a combination of cloud computing resources with a VPN infrastructure to give users the abstraction of a private set of cloud resources that are transparently and securely connected to their own infrastructure”. In the context of this paper, we propose to generalize this concept, in order to embrace any kind of private network service, either materialized as a VPN, or as more advanced service types, including those based on fully virtualized networks. Moreover, a network service should allow the handling of network resources (e.g. bandwidth, add/remove a customer site) with a certain level of freedom in order for it to keep up with the cloud.

In order to address the coupling of both cloud and network services, the concept of CN was put forward. Similarly to CC, CN has no standard definition, but we can say that it goes beyond classical networks, encompassing on-demand provisioning i.e. scalability, guaranteed performance, self-healing and extensible management. So

far, no real attempt to merge networking and cloud resources in a common framework has actually taken place.

We pursue the concept of CN by envisioning a unified management framework for computing and communication, where the network operator can provide simultaneously the network and cloud resources (IaaS), in an integrated approach, optimizing overall resource allocations by considering network and computing resources as a unified whole. In this work, network services are materialized in VNs, however we do not exclude the possibility of other network approaches (e.g. VPN, OpenFlow).

In the following section we identify what we consider to be the most important challenges to CN that resources management raises.

IV. RESOURCE MANAGEMENT IN CLOUD NETWORKING

Bringing together network and CC resources so that users can access services in the cloud with guaranteed performance and reliability raises several challenges. The discovery, allocation, adaptation and re-optimization of resources, addressing simultaneously both network and cloud resources, are the main inherent challenges of resource management in CN. The management of these resources lays upon concepts of virtual resource mapping in the physical infrastructure with self-organized reconfiguration of resources, devices and associated network, according to the services and user requirements, policies (with respect to e.g. location) and changes in the infrastructure.

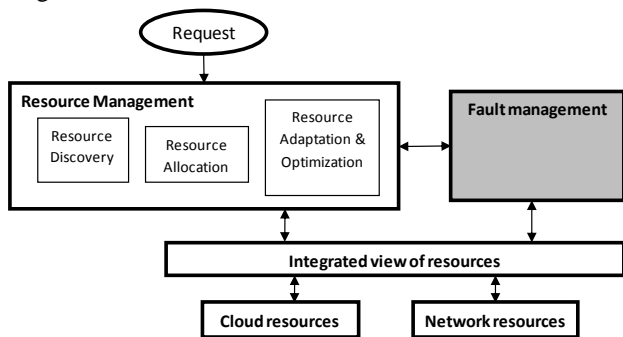


Figure 2. Cloud Networking Management diagram.

In Figure 2 we present a management block diagram composed by three main blocks: the *Resource Management* block (RM); the *Fault Management* block (FM); and an underlying block entitled *Integrated view of resources* (IVR). This work is focused on the RM and IVR. The former is composed by three sub-blocks: the *Resource Discovery* block (RD); *Resource Allocation* block (RA); and the *Resource Adaptation & Optimization* block (RAO). These sub-blocks will be detailed ahead. As for the latter, the IVR, it has the purpose of providing the upper blocks with the domain agnostic ability to view and interact with resources, whether they are cloud or network resources. Regarding the FM, it is illustrated in the picture to facilitate the interpretation of the management system, mainly regarding its interaction with the RAO sub-block.

A. Resource Discovery (and Monitoring)

A fundamental requirement in virtualized environments is the integrated view of the existing physical and virtual topologies, the resources’ characteristics, as well as the status of all network elements and links. This knowledge can be provided either by a centralized or by a distributed approach [10].

Today the cloud, i.e. data centers, and the operator’s network are two distinct domains which CN aims at integrating. However, there are boundaries that cannot be crossed as these domains will not be willing to share full information about their domain. In this approach, we assume to have access to network information such as topology and physical resources, as well as the ability to retrieve information on the virtual resources that the physical resources may host. On the data center side, we do not expect to have such detailed information, we rather expect to see a data center as a single node in the network with unlimited capacity and a set of associated information elements (similar to today’s cloud services, e.g., instance types, available OSs, pricing, plus location).

B. Resource Allocation

Virtual resources should be provisioned and placed in an optimal location according to the available resources at the time of the request, based on a number of possible criteria from both cloud and network, e.g.: type of VMs and possible restriction on location of these VMs; latency, bandwidth topology, geographical places where users will access the service, and other possible restrictions.

In order to map resources, a combined mechanism, able to perform balanced decisions taking into account the abovementioned requirements of both network and cloud resources, is needed. This mechanism must be able to determine a possible solution, i.e., physical hosts able to allocate the cloud resources which, at the same time, can have an associated network service able to fulfill the requirements in the access to the cloud.

C. Resource Adaptation and Optimization

With the dynamism of the cloud, reconfigurations and re-optimizations become common operations, whether to cater for possible side effects of new virtual resources being instantiated and existing virtual resources being resized, released or migrated, business policies, or triggered by unexpected events (e.g. node or link failure). These unexpected events are triggered by the FM which is responsible for monitoring the resources, detecting faults and collecting performance metrics.

Depending on the specific environment, actions can be taken at different levels: in the cloud, in the network, or in both. Thus, mechanisms are required for extending or moving cloud resources to other data centers, creating new network paths and reconfiguring existing ones (need for more bandwidth, less latency, failure, load balancing network resources). Those algorithms must decide on (1) when to reconfigure and (2) how to reconfigure. These decisions must be done based on information provided by the FM, or by an explicit request from the user.

Towards this aim, the Network Virtualization System Suite (NVSS) presented in [9] was extended with the control and management of the Suite enabling now cloud (IaaS) and network services (NaaS and CaaS) to be provisioned together to meet the user’s requirements, apart from its original feature, the deployment of VNs. The implementation work presented in this paper specifically targets the challenge of resource discovery (and monitoring) of cloud resources and resource allocation of both cloud and network. The next section gives an overview on the evolved NVSS.

V. NETWORK-AWARE CLOUD SYSTEM SUITE

The Network-aware Cloud System Suite (NCSS) is a platform that provides integrated deployment and management of cloud and network resources in a single tool. This platform is an evolution of NVSS [7], an experimental platform that provides VN design, embedding, creation, discovery, monitoring, and management. NCSS extends NVSS in two fundamental ways: it handles cloud resources (rather than just network resources) and CaaS (rather than just NaaS).

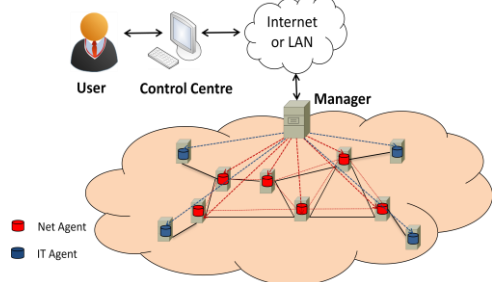


Figure 3. NCSS Architecture

NCSS is composed of 3 software modules: the Agent module, the Manager module and the Control Centre module. Their hierarchical decomposition can be analyzed on Figure 3. The Agent module is designed to run on network nodes (‘Net Agent’), as well as on those acting as computing nodes (‘IT Agent’), in order to act upon them and periodically gather data from them. The two types of Agents, besides interacting with each other, receive and send requests to the Manager, which is a centralized entity in charge of aggregating all Agents’ knowledge and sending them commands. Additionally, the Manager also communicates with the Control Centre, which is the user’s front-end, and provides him with graphical and simple to use VN creation, management, and monitoring functionalities.

A. Functionalities

The NCSS platform provides a set of main functionalities: distributed network and cloud resource discovery, network and cloud mapping and creation, network and computing monitoring, and network and cloud resource management. These functionalities are described below.

1) Distributed Network and Cloud Discovery.

Network and cloud resource discovery is not only an administrator’s utility that provides a fast and easy way of viewing how the cloud resources and the network resources are been used and where they are been consumed, but it is

also fundamental when embedding new cloud and network resources, since the embedding process requires an accurate and up-to-date view of the substrate and currently running cloud and network resources.

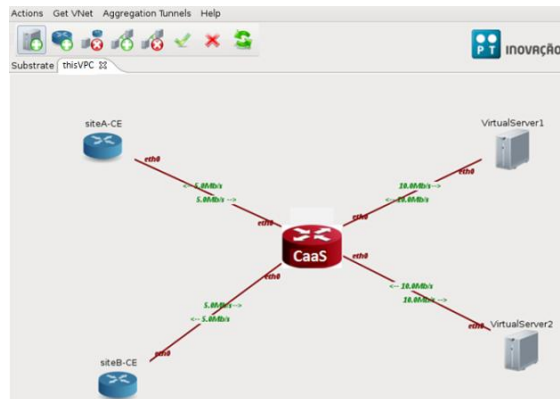


Figure 4. Control Centre - VPC Requirement Example

2) Network and Cloud Mapping and Creation

The Control Centre module provides the user with means to create and embed new cloud and network resources in runtime. By selecting and placing either cloud or network resources, i.e. servers or routers, on the platform GUI and by connecting them with links, as depicted in Figure 4. The user can specify both cloud and network resource capabilities, CPU, RAM amount, location, number of interfaces and also perform network addressing configurations.

The final step in creating a new set of cloud and network elements is to commit it to the Manager, which will then map it in the physical infrastructure.

The embedding problem of cloud and network elements is a complex one, which requires a trade-off between computation time and embedding optimization. In order to lower the computational requirements, a heuristic mapping algorithm was developed, which aims to embed both types of elements taking into consideration both the load of physical links and load of network and computing nodes.

3) Cloud and Network Monitoring

Dynamic resource monitoring is fundamental to provide an accurate view of the state of both types of resources and to quickly react to failures or configuration problems. The implemented monitoring functions periodically update the information on resources; therefore it is possible to quickly identify diverse situations, such as failures and high resource usage.

4) Cloud and Network Management.

The management feature provides functionalities like the change of the resource state (i.e., reboot, shutdown, suspend or power up), the change of the assigned RAM memory in runtime and the deletion of either a single resource or a complete set of resources, which greatly simplifies the administrator work.

The following section describes the process of establishing a VPC using the NCSS.

VI. VIRTUAL PRIVATE CLOUD ESTABLISHMENT & EVALUATION

The establishment of a VPC using the NCSS can be supported by a NaaS or CaaS. The process of establishing a VPC, from the moment a user requests a VPC until the moment it is ready to be enforced in the physical infrastructure, will be detailed in this section. Two VPC requests will be considered, one for NaaS and another for CaaS. In addition, results on the request’s processing time are presented. Note that these results do not intend to perform any comparison between a VPC with NaaS and CaaS since they are two different services.

The process of establishing a VPC supported by a NaaS service is divided in 6 main phases, as Figure 5 shows: *request formulation*; *request conversion*; *resource discovery*; *node mapping*; *link mapping*; *node and link embedding*. Phase 1 encompasses the formulation of a request, in which the user defines all resources: virtual servers (CPU, RAM and HDD); network topology and the characteristics of all nodes and links. The request is then sent to the *Manager* which first performs the conversion of the request from XML to a structured topology (phase 2). Moreover it is performed the resource discovery (phase 3), and the nodes and links are mapped (phase 4 and 5) using a mapping algorithm which considers both the occupation of physical nodes and links. Finally, the VPC is enforced (phase 6).

Based on the tools already available in the NCSS, which already supported NaaS, we developed the necessary mechanisms to support CaaS. The request of a CaaS service was done taking into consideration some aspects of the widespread VPN concept as example - PE, CE, fully mesh topology.

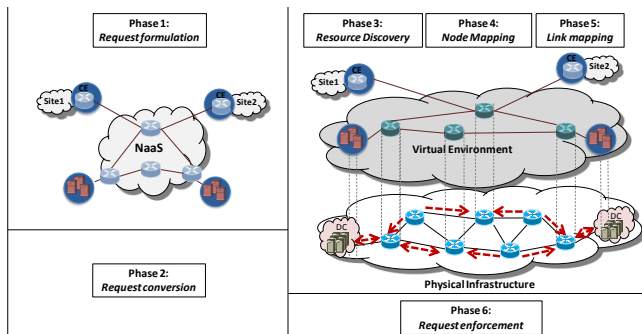


Figure 5. Virtual Private Cloud mapping process with NaaS.

The user is able to define CEs as if he was requesting a VPN, specify ingress and egress bandwidth requirements for each network endpoint (hose model), rather than specifying the requirements between all pairs of endpoints, as in NaaS. The user can drag and drop graphical depictions of routers and servers, which represent the sites and cloud resources of the network, and then connect them to a central graphical element representing an abstraction of the network (Figure 4). These connections contain information about the bandwidth from each element to and from the network. In the end the user just has to press the commit button and the CaaS information is processed and the request enforced. In the end, the user gets a VN which connects the customer sites and cloud resources, according to the user’s configuration parameters.

A VPC supported by CaaS releases the user from the NaaS complexity, but adds an intermediate process step. Nevertheless this does not imply an increase of time from the request to the enforcement moment.

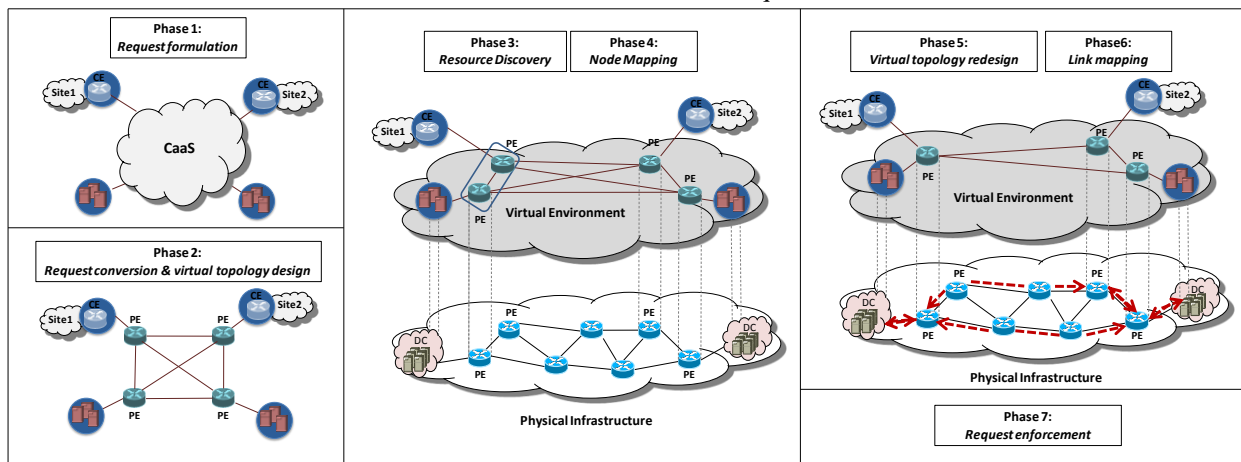


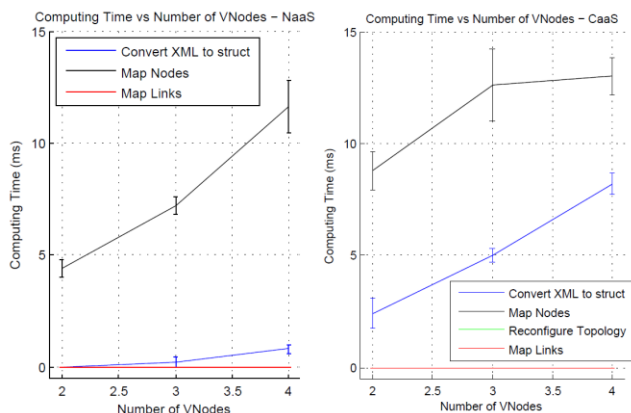
Figure 6. Virtual Private Cloud mapping process with CaaS.

As depicted in Figure 6, the process has 7 phases: *request formulation*; *request conversion* (includes the creation of the virtual topology which is not entirely defined); *resource discovery*; *node mapping*; *virtual topology reconfiguration*; *link mapping*; *node and link embedding*. First, the user configures the CaaS requirements on the GUI (phase 1). Once the *Manager* has received the request, it converts it to a topology structure, where a virtual PE router is connected to each site and cloud

resource, with the PE routers connected in full mesh (phase 2). The bandwidth of the links is set to the minimum necessary to fulfill the worst case hose-model requirements. Then resources are discovered (phase3), and virtual PE routers and cloud resources are mapped to physical PEs and datacenters according to the temporary topology (phase 4). Note that the set of candidate PE routers for a CE encompasses those located near that CE, which eases the mapping process. Phase 5 comprises the

virtual topology reconfiguration, so that virtual PEs mapped to the same physical PEs are joined together in one virtual PE. The resulting links are mapped onto the physical substrate in phase 6. In the end, all elements are enforced, phase 7.

To finalize, Figure 7 presents the time that the *Manager* takes to process a VPC request. The presented values are an average of 5 requests.



(a) Computing times for mapping NaaS&Cloud (b) Computing times for mapping CaaS&Cloud

Figure 7. Virtual Private Cloud mapping process with NaaS.

Figure 7 (a) shows the results for NaaS. The time to convert from the XML message to a structure increases with the number of nodes (0-1ms), but is still a small value when compared to the time to map the nodes (4.5-11.5ms).

As for the mapping with CaaS, Figure 7 (b), we can see an extra time stage, topology reconfiguration, which is not visible because it is at a 0ms value in both cases, and thus is below the red line. The conversion process takes a little longer with CaaS, 3-14ms, since there is the need to create the full topology. Node mapping values for CaaS are higher, 9-13ms. This might be explained by the fact that the node mapping process in CaaS is made using the temporary topology, which includes 1 PE router per site or server + 1 server element per virtual server. Times for topology reconfiguration in the CaaS are very small, which might be explained by the simple nature of this action, which mainly consist in removing some virtual nodes and links. Link mapping on both cases is always close to 0 (not perceptible since the out time unit while measuring was ms). This might be due to the small number of nodes in both VNs and substrate.

VII. CONCLUSION AND FUTURE WORK

A major limitation of CC is the lack of coordination between CC resource control and network resource control. To overcome this limitation, elasticity and agility of the cloud must be extended to the network infrastructure. In this sense we have associated the concept of VPC with network virtualization, allowing

cloud and network resources to be handled as a single set in a dynamic and flexible way.

Two network service models are proposed, CaaS and NaaS. The former roughly corresponds to the traditional managed network-based VPN paradigm. The latter provides a service which is functionally identical to a network. Both models should have a role to play in future networks. Moreover, we present a platform able to handle NaaS and CaaS services along with cloud resources.

One of the tasks that is still open for future work is the integration of BGP/MPLS VPNs in the platform as a CaaS, since VPNs are today in the market and represent the strongest short-time deployment possibility. The integration with the cloud using standardized application programming interfaces (e.g. Open Grid Forum Open Cloud Computing Interface) is also in the evolution roadmap.

ACKNOWLEDGMENT

The authors are thankful to the members of the SAIL project, particularly those involved in Work Package D, for the collaboration and fruitful discussions.

REFERENCES

- [1] P.Mell and T. Grance, "The NIST Definition of Cloud Computing (Draft)", National Institute of Standards and Technology, January 2011.
- [2] T. T. Huu, G. Koslovski, F. Anhalt, P. Vicat-Blanc Primet, and J. Montagnat. "Joint elastic cloud and virtual network framework for application performance optimization and cost reduction". Journal of Grid Computing (JoGC), pp. 27-47, 2010.
- [3] F. Anhalt, G. Koslovski, and P. Vicat-Blanc Primet. "Specifying and provisioning virtual infrastructures with HIPerNET". Int. J. Netw. Manag., pp. 129-148 May 2010.
- [4] T. Miyamoto, M. Hayashi, and K. Nishimura, "Sustainable Network Resource Management System for Virtual Private Clouds," Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on , vol., no., pp.512-520, Nov. 30 2010-Dec. 3 2010.
- [5] FP7 Project SAIL, <http://www.sail-project.eu/>.
- [6] FP7 Project GEYSERS, <http://www.geysers.eu/>.
- [7] So et al, "VPN Extensions for Private Clouds" IETF Internet Draft, February 2011.
- [8] T. Wood, A. Gerber, K. Ramakrishnan, P. Shenoy, J. Van der Merwe, "The Case for Enterprise-Ready Virtual Private Clouds", HotCloud' 09, 2009.
- [9] J. Nogueira, M. Melo, J. Carapinha, and S. Sargento, "Network Virtualization System Suite: Experimental Network Virtualization Platform", in TridentCom 2011, 7th International ICST Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities, April 2011
- [10] J. Nogueira, M. Melo, J. Carapinha, and S. Sargento, "A distributed approach for virtual network discovery," GLOBECOM Workshops (GC Wkshps), 2010 IEEE , vol., no., pp.277-282, 6-10 Dec. 2010
- [11] J. Nogueira, M. Melo, J. Carapinha, S. Sargento, "Virtual Mapping into Heterogeneous Substrate Networks", Computers and Communications (ISCC), IEEE Symposium, pp.438-444, 2011.