Advances in Intelligent Systems and Computing 429

Leszek Borzemski
Adam Grzech
Jerzy Świątek
Zofia Wilimowska *Editors*

Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I

Springer

Advances in Intelligent Systems and Computing

Volume 429

About this Series

The series "Advances in Intelligent Systems and Computing" contains publications on theory, applications, and design methods of Intelligent Systems and Intelligent Computing. Virtually all disciplines such as engineering, natural sciences, computer and information science, ICT, economics, business, e-commerce, environment, healthcare, life science are covered. The list of topics spans all the areas of modern intelligent systems and computing.

The publications within "Advances in Intelligent Systems and Computing" are primarily textbooks and proceedings of important conferences, symposia and congresses. They cover significant recent developments in the field, both of a foundational and applicable character. An important characteristic feature of the series is the short publication time and world-wide distribution. This permits a rapid and broad dissemination of research results.

More information about this series at http://www.springer.com/series/11156

Leszek Borzemski · Adam Grzech
Jerzy Świątek · Zofia Wilimowska
Editors

Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I

ISAT 2015

Springer

Editors
Leszek Borzemski
Faculty of Computer Science
and Management
Wrocław University of Technology
Wrocław
Poland

Jerzy Świątek
Faculty of Computer Science
and Management
Wrocław University of Technology
Wrocław
Poland

Adam Grzech
Faculty of Computer Science
and Management
Wrocław University of Technology
Wrocław
Poland

Zofia Wilimowska
Faculty of Computer Science
and Management
Wrocław University of Technology
Wrocław
Poland

ISSN 2194-5357 ISSN 2194-5365 (electronic)
Advances in Intelligent Systems and Computing
ISBN 978-3-319-28553-5 ISBN 978-3-319-28555-9 (eBook)
DOI 10.1007/978-3-319-28555-9

Library of Congress Control Number: 2016930056

Printed on acid-free paper

This Springer imprint is published by SpringerNature
The registered company is Springer International Publishing AG Switzerland

Preface

This four volume set of books includes the proceedings of the 2015 36th International Conference Information Systems Architecture and Technology (ISAT), or ISAT 2015 for short, held on September 20–22, 2015, in Karpacz, Poland. The conference was organized by the Department of Computer Science and Department of Management Systems, Faculty of Computer Science and Management, Wrocław University of Technology, Poland.

The International Conference Information Systems Architecture is organized by the Wrocław University of Technology from the seventies of the last century. The purpose of the ISAT is to discuss a state of the art of information systems concepts and applications as well as architectures and technologies supporting contemporary information systems. The aim is also to consider an impact of knowledge, information, computing, and communication technologies on managing the organization scope of functionality as well as on enterprise information systems design, implementation, and maintenance processes taking into account various methodological, technological, and technical aspects. It is also devoted to information systems concepts and applications supporting exchange of goods and services by using different business models and exploiting opportunities offered by Internet-based electronic business and commerce solutions.

ISAT is a forum for specific disciplinary research, as well as on multi-disciplinary studies to present original contributions and to discuss different subjects of today's information systems planning, designing, development, and implementation. The event is addressed to the scientific community, people involved in variety of topics related to information, management, computer, and communication systems, and people involved in the development of business information systems and business computer applications.

This year, we received 130 papers from 17 countries. The papers included in the four proceeding volumes published by Springer have been subject to a thoroughgoing review process by highly qualified peer reviewers. Each paper was reviewed by at least two members of Program Committee or Board of Reviewers. Only 74 best papers were selected for oral presentation and publication in the 36th International

Conference Information Systems Architecture and Technology 2015 proceedings. The final acceptance rate was 57 %.

Professor Peter Nelsen (Denmark) presented his keynote speech on Some Insights from Big Data Research Projects. He also organized the special session on the advances in methods for managing complex planning environments.

The conference proceedings are divided into four volumes and present papers in the areas of managing complex planning environments, systems analysis and modeling, finance, logistics and market, artificial intelligence, knowledge-based management, Web systems, computer networks and distributed computing, high performance computing, cloud computing, multi-agent systems, Internet of Things, mobile systems, service-oriented architecture systems, knowledge discovery, and data mining.

We would like to thank the Program Committee and external reviewers, essential for reviewing the papers to ensure a high standard of the ISAT 2015 conference and the proceedings. We thank the authors, presenters, and participants of ISAT 2015; without them, the conference could not have taken place. Finally, we thank the organizing team for the efforts this and previous years in bringing the conference to a successful conclusion.

September 2015

Leszek Borzemski
Adam Grzech
Jerzy Świątek
Zofia Wilimowska

ISAT 2015 Conference Organization

General Chair

Leszek Borzemski, Poland

Program Co-chairs

Leszek Borzemski, Poland
Adam Grzech, Poland
Jerzy Świątek, Poland
Zofia Wilimowska, Poland

Local Organizing Committee

Leszek Borzemski, Chair
Zofia Wilimowska, Vice-Chair
Mariusz Fraś, Conference Secretary and ISAT 2015 Website Administrator
Arkadiusz Górski, Katarzyna Gwóźdź, Technical Editors
Anna Kiłyk, Ziemowit Nowak, Agnieszka Parkitna, Technical Chairmen

International Program Committee

Witold Abramowicz, Poland
Dhiya Al-Jumeily, UK
Iosif Androulidakis, Greece
Patricia Anthony, New Zeland
Zbigniew Banaszak, Poland
Elena N. Benderskaya, Russia
Leszek Borzemski, Poland
Janos Botzheim, Japan

Patrice Boursier, France
Wojciech Cellary, Poland
Haruna Chiroma, Malaysia
Edward Chlebus, Poland
Gloria Cerasela Crisan, Romania
Marilia Curado, Portugal
Czesław Daniłowicz, Poland
Zhaohong Deng, China
Małgorzata Dolińska, Poland
El-Sayed M. El-Alfy, Saudi Arabia
Naoki Fukuta, Japan
Piotr Gawkowski, Poland
Manuel Graña, Spain
Wiesław M. Grudzewski, Poland
Adam Grzech, Poland
Irena Hejduk, Poland
Katsuhiro Honda, Japan
Marian Hopej, Poland
Zbigniew Huzar, Poland
Natthakan Iam-On, Thailand
Biju Issac, UK
Arun Iyengar, USA
Jürgen Jasperneite, Germany
Janusz Kacprzyk, Poland
Henryk Kaproń, Poland
Yannis L. Karnavas, Greece
Ryszard Knosala, Poland
Zdzisław Kowalczuk, Poland
Binod Kumar, India
Jan Kwiatkowski, Poland
Antonio Latorre, Spain
Gang Li, Australia
José M. Merigó Lindahl, Chile
Jose M. Luna, Spain
Emilio Luque, Spain
Sofian Maabout, France
Zygmunt Mazur, Poland
Pedro Medeiros, Portugal
Toshiro Minami, Japan
Marian Molasy, Poland
Zbigniew Nahorski, Poland
Kazumi Nakamatsu, Japan
Peter Nielsen, Denmark
Tadashi Nomoto, Japan
Cezary Orłowski, Poland

Michele Pagano, Italy
George A. Papakostas, Greece
Zdzisław Papir, Poland
Marek Pawlak, Poland
Jan Platoš, Czech Republic
Tomasz Popławski, Poland
Edward Radosiński, Poland
Dolores I. Rexachs, Spain
José S. Reyes, Spain
Leszek Rutkowski, Poland
Gerald Schaefer, UK
Habib Shah, Malaysia
Jeng Shyang, Taiwan
Anna Sikora, Spain
Małgorzata Sterna, Poland
Janusz Stokłosa, Poland
Remo Suppi, Spain
Edward Szczerbicki, Australia
Jerzy Świątek, Poland
Eugeniusz Toczyłowski, Poland
Elpida Tzafestas, Greece
José R. Villar, Spain
Bay Vo, Vietnam
Hongzhi Wang, China
Leon S.I. Wang, Taiwan
Jan Werewka, Poland
Thomas Wielicki, USA
Zofia Wilimowska, Poland
Bernd Wolfinger, Germany
Józef Woźniak, Poland
Roman Wyrzykowski, Poland
Jaroslav Zendulka, Czech Republic
Bernard Ženko, Slovenia

ISAT 2015 Reviewers

Patricia Anthony, New Zeland
Zbigniew Banaszak, Poland
Elena N. Benderskaya, Russia
Grzegorz Bocewicz, Poland
Leszek Borzemski, Poland
Janos Botzheim, Japan
Patrice Boursier, France
Krzysztof Brzostowski, Poland

Wojciech Cellary, Poland
Gloria Cerasela Crisan, Romania
Marilia Curado, Portugal
Mariusz Czekała, Poland
Grzegorz Debita, Poland
El-Sayed M. El-Alfy, Saudi Arabia
Stefan Forlicz, Poland
Mariusz Fraś, Poland
Naoki Fukuta, Japan
Piotr Gawkowski, Poland
Manuel Graña, Spain
Dariusz Gąsior, Poland
Adam Grzech, Poland
Irena Hejduk, Poland
Katsuhiro Honda, Japan
Zbigniew Huzar, Poland
Biju Issac, UK
Jerzy Józefczyk, Poland
Krzysztof Juszczyszyn, Poland
Yannis L. Karnavas, Greece
Radosław Katarzyniak, Poland
Grzegorz Kołaczek, Poland
Zdzisław Kowalczuk, Poland
Binod Kumar, India
Jan Kwiatkowski, Poland
Antonio Latorre, Spain
Arkadiusz Liber, Poland
Wojciech Lorkiewicz, Poland
Zygmunt Mazur, Poland
Pedro Medeiros, Portugal
Izabela Nielsen, Denmark
Peter Nielsen, Denmark
Tadashi Nomoto, Japan
Cezary Orłowski, Poland
Michele Pagano, Italy
George A. Papakostas, Greece
Marek Pawlak, Poland
Jan Platoš, Czech Republic
Tomasz Popławski, Poland
Dolores I. Rexachs, Spain
José S. Reyes, Spain
Gerald Schaefer, UK
Habib Shah, Saudi Arabia
Anna Sikora, Spain
Małgorzata Sterna, Poland

Janusz Stokłosa, Poland
Remo Suppi, Spain
Edward Szczerbicki, Australia
Jerzy Świątek, Poland
Elpida Tzafestas, Greece
José R. Villar, Spain
Tomasz Walkowiak, Poland
Hongzhi Wang, China
Leon S.I. Wang, Taiwan
Adam Wasilewski, Poland
Jan Werewka, Poland
Zofia Wilimowska, Poland
Bernd Wolfinger, Germany
Józef Woźniak, Poland
Jaroslav Zendulka, Czech Republic
Maciej Zięba, Poland
Bernard Ženko, Slovenia

ISAT 2015 Keynote Speaker

Professor Peter Nielsen, Aalborg University, Aalborg, Denmark
Topic: Some Insights from Big Data Research Projects

ISAT 2015 Invited Session

Advances in Methods for Managing Complex Planning Environments
Chair: Peter Nielsen, Denmark

Contents

Part I
Keynote Speech

Big Data Analytics—A Brief Research Synthesis

Peter Nielsen

Abstract This paper presents a brief research synthesis based on the author's experience from a number of big data analytic projects completed with privately owned companies spanning transportation, telecommunication and manufacturing industries. The projects have primarily been completed using records of the companies' transactions (sales, production orders, client calls etc.). The main idea is to give a synthesis of the challenges in completing big data analytic projects to support both strategic but especially operational decisions and present the potential pit falls in these projects. The main conclusions are: that big data analytics requires high quality data if one desires to develop complex explanatory and predictive models, that developing complex models is not a goal in itself, that responses between variables may be lagged in time and difficult to establish and that the challenge of the future is to create automated decision making systems exploiting the models created from big data analytics.

Keywords Big data · Research projects · Analytics

1 Big Data (Analytics)

Big data is currently on top of the agendas of researchers, companies and funding agencies alike and is seen as the main way to achieve competitive advantages in the future [1]. The main reason seems to be that companies in the last couple of decades have invested massively in IT-systems and now want to achieve side benefits from all the data stored in these. Furthermore, there seems to be a drive to go further than just presenting data (business intelligence) by analyzing the data and explaining connections between events. Several definitions of big data exist, but in the context

P. Nielsen (✉)
Department of Mechanical and Manufacturing Engineering,
Aalborg University, Aalborg 9220, Denmark
e-mail: peter@m-tech.aau.dk

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*,
Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_1

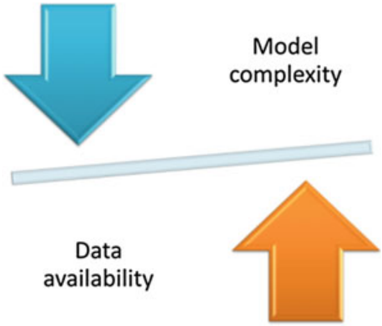

Fig. 1 The opposing forces of big data analytics—data availability versus model complexity

of this paper big data is understood as data comprising different sources and a large number of entries in some (if not all) of these sources.

The important question one must pose is:

How do we gain business advantages using our data?

So the question is not just one of mining data and presenting this, but one of finding explanatory (and exploitable) links in data. In the projects, in which I have been involved, the aim has always been very specifically formulated at the outset. However, the road to the goal has never been simple or linear. Example: we need to have a better model of delivery lead times so we can establish how to plan better. A simple aim that is difficult to address in practice because of data availability and the resulting complexity of the lead time models (see e.g. Nielsen et al. [2, 3] and similar conclusion for demand in Nielsen et al. [4]). The aim of analytics is to establish and explain links between actions such as—we lower prices, we gain how much market share/loose how much profit? In reality one must work with the data that is available and at best establish which data one wants to log/create/establish in the future to create better models. As Fig. 1 illustrates, big data analytics is about balancing two inseparable opposing forces: the desire to have as accurate a model as possible vs. using the available data.

The remainder of the paper explores these topics and with real life examples illustrate how to overcome some of the challenges or live with the consequences.

2 Data and Data Quality

Big data analytics implies many observations need to be evaluated and included in a model. The main challenge is to get from the many individual observations to an aggregated model explaining the object of investigation as a whole and back to the individual observation as illustrated in Fig. 2.

The need to aggregate information creates its own challenges as it is not possible to aggregate data without losing information [5, 6], but it is a necessary price to pay

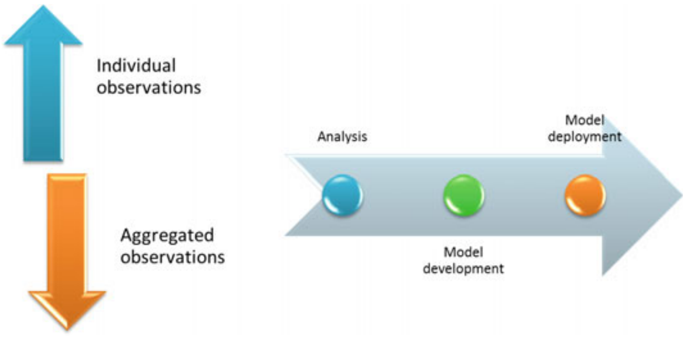

Fig. 2 The need to work on individual observations versus aggregated information during various phases of a big data analytics project

if one wants to be able to explain relations. There is no one way to go about this, but my personal experience from projects has been that it is better to use robust statistics (e.g. medians instead of means) and use time on cleaning the data to remove the worst noise. The challenge with big data analytics from this perspective is of course how to manage the many observations. When faced with this many observations it is not possible to establish why a particular observation is an outlier, so one must automatically remove the observations that are wrong or misleading or approach the problem with models that do not over-fit to the extreme observations. This is tricky and my experience from a number of industries and problems is that there is no best way to go about this. It is hard work and data cleansing and outlier identification and removal takes up the majority of any research project in this field. Even when using data automatically gathered using e.g. RFID one is still faced with the odd data point that cannot be explained.

A further complexity is that data in companies is not static, but changing continuously. This can mean that a company can reach different conclusion using the same type of data depending on when they pull the information from their system. In one project [7] we investigated the amount of changes to the customer orders in a company to establish how much the company was changing the data as they changed delivery times and order quantities to customers. The conclusion was not necessarily uplifting, since the analysis established that company was in actuality creating a more difficult planning environment for themselves.

3 Exogenous Explanatory Variables

Having established the available data, structured it in a reasonable way for analysis and removed the worst of the noise, one is typically faced with the problem that some vital information is missing or that the model does not still explain the behavior in a satisfactory manner. If one wants to explain oil price movements one must look not just at the short term price development but also at macroeconomic data. In one particular project we looked at the freight rates for a liner shipping company [8]. Here we aggregated millions of individual transactions into average weekly pricing information, combined it with market price indexes and developed a short term forecast model for freight rates. The research clearly showed the need to use the external information from the price index was required to explain short term developments in rates [8]. It also clearly demonstrated that the significance of the variables in the model changed depending on how far into the future one wants to predict freight rates (i.e. the relationship between variables is not the same on different time horizons). So not only were the most significant variables not based on the company's own data, their importance changed depending on the company's model requirements. In telecommunications we see the same behavior, calls depend on holidays, break downs, natural disasters etc. all exogenous variables, some of which are impossible to predict or occur with such low frequency that even though you may have literately billions of observations of calls, but only one occurrence of a specific national holiday falling on the first of a particular month that happens to be a Monday. This makes for difficult analysis—and it also underlines a point of modelling: sometimes one cannot make great models because the most important variables act in a completely unpredictable manner.

4 Model Complexity

Achieving the right model complexity is always tricky and it is impossible to separate the model complexity from data quality and availability. The aim must be a model that is sufficiently complex to predict responses to changes in conditions, but not so complex as to be unstable. On top it is critical to understand that a model is only valid as long as the conditions under which it is developed are still present. This means that one must establish achieve a sufficient time of status quo operations in the company, something that is not always possible. The potential usage of models is almost uncountable, but some simple facts tend to hold true:

- A few variables explain most of the behavior of a particular investigated variable. Including more explanatory factors increase the precision but almost always result in over-fitted unstable models (see e.g. Nielsen et al. [8]).
- Models should be flexible—i.e. be able to be adapted to changing circumstances.
- Never discount interactions.
- The world is stochastic—the models should reflect this.

To further increase model complexity my personal experience from a number of research projects is that the response between variables may be lagged in time. Suppose company A lowers prices, its customers require a certain time to adapt (this time horizon may vary significantly from industry to industry) before they change from competitors to company A. How long do they require? And how do we include this in a model?

All in all with the challenges of data quality, short periods of status quo operations this tends to mean that one develops simple but robust models. An important lesson here is that one must be ready to change or completely abandon a model when it no longer fits with the data. Business insight and data monitoring become critical tools for interpreting explanations, but mostly to determine when to abandon a model. Abandoning a model is sometimes the most difficult thing to do, especially when many resources have been invested in its development.

5 Future Perspectives and Conclusions

Returning to the main question posed in the first section of the paper: how do we gain business advantages using our data? We can now try to speculate on where we will end up in the future. The goal of analytics and big data analytics in specific is to explain data in a manner that creates exploitable business understanding. The natural next step is to formalize this in decision support systems and directly assist business managers in making decisions. There are numerous examples of decision support systems [9–11], but the interesting challenge here is what next—where should we go?

Faced with an increasing number of transactions and thus decisions to be made, companies find themselves in the situation that they need to focus on making quick decisions with limited resources available for making the decisions. The natural next step seems to be *automated decision making* systems. Figure 3 illustrates the

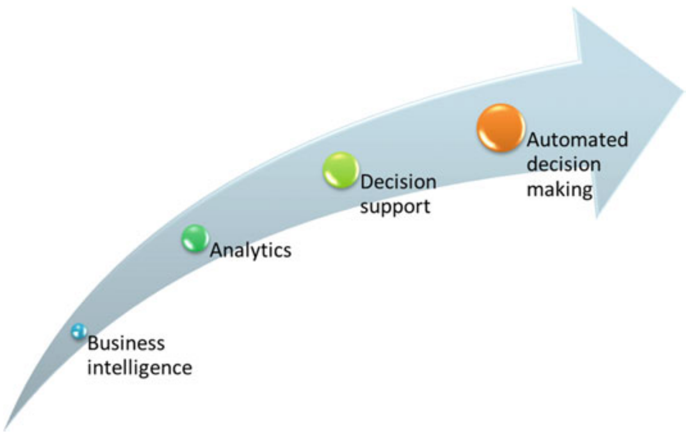

Fig. 3 The development of data driven decision making, from the earliest business intelligence to the future of automated decision making

progress from business intelligence to (Big data) analytics to decision support and finally to automated decision making.

There are a number of challenges that must be solved to enable automated decision making. Primary among these is creating sufficient data of sufficiently good quality that is updated in real time mode. It is easy enough to arrive at wrong conclusions from an analysis of data. However, the analytical approaches typically have the built in safeguard that there is a human evaluation of the model (the axiom "correlation does not imply causation" is worth keeping in mind). In the case of automated decision making there is no such safety option and the systems should be self-reliant and robust. This puts hitherto unseen demands on the quality of the data used in two dimensions:

- The data should be accurate (correct)
- Must be updated quickly (real time)

Furthermore, one can consider if some sort of data monitoring should be implemented such that if the observations are drifting from the expectations the model is flagged as poorly performing and human intervention is again required.

Big data has changed the way companies manage their businesses. But one must be wary of not blindly trusting the models. Ironically trust is perhaps also the main challenge in implementing automated decision making. If one does not trust the data how can one trust the conclusions and even worse put the fate of ones company in the hands of a system basing its decisions on this data? The conclusion must be that the building block for all future work is creating systems for gathering high quality data, thus enabling automated decision making systems to be developed and deployed.

References

1. McAfee, A., Brynjolfsson, E.: Big data: the management revolution. Harvard Bus. Rev. **90** (10), 60–66, 68, 128 (2012)
2. Nielsen, P., Michna, Z., Do, N.A.D.: An empirical investigation of lead time distributions, advances in production management systems. In: Innovative and Knowledge-Based Production Management in a Global-Local World: IFIP WG 5.7 International Conference, APMS 2014, Ajaccio, France, 20–24 Sept 2014, Proceedings, Part I, vol. 438, pp. 435–442. Springer, Heidelberg (2014a)
3. Nielsen, P., Michna, Z., Do, N.A.D., Sørensen, B.B.: Lead times and order sizes—a not so simple relationship. In: 36th International Conference Information Systems Architecture and Technology, Karpacz (2015)
4. Nielsen, P., Nielsen, I., Steger-Jensen, K.: Analyzing and evaluating product demand interdependencies. Comput. Ind. **61**, 869–876 (2010)
5. Orcutt, G.H., Watts, H.W., Edwards, J.B.: Data aggregation and information loss. Am. Econ. Rev. **58**, 773–787 (1968)
6. Theil, H.: Linear Aggregation of Economic Relations, 1st. edn. North-Holland Publishing Company (1955)
7. Nielsen, P., Do, N.A.D., Eriksen, T., Nielsen, I.: Towards an analysis methodology for identifying root causes of poor delivery performance. Found. Manag. **6**(2), 31–42 (2014)

8. Nielsen, P., Liping, J., Rytter, N.G.M., Chen, G.: An investigation of forecast horizon and observation fit's influence on an econometric rate forecast model in the liner shipping industry. Marit. Policy Manag. **41**, 667–682 (2014)
9. Sitek, P., Wikarek J.: A hybrid framework for the modelling and optimisation of decision problems in sustainable supply chain management. Int. J. Prod. Res. 1–18 (2015). doi: 10.1080/00207543.2015.1005762
10. Shim, J.P., Warkentin, M., Courtney, J.F., Power, D.J., Sharda, R., Carlsson, C.: Past, present, and future of decision support technology. Decis. Support Syst. **33**(2), 111–126 (2002)
11. Wikarek, J., Nielsen, I.E.: A multi-agent hybrid approach to decision support in job groups handling. Commun. Comput. Info. Sci. **524**, 80–89 (2015)

Part II
Invited Session: Advances in Methods for Managing Complex Planning Environments

Towards Cyclic Scheduling of Grid-Like Structure Networks

Robert Wójcik, Grzegorz Bocewicz and Zbigniew Banaszak

Abstract The paper treats about a grid-like topology of different means of transport acting in mesh-like streets network in which several modes interact each other via common shared hubs (i.e. stops, interchange stations, cross-platforms, etc.) as to provide mass customized passenger services, tailored to each travel destination. In that context, a grid-like layout of various transport modes such as tram, bus, train, subway where passenger flows are treated as multimodal processes can be seen as a real-life example of the considered case. The goal is to provide a declarative model representation enabling to state a constraint satisfaction problem aimed at multimodal transportation processes scheduling aimed at dedicated flows of passengers service. The main objective is to provide conditions guaranteeing the right match-up of local cyclic acting tram/bus/metro line schedules to a given passenger flow itineraries.

Keywords Grid structure · Cyclic scheduling · Multimodal processes · Declarative modeling · Constraint satisfaction problem

R. Wójcik (✉)
Department of Computer Engineering, Faculty of Electronics,
Wrocław University of Technology, Wroclaw, Poland
e-mail: robert.wojcik@pwr.wroc.pl

G. Bocewicz
Department of Computer Science and Management,
Koszalin University of Technology, Koszalin, Poland
e-mail: bocewicz@ie.tu.koszalin.pl

Z. Banaszak
Department of Business Informatics, Warsaw University of Technology,
Wroclaw, Poland
e-mail: Z.Banaszak@wz.pw.edu.pl

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*,
Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_2

1 Introduction

Numerous road network patterns deployed in cities rang from the tightly structured mesh-like network with perpendicular roads in a regular raster pattern to the hierarchical network with sprawling secondary and tertiary roads feeding into arterial roads in a branch like system [4, 6]. A grid-like topology of different means of transport providing circular service and operating in such mesh-like streets network consists of different bus/tram/subway lines interacting each other via distinguished subsets of common shared hubs as to provide a variety of mass customized passenger services and guarantee no one area of city network gets preferential service treatment over another [8]. Multimodal processes [2, 3] executed in grid-like network of transport modes (GNTM), see Fig. 1 providing

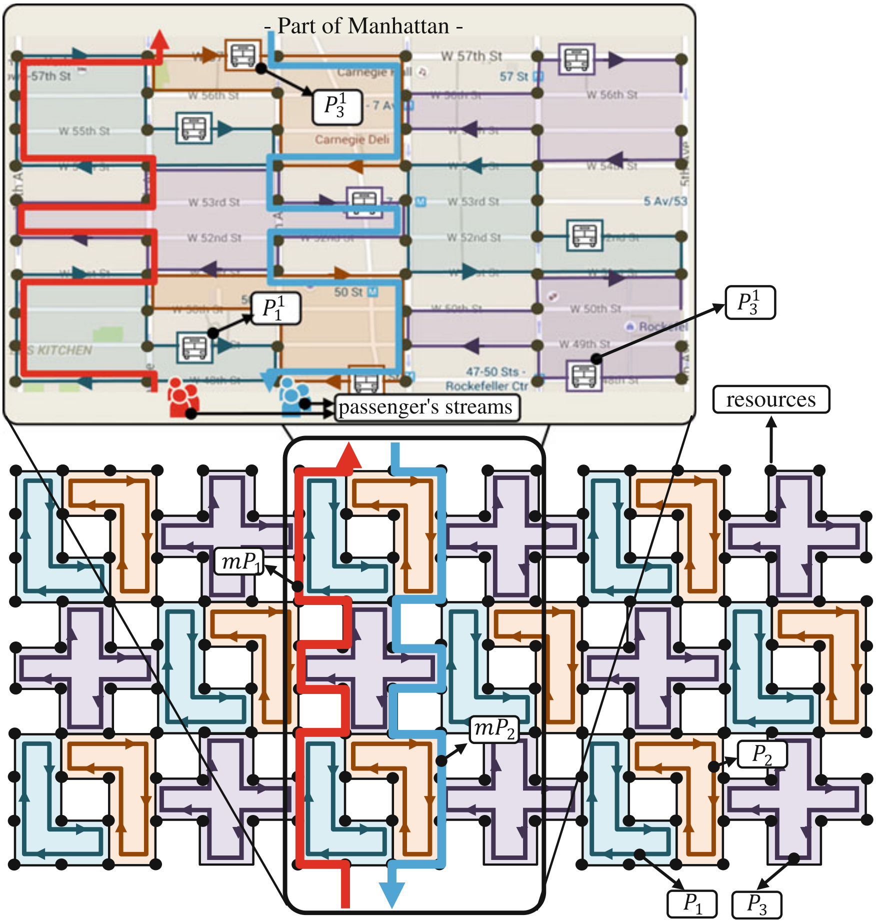

Fig. 1 An example of grid-like structure network

connection from origin to destination, can be seen as passengers and/or goods flows transferred between different modes to reach their destination [9]. The throughput of passengers and/or freight depends on geometrical and operational characteristics of GNTM. In that context the solutions of the layout designs exposing the grid-like structures are frequently observed [4].

The problems arising in these kind of networks concern multimodal routing of freight flows and supporting them multimodal transportation processes (MTP) scheduling. Since the transportation processes executed by particular lines are usually cyclic, hence the multimodal processes supported by them have also periodic character. That means, the periodicity of MTP depends on periodicity of local processes executed in GNTM.

Many models and methods aimed at cyclic scheduling have been considered so far [7]. Among them, the mathematical programming approach [1, 13], max-plus algebra [10], constraint programming [2], Petri nets [12] frameworks belong to the more frequently used. Most of them are oriented at finding of a minimal cycle or maximal throughput while assuming deadlock-free processes flow. However, the approaches trying to estimate the cycle time from cyclic processes structure and the synchronization mechanism employed (i.e. mutual exclusion instances) while taking into account deadlock phenomena are quite unique. In that context our main contribution is to propose a new modeling framework enabling to evaluate the cyclic steady state of a given GNTM encompassing the behavior typical for passenger transportation services (see Fig. 1a) in the mesh-like streets network.

2 Grid-Like Structure Networks

The GNTM shown in Fig. 1 can be seen as a network of bus/trams lines providing periodic service along cyclic routes and can be modeled in terms of a System of Concurrently executed Cyclic Processes (SCCP) as shown in Fig. 2a, b [2, 5]. The structure of considered SCCP consists of patterns indexed by (i) and composed of following shapes "$\llcorner$", "$\ulcorner$", "+" encompassing three kinds of **local cyclic processes**, viz. P_1, P_2,P_3, (distinguished in Fig. 2 by ${}^{(i)}P_1$, ${}^{(i)}P_2$, ${}^{(i)}P_3$, respectively). The processes follow the **routes** composed of transportation sectors and workstations (distinguished in Fig. 2b by the **set of resources** $R = \{R_1, \ldots, R_c, \ldots, R_{18}\}$, R_c—the cth resource). The local cyclic processes P_i contain the **streams** P_i^k (the kth stream of the ith local process P_i is denoted as P_i^k): $P_i = \{P_i^1, \ldots, P_i^k, \ldots P_i^{ls(n)}\}$. In the considered case all processes contain only the unique streams: $P_1 = \{P_1^1\}, P_2 = \{P_2^1\}, P_3 = \{P_3^1\}$. Apart from local processes, we consider two **multimodal processes** (i.e. processes executed along the routes consisting parts of the routes of local processes): mP_1, mP_2.

For example, the transportation route depicted by the red line corresponds to the multimodal process mP_1 supported by vehicles (busses and etc.), which in turn encompass local transportation streams P_3^1 and P_1^1. This means that the

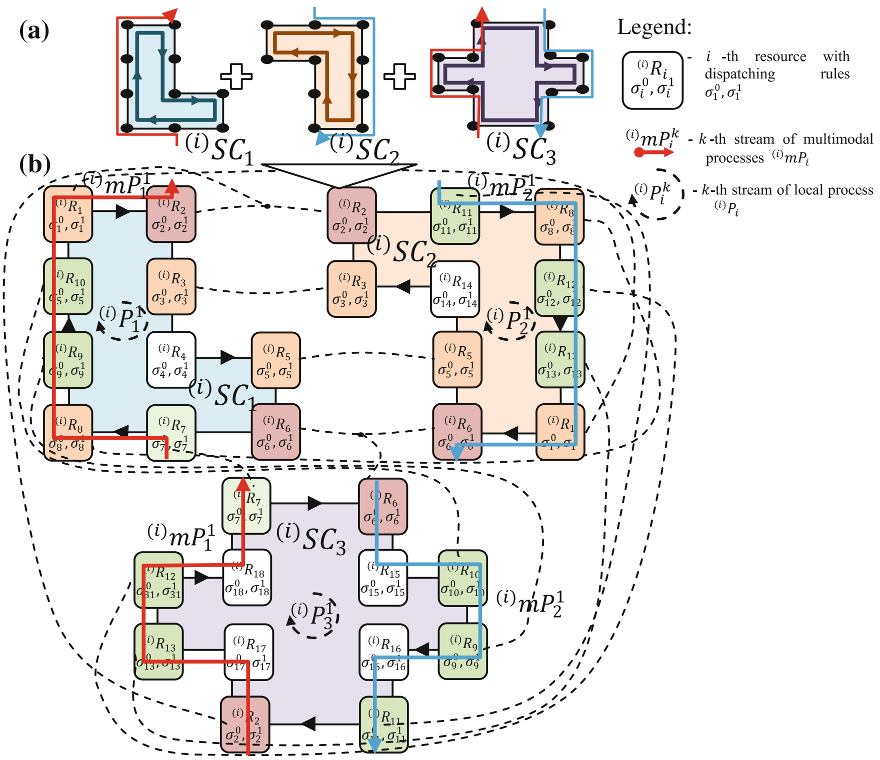

Fig. 2 The pattern composed by substructures: ${}^{(i)}SC_1$, ${}^{(i)}SC_2$,${}^{(i)}SC_3$ distinguished in Fig. 1 (**a**), and represented in terms of SCCP concept (**b**)

transportation routes specifying how a multimodal process is executed can be considered as composed of parts of the routes of local cyclic processes. Similar as in the case of local processes, each multimodal process consists of one stream: $mP_i = \{mP_i^1\}, i = 1, 2$.

Processes can interact with each other through shared resources, i.e. the transportation sectors. The routes p_i^k of the local processes P_i^k are as follows (see Fig. 2): $p_1^1 = (R_1, R_2, \ldots, R_{10})$, $p_2^1 = (R_{11}, R_8, R_{12}, R_{13}, R_1, R_6, R_5, R_{14}, R_3, R_2)$, $p_3^1 = (R_6, R_{15}, R_{10}, R_9, R_{16}, R_{11}, R_2, R_{17}, R_{13}, R_{12}, R_{18}, R_7)$.

Similarly the streams of cyclic multimodal processes: mP_1, mP_2, follow the routes: $mp_1^1 = ((R_7, R_8, R_9, R_{10}, R_1, R_2)) \cap ((R_2, R_{17}, R_{13}, R_{12}, R_{18}, R_7)) = (R_7, R_8, R_9, R_{10}, R_1, R_2, R_{17}, R_{13}, R_{12}, R_{18}, R_7)$, $mp_2^2 = (R_{11}, R_8, R_{12}, R_{13}, R_1, R_6) \cap (R_6, R_{15}, R_{10}, R_9, R_{16}, R_{11}) = (R_{11}, R_8, R_{12}, R_{13}, R_1, R_6, R_{15}, R_{10}, R_9, R_{16}, R_{11})$, where: $(R_7, \ldots, R_2) \cap (R_2, \ldots, R_7)/(R_{11}, \ldots, R_6) \cap (R_6, \ldots, R_{11})$—subsequences of routes $p_1^1, p_3^1/p_2^1, p_3^1$, defining the transportation sections of mp_1^1/mp_2^1, $u \cap v$—concatenation of sequences: $u = (u_1, \ldots, u_a)$, $v = (v_1, \ldots, v_b)$, $u \cap v = (u_1, \ldots, u_a, v_2, \ldots, v_b)$.

A resource conflict (caused by the application of the mutual exclusion protocol) is resolved with the aid of a priority dispatching rule [2, 3], which determines the order in which streams access shared resources. For instance, in the case of the resource R_2 (for substructures ${}^{(i)}SC_1$, ${}^{(i)}SC_2$, ${}^{(i)}SC_3$—see Fig. 2b), the priority dispatching rule: $\sigma_2^0 = (P_1^1, P_2^1, P_3^1)$, determines the order in which streams of local processes can access the shared resource R_2. That means, stream P_1^1 is allowed to access first, then the stream P_2^1 and next stream P_3^1, and then once again P_1^1, and so on. The SCCP shown in Fig. 2 is specified by the set of dispatching rules: $\Theta = \{\Theta^0, \Theta^1\}$, where: $\Theta^0 = \{\sigma_1^0, \ldots, \sigma_c^0, \ldots, \sigma_{18}^0\}, (\Theta^1 = \{\sigma_1^1, \ldots, \sigma_c^1, \ldots, \sigma_{18}^1\})$, $(\Theta^1 = \{\sigma_1^1, \ldots, \sigma_c^1, \ldots, \sigma_{18}^1\})$—set of rules determining the orders of local (multimodal) processes.

In general, the following notation is used:

- a sequence $p_i^k = (p_{i,1}^k, \ldots, p_{i,j}^k, \ldots, p_{i,lr(i)}^k)$ specifies **the route of the stream of the local process P_i^k** (the kth stream of the ith local process P_i). Its components $p_{i,j}^k \in R$ define the resources used in the execution of operations, In the rest of the paper, **the jth operation executed on the resource $p_{i,j}^k$ in the stream P_i^k** will be denoted by $o_{i,j}^k$; $lr(i)$ is the length of the cyclic process route.
- $x_{i,j}^k(l) \in \mathbb{N}$—the moment of operation beginning $o_{i,j}^k$ in the lth cycle,
- $t_i^k = (t_{i,1}^k, t_{i,2}^k, \ldots, t_{i,j}^k, \ldots, t_{i,lr(i)}^k)$ specifies **the operation times of local processes**, where $t_{i,j}^k$ denotes the time of execution of operation $o_{i,j}^k$.
- $mp_i^k = \left(mpr_{i_1}^{q_1}(a_{i_1}, b_{i_1}) \cap \ldots \cap mpr_{i_y}^{q_y}(a_{i_y}, b_{i_y})\right)$ specifies **the route of the stream mP_i^k from the multimodal process mP_i** (the kth stream of the ith multimodal process mP_i), where: $mpr_i^q(a, b)$ is the subsequence of the route p_i^q containing elements from $p_{i,a}^q$ to $p_{i,b}^q$. In the rest of the paper, **the jth operation executed in the stream mP_i^k** will be denoted by $mo_{i,j}^k$,
- $mx_{i,j}^k(l) \in \mathbb{N}$—the moment of operation beginning $mo_{i,j}^k$ in the lth cycle.
- $mt_i^k = \left(mt_{i,1}^k, mt_{i,2}^k, \ldots, m_{i,j}^k, \ldots, mt_{i,ldm(i)}^k\right)$ specifies **the operation times of multimodal processes**, where $mt_{i,j}^k$ denotes the time of execution of operation $mo_{i,j}^k$,
- $\Theta = \{\Theta^0, \Theta^1\}$ is the set of **priority dispatching rules**, $\Theta^i = \{\sigma_1^i, \ldots, \sigma_c^i, \ldots, \sigma_m^i\}$ is the set of priority dispatching rules for local ($i = 0$)/multimodal ($i = 1$) processes where: σ_c^i are sequence which determine the order in which the processes can be executed on the resource R_c.

Using the above notation, a SCCP can be defined as a tuple:

$$SC = ((R, SL), SM), \tag{1}$$

where $R = \{R_1, \ldots, R_c, \ldots, R_m\}$—the set of resources, $SL = (U, T, \Theta^0)$—the structure of local processes of SCCP, i.e.: U—the set of routes of local process, T—the set of sequences of operation times in local processes, $\Theta^0 = \{\sigma_1^0, \ldots, \sigma_c^0, \ldots, \sigma_m^0\}$—the set of priority dispatching rules for local processes. $SM = (M, mT, \Theta^1)$—the structure of multimodal processes of SCCP, i.e.: M—the set of routes of a multimodal process, mT—the set of sequences of operation times in multimodal processes, Θ^1—the set of priority dispatching rules for multimodal processes.

The behavior of the structure of SCCP (1) will be characterized by the schedule (2):

$$X' = ((X, \alpha), (mX, m\alpha)), \tag{2}$$

where $X = \left\{x_{1,1}^1, \ldots, x_{i,j}^k, \ldots, x_{n,lr(n)}^{ls(n)}\right\}$—a set of the moments of local processes operations beginning in $l = 0$ of the cycle, $x_{i,j}^k$—determines the value: $x_{i,j}^k(l) = x_{i,j}^k + \alpha$, α—periodicity of local processes executions, $mX = \left\{mx_{1,1}^1, \ldots, mx_{i,j}^k, \ldots, mx_{w,lm(w)}^{lsm(w)}\right\}$—a set of moments of operations of multimodal processes beginning in $l = 0$ of cycle, $mx_{i,j}^k$—determines the value $mx_{i,j}^k(l) = mx_{i,j}^k + m\alpha \cdot l$, $m\alpha$—periodicity of multimodal processes executions.

Consider SCCP following the grid-like structure shown in Fig. 1 and created as a result of multiple composition of the structure shown in Fig. 2. Formally, the grid-like structure is defined as SC (1) structure, that can be decomposed into the set of isomorphic substructures: $SC^* = \{SC_1, \ldots, SC_i, \ldots, SC_{lc}\}$ and following:

(a) each substructure $SC_i \in SC^*$ of the structure SC is defined analogically as (1):

$$SC_i = ((Rp_i, SLp_i), SMp_i). \tag{3}$$

where $Rp_i \subset R$—the set of resources of sub-structure SC_i, SLp_i—level of local processes of substructure SC_i, including: local processes $Pp_i \subset P$ with corresponding route sequences $Up_i \subset U$; the operation times $Tp_i \subset T$; the set of dispatching rules Θ_i^0. SMp_i—level of multimodal processes of substructure SC_i, including fragments of multimodal processes $mP_j(a, b)$ (fragment of the process mP_j related with executing the operation from a, $a + 1$, …, b) forming the set mPp_i.

(b) $SC^* = \{SC_1, \ldots, SC_i, \ldots, SC_{lc}\}$ is a set of substructures of the structure SC if [3]: $\bigcup_{i=1}^{lc} Rp_i = R$—substructures include all resources of the structure SC, $\bigcup_{i=1}^{lc} Pp_i = P$; $\prod_{i=1}^{lc} Pp_i = \varnothing$ and $\bigcup_{i=1}^{lc} Up_i = U$; $\prod_{i=1}^{lc} Up_i = \varnothing$—substructures use all the local processes and one process occurs in exactly one structure, $\prod_{i=1}^{lc} mPp_i = \varnothing$—elements of mPp_i occurs in exactly one substructure.

(c) Two sub-structures SC_a, $SC_b \in SC^*$ are called isomorphic if:

- each resource $R_a \in Rp_a$ of substructure SC_a is corresponding to exactly one resource $R_b \in Rp_b$ of the structure SC_b: $R_b = f(R_a)$,
- each process P_a/mP_a (local as well as multimodal) of the substructure SC_a is corresponding to exactly one process P_b/mP_b of the structure SC_b: $P_b = f(P_a)$,
- routes p_b/mp_b and p_a/mp_a of the corresponding processes are sequences consisting of corresponding resources,
- each operation $o^h_{a,j}\Big/mo^h_{a,j}$ executed within the substructure SC_a is corresponding to exactly one operation $o^h_{b,j}\Big/mo^h_{b,j}$ executed within the substructure $SC_b : o^h_{a,j} = f\left(o^h_{b,j}\right)/mo^h_{a,j} = f\left(mo^h_{b,j}\right)$;
- dispatching rules σ^l_a/σ^l_b of the corresponding resources are sequences consisting of elements $s^l_{a,d}/s^l_{b,d}$ indicating the streams of corresponding processes.

The structure shown in Fig. 1 consists of three types of isomorphic substructures presented in Fig. 2 and denoted as ${}^{(i)}SC_1$, ${}^{(i)}SC_2$, ${}^{(i)}SC_3$. Each of them includes one local processes ${}^{(i)}\mathrm{P}_1$, ${}^{(i)}\mathrm{P}_2$, ${}^{(i)}\mathrm{P}_3$ respectively and one (${}^{(i)}SC_1$, ${}^{(i)}SC_2$) or two (${}^{(i)}SC_3$.) fragments of multimodal processes.

3 Problem Formulation

Given a grid-like structure SC (1), where the dispatching rules Θ are unknown. An answer is sought to the question whether there are such values Θ that can guarantee that the cyclic behavior represented by the schedule X' (2) will be attainable in the structure SC (1).

The grid-like structure SC (1) can be decomposed into a set of isomorphic substructures $SC^* = \{SC_1, \ldots, SC_i, \ldots, SC_{lc}\}$. Therefore, the selection of dispatching rules Θ can be carried out independently for each substructure. If, for every substructure SC_i, there is a subset of parameters Θ that guarantee its cyclic behavior, then the considered problem should provide an answer to the following question: **Does there exist a way of substructures *SC**, that can guarantee the cyclic work of the system *SC***?

In order to answer this question the **operator of substructures composition** $\oplus$ is introduced as well as constraint assuming the result of composition of two substructures SC_a, SCb through mutually shared resources ($Rp_a \cap Rp_b \neq \emptyset$) results in $SC_a \oplus SC_b = SC_c$ where:

$$SC_c = ((Rp_c, SLp_c), SMp_c), \tag{4}$$

and $Rp_c = Rp_a \cup Rp_b$—the set of resources, and variables characterizing SLp_c are determined in the following way: $Pp_c = Pp_a \cup Pp_b$; $Up_c = Up_a \cup Up_b$; $Tp_c = Tp_a \cup Tp_b$, $\Theta_c^0 = \{\sigma_{k,c}^0 | k = 1 \ldots lk\}$, where:

$$\sigma_{k,c}^l = \begin{cases} \sigma_{k,a}^l & \text{for } R_k \in Rp_a \text{ and } R_k \notin Rp_b \\ \sigma_{k,b}^l & \text{for } R_k \in Rp_b \text{ and } R_k \notin Rp_a \\ \vartheta(\sigma_{k,a}^l, \sigma_{k,b}^l) & \text{for } R_k \in Rp_a \text{ and } R_k \in Rp_b \end{cases}, \tag{5}$$

$\vartheta\left(\sigma_{k,a}^0, \sigma_{k,b}^0\right)$—function determining the dispatching rules for the mutual resource R_k of the composed structures. SMp_c consist of mPp_c composed of parts of multimodal processes following the sets mPp_a and mPp_b [3].

4 Cyclic Scheduling

4.1 Determining Cyclic Steady Processes

Figure 2 shows in detail the substructures arrangement of the system from Fig. 1. There is three type of elementary isomorphic substructures ${}^{(i)}SC_1$, ${}^{(i)}SC_2$, ${}^{(i)}SC_3$ which are put together by means of integrating mutual resources. As Fig. 2 shows, for every substructure ${}^{(i)}SC_1$, ${}^{(i)}SC_2$,${}^{(i)}SC_3$ processes are implemented in the same manner: operations are performed along the same routes, the same dispatching rules are applied, etc. In this context the introduced operator of substructures composition ($\oplus$) SC can be shown as a multiple composition:

$$SC = \oplus_{i=1}^{lc}\left({}^{(i)}SC_1 \oplus^{(i)} SC_2 \oplus^{(i)} SC_3\right), \tag{6}$$

where $\oplus_{i=1}^{lc}\left({}^{(i)}SC\right) =^{(1)} SC \oplus \cdots \oplus^{(i)} SC \oplus \cdots \oplus^{(lc)} SC$—means composition according to Eqs. (4) and (5) i.e. each substructure ${}^{(i)}SC$ is put together with the others by means of integrating the resources belonging to the same set of corresponding resources. For example, the structure ${}^{(i)}SC_1$ from Fig. 2 is put together with ${}^{(i)}SC_2$ by the resources ${}^{(i)}R_2$, ${}^{(i)}R_3$, ${}^{(i)}R_5$, ${}^{(i)}R_6$ and with ${}^{(i)}SC_3$ by the resources ${}^{(i)}R_7$, ${}^{(i)}R_6$, ${}^{(i)}R_9$, ${}^{(i)}R_{10}$. Due to the same manner of process execution, as well as the same manner of substructures composition, the cyclic schedule representing the behavior of the whole structure can be perceived as a composition of corresponding (isomorphic) schedules:

$$X' = \Cup_{i=1}^{lc}\left({}^{(i)}X'_1 \Cup {}^{(i)}X'_2 \Cup {}^{(i)}X'_3 \right), \tag{7}$$

where ${}^{(i)}X'_Z$—the cyclic schedule of the substructure ${}^{(i)}SC_Z$:

$$ {}^{(i)}X'_Z = \left(\left({}^{(i)}X_Z, {}^{(i)}\alpha_Z \right), \left({}^{(i)}mX_Z, {}^{(i)}m\alpha_Z \right) \right), \tag{8}$$

${}^{(i)}X_Z$/${}^{(i)}mX_Z$—set of the initiation moments of local/multimodal process operations of the substructure ${}^{(i)}SC_Z$; ${}^{(i)}\alpha_Z$/${}^{(i)}m\alpha_Z$—periodicity of local/multimodal processes executions; $\Cup_{i=1}^{lc}\left({}^{(i)}X' \right) = {}^{(1)}X' \Cup \ldots \Cup {}^{(i)}X' \Cup \ldots \Cup {}^{(lc)}X'$—composition of schedules ${}^{(i)}X'$, ${}^{(a)}X' \Cup {}^{(b)}X'$—the schedule composition ${}^{(a)}X'$, ${}^{(b)}X'$:

$$ {}^{(a)}X' \Cup {}^{(b)}X' = \left(\left({}^{(a)}X \cup {}^{(b)}X, lcm\left({}^{(a)}\alpha, {}^{(b)}\alpha \right) \right), \left({}^{(a)}mX \cup {}^{(b)}mX, lcm\left({}^{(a)}m\alpha, {}^{(b)}m\alpha \right) \right) \tag{9}$$

In order to determine the schedule X' it is enough to know the schedule ${}^{(i)}X'_1 \Cup {}^{(i)}X'_2 \Cup {}^{(i)}X'_3$ of composition ${}^{(i)}SC_1 \oplus {}^{(i)}SC_2 \oplus {}^{(i)}SC_3$. However, to make the composition (7) possible, it is necessary to make sure that the operations executed according to ${}^{(i)}X'_Z$, do not lead to deadlocks.

In order to determine such parameters as dispatching rules ${}^{(i)}\Theta_Z$ of the substructures ${}^{(i)}SC_1$, ${}^{(i)}SC_2$, ${}^{(i)}SC_3$ (Fig. 2) that guarantee the attainability of the cyclic schedule ${}^{(i)}X'_Z$ within the structure, it is possible to apply the constraint satisfaction problem (10) [11]:

$$PS'_{REXi} = \left(\left(\left\{ {}^{(i)}X'_Z, {}^{(i)}\Theta_Z, {}^{(i)}\alpha'_z \right\}, \{D_X, D_\Theta, D_\alpha\} \right), \{C_L, C_M, C_D\} \right) \tag{10}$$

where ${}^{(i)}X'_Z$, ${}^{(i)}\Theta_Z$, ${}^{(i)}\alpha'_z$—decision variables, ${}^{(i)}X'_Z$—cyclic schedule (8) of substructure ${}^{(i)}SC_Z$, ${}^{(i)}\Theta_Z = \left\{ {}^{(i)}\Theta_z^0, {}^{(i)}\Theta_z^1 \right\}$—the set of priority dispatching rules for ${}^{(i)}SC_Z$, ${}^{(i)}\alpha'_z = \left({}^{(i)}\alpha_z, {}^{(i)}m\alpha_z \right)$—periodicity of local/multimodal processes for ${}^{(i)}SC_Z$, D_X, D_Θ, D_α—domains determining admissible value of decision variables.

$\{C_L, C_M, C_D\}$—the set of constraints C_L and C_M describing SCCP behavior; C_L—constraints determining cyclic steady state of local processes, i.e. their cyclic schedule [2], C_M—constraints determining multimodal processes behavior [2], C_D—constraints that guarantee the smooth implementation of the stream on mutual resources.

The schedule ${}^{(i)}X'_Z$ that meets all the constraints from the given set $\{C_L, C_M, C_D\}$ is the solution sought for the problem (10). It means that it is possible to smoothly execute the operations occurring in ${}^{(i)}SC_Z$ as well as in neighboring substructures.

4.2 Principle of Match-up Structures Coupling

The constraints C_L, C_M guarantee that in the substructure ${}^{(i)}SC_Z$ from Fig. 2 the processes will be executed in a cyclic and deadlock-free manner [2]. These constraints, however, cannot ensure the lack of interferences between the operations of neighboring substructure streams with the substructure ${}^{(i)}SC_Z$. In order to avoid interferences of this kind, additional constraints C_D, are introduced, which describe the relationships between the process operations of the constituted structures. For that purpose the principle of match-up structures coupling is applied.

The idea of the principle of match-up structures coupling is to attain the cyclic schedule X'_c (that does not lead to any collisions between operations) in the substructure SC_c, gained as a result of the composition $SC_a \oplus SC_b$. The cyclic schedule is a composition of the schedules X'_a, X'_b: $X'_c = X'_a \uplus X'_b$ (7) if:

- the value of the periodicity of schedule X'_a is the total multiple of the periodicity of schedule X'_b: $m\alpha_a MOD\ m\alpha_b = 0$; and $\alpha_a MOD \alpha_b = 0$
- the operations of mutual resources $Rk = Rp_a \cap Rp_b = \{R_{k_1}, \ldots, R_{k_i}, \ldots, R_{k_q}\}$ are executed without mutual interferences.

Formally, the constraints that guarantee the lack of interferences while executing the process operations on mutual resources are defined in the following way:

Constraints for local process operations In order to guarantee the smooth process implementation on the resource $R_{k_i} \in Rk$ the extension of the conventional constraints of non-superimposition of time intervals is used. The two operations $o^h_{i,j}, o^s_{q,r}$ do not interfere (on the mutually shared resource R_{k_i}) if the operation $o^h_{i,j}$ begins (moment $x^h_{i,j}$) after the release (with the delay Δt) of the resource by the operation $o^s_{q,r}$ (moment x^s_{q,r^*} of the subsequent operation initiation) and releases the resource (moment x^h_{i,j^*} of the subsequent operation initiation) before the beginning of the next execution of the operation $o^s_{q,r}$ (moment $x^s_{q,r} + \alpha_b$). The collision-free execution of the local process operations is possible if the following constraint holds:

$$\begin{aligned}&\left[\left(x^h_{i,j} \geq x^s_{q,r^*} + k'' \cdot \alpha_b + \Delta t\right) \wedge \left(x^h_{i,j^*} + k' \cdot \alpha_a + \Delta t \leq x^s_{q,r} + \alpha_b\right)\right] \\ &\qquad \vee \left[\left(x^s_{q,r} \geq x^h_{i,j^*} + k' \cdot \alpha_a + \Delta t\right) \wedge \left(x^s_{q,r^*} + k'' \cdot \alpha_b + \Delta t \leq x^h_{i,j} + \alpha_a\right)\right]\end{aligned} \tag{11}$$

where $j^* = (j+1)MODlr(i), \quad r^* = (r+1)MODlr(q)$,

$$k' = \begin{cases} 0 \text{ when } & j+1 \leq lr(i) \\ 1 \text{ when } & j+1 < lr(i) \end{cases}, \quad k'' = \begin{cases} 0 \text{ when } & r+1 \leq lr(q) \\ 1 \text{ when } & r+1 < lr(q) \end{cases}, \tag{12}$$

Constraints for multimodal processes In order to guarantee an interference-free implementation of the multimodal processes (when the condition of mutual exclusion is applied) the applied conditions are similar to those used for local processes. Two operations $mo^h_{i,j}, mo^s_{q,r}$ can be executed without any interferences on the mutually shared resource $R_{k_i} \in Rk$ if one operation is executed between the subsequent executions of the other. In this context, the collision-free execution of the multimodal process operations is possible if the following constraint is satisfied:

$$\left[\left(mx^h_{i,j} \geq mx^s_{q,r^*} + k'' \cdot m\alpha_b + \Delta t\right) \wedge \left(mx^h_{i,j^*} + k' \cdot m\alpha_a + \Delta t \leq mx^s_{q,r} + m\alpha_b\right)\right] \\ \vee \left[\left(mx^s_{q,r} \geq mx^h_{i,j^*} + k' \cdot m\alpha_a + \Delta t\right) \wedge \left(mx^s_{q,r^*} + k'' \cdot m\alpha_b + \Delta t \leq mx^h_{i,j} + m\alpha_a\right)\right] \tag{13}$$

where j^*, r^*, k' and k'' defined as in Eq. (11), $mx^h_{i,j^*}, mx^s_{q,r}$—initiation moments of the operations $mo^h_{i,j}, mo^s_{q,r}$ of substructures SC_a, SC_b, respectively;mx^h_{i,j^*}, mx^s_{q,r^*}—initiation moments of operations executed after $mo^h_{i,j}, mo^s_{q,r}$, respectively.

The constraints (11) and (13) must be satisfied so that the composition of two substructures $SC_c = SC_a \oplus SC_b$ of the known cyclic behaviors, is also characterized by the cyclic behavior X'_c. If these constraints are satisfied, the manner of executing operations on mutual resources Rk determines the form of dispatching rules $\sigma^0_{k,c}$ (5), and, to be more exact, the form of functions $\vartheta\left(\sigma^0_{k,a}, \sigma^0_{k,b}\right)$ and $\vartheta\left(\sigma^1_{k,a}, \sigma^1_{k,b}\right)$. The $\vartheta\left(\sigma^l_{k,a}, \sigma^l_{k,b}\right)$ is determined based on moments of operations executed on the R_k:

$$\vartheta\left(\sigma^l_{k,a}, \sigma^l_{k,b}\right) = \left(s^l_{k,1,c}, \ldots, s^l_{k,j,c}, \ldots, s^l_{k,lh_c,c}\right) \text{ when } x^l_{k,1,c} < \cdots < x^l_{k,lh_c,c}, \quad l \in \{0,1\} \tag{14}$$

where $s^l_{k,j,c}$—jth element of the rule $\sigma^l_{k,c}$ determining the stream of the process of the lth behavior level initiating its operation on the resource R_k in the moment: $x^l_{k,j,c}$; $s^l_{k,j,c}$ is one of the elements of the rules $\sigma^l_{k,a}, \sigma^l_{k,b}$; $x^0_{k,j,c} \in X_a \cup X_b$.

In other words, there are such dispatching rules on mutual R_k as the sequence of operations resulting from the schedules X'_a, X'_b satisfying the constraints (11) and (13).

5 Computational Experiments

The evaluation of the cyclic behaviour (the existence of the schedule X') of the grid-like structure SC from Fig. 1 can be obtained as a result of evaluating the parameters of isomorphic substructures ${}^{(i)}SC_1$, ${}^{(i)}SC_2$,${}^{(i)}SC_3$ from Fig. 2 (it is assumed that the all the operation times are the same and equal to $t^k_{i,j}$ = 5). The problem PS'_{REXi} (10) was formulated in which the constraints C_L, C_M determining the relationships between the behaviour and the structure are formulated according to [2]. In order to formulate the constraints C_D the principle of match-up structures coupling was applied. It is necessary that they guarantee a collision-free execution of stream operations ${}^{(i)}P_1^1, {}^{(i)}P_2^1$, (on the resources ${}^{(i)}R_1$, ${}^{(i)}R_3, {}^{(i)}R_5, {}^{(i)}R_8$), ${}^{(i)}P_1^1, {}^{(i)}P_3^1$, (on the ${}^{(i)}R_7, {}^{(i)}R_9$, ${}^{(i)}R_{10}$), ${}^{(i)}P_2^1, {}^{(i)}P_3^1$, (on the ${}^{(i)}R_{11}, {}^{(i)}R_{12}$, ${}^{(i)}R_{13}$) and ${}^{(i)}P_1^1, {}^{(i)}P_2^{1(i)}P_3^1$, (on the ${}^{(i)}R_2, {}^{(i)}R_6$). The relations regarding to these constraints C_D were shown in Fig. 2 by dot dashed lines. The problem PS'_{REXi}, formulated in this manner, was implemented and solved in the constraint programming environment OzMozart (CPU Intel Core 2 Duo 3 GHz RAM 4 GB). The first acceptable solution was obtained in less than one second. The result of the problem solution are the cyclic schedule ${}^{(i)}X'$ and the dispatching rules ${}^{(i)}\Theta$ shown in Fig. 3 and Table 1. It shows that the operations executed on the mutual resources do not superimpose on each

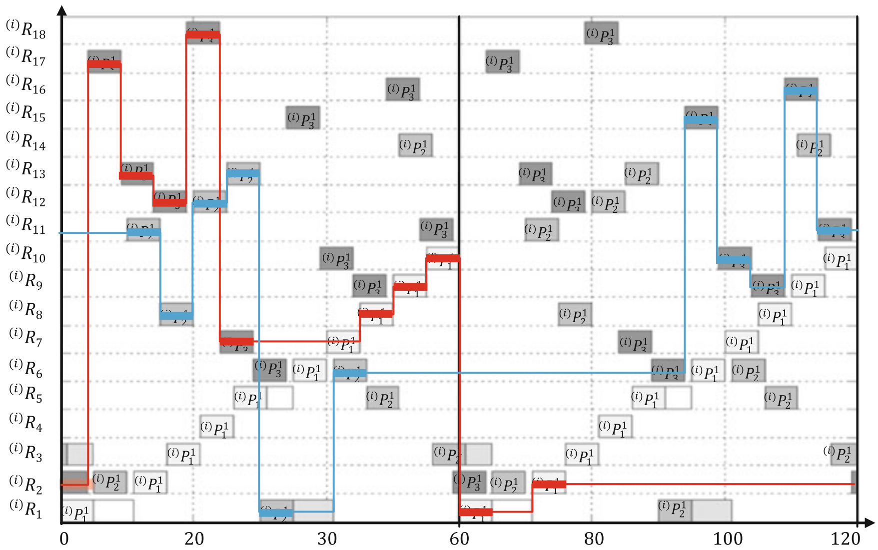

Fig. 3 The schedule of SCCP following ${}^{(i)}SC_1 \oplus {}^{(i)} SC_2 \oplus {}^{(i)} SC_3$ from Fig. 2b

Table 1 The dispatching rules of SCCP following ${}^{(i)}SC_1 \oplus {}^{(i)} SC_2 \oplus {}^{(i)} SC_3$ from Fig. 2b

Dispatching rule for local processes				Dispatching rule for multimodal processes		
${}^{(i)}\sigma_1^0$	$({}^{(i)}P_1^1, {}^{(i)} P_2^1)$	${}^{(i)}\sigma_6^0$	$({}^{(i)}P_1^1, {}^{(i)} P_2^1, {}^{(i)} P_3^1)$	${}^{(i)}\sigma_1^1$	${}^{(i)}\sigma_{12}^1$	$({}^{(i)}m^1P_1^1, {}^{(i)} m^1P_2^1)$
${}^{(i)}\sigma_2^0$	$({}^{(i)}P_1^1, {}^{(i)} P_3^1, {}^{(i)} P_2^1)$	${}^{(i)}\sigma_7^0$	$({}^{(i)}P_1^1, {}^{(i)} P_3^1)$	${}^{(i)}\sigma_8^1$	${}^{(i)}\sigma_{13}^1$	$({}^{(i)}m^1P_1^1, {}^{(i)} m^1P_2^1)$
${}^{(i)}\sigma_3^0$	$({}^{(i)}P_1^1, {}^{(i)} P_2^1)$	${}^{(i)}\sigma_8^0$	$({}^{(i)}P_2^1, {}^{(i)} P_1^1)$	${}^{(i)}\sigma_9^0$		$({}^{(i)}m^1P_1^1, {}^{(i)} m^1P_2^1)$
${}^{(i)}\sigma_5^0$	$({}^{(i)}P_1^1, {}^{(i)} P_2^1)$	${}^{(i)}\sigma_7^0$	$({}^{(i)}P_3^1, {}^{(i)} P_1^1)$	${}^{(i)}\sigma_{10}^0$		$({}^{(i)}m^1P_1^1, {}^{(i)} m^1P_2^1)$

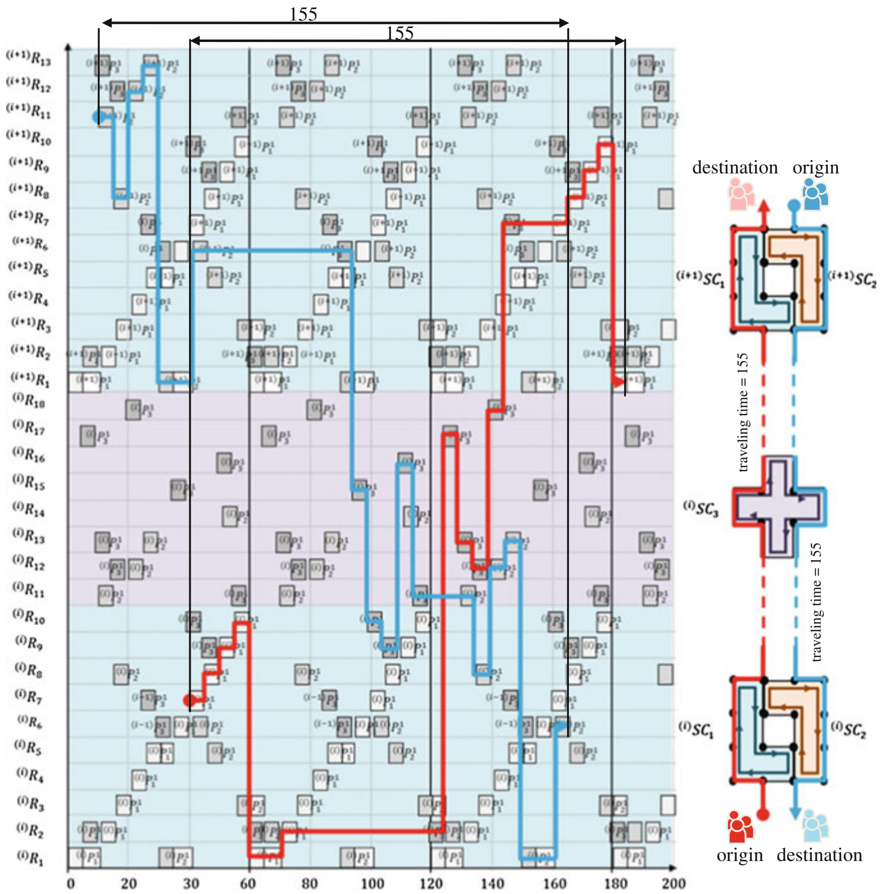

Fig. 4 The Gantt's chart of grid-like network from Fig. 1 including both local and multimodal processes

other. According to Eq. (7) the attained schedule is a component of the schedule X' that characterizes the behavior of the whole structure SC.

The schedule X' (7) being a multiple composition of the schedules ${}^{(i)}X'_1$, ${}^{(i)}X'_2$, ${}^{(i)}X'_3$ is presented in Fig. 4. It is evident that the composition of these schedules of all the substructures of the structure SC does not lead to interferences in the execution of the operation. On the basis of the obtained schedules it is also possible to determine (according to Eq. (14)) the dispatching rules for all the resources of the structure SC; the rules are presented in Table 1.

Referring back to the layout presented in Fig. 1a, the obtained schedule should be treated as an illustration of vehicle movement (local processes) and the method of executing transportation routes (multimodal processes) in a network consisting of numerous fragments of the same type. It should be emphasized that the periodicity of local processes in the network of this kind amounts to $\alpha = 60$ u.t. (units time), and the journey times the passengers have to spent while traveling along the red or blue streams (processes ${}^{(i)}mP_1^1$, ${}^{(i)}mP_2^1$) are equal to 155 u.t.

6 Conclusions

A declarative modeling approach to MTP scheduling in GNTM networks environment is considered. Opposite to traditional approach a given network of local cyclic acting different modes of transportation services is assumed. In considered regular network, composed of elementary, structurally isomorphic subnetworks, the passengers pass their origin-destination itineraries along routes composed of local transportation means. The solution sought assumes that schedules of locally acting subnetworks composed of a set of assumed transportation lines will match-up the schedules of assumed set of passengers itinerary. The relevant sufficient conditions guaranteeing such a match-up exists were provided.

References

1. Abara, J.: Applying integer linear programming to the fleet assignment problem. Interfaces **19**, 4–20 (1989)
2. Bocewicz, G., Nielsen, I., Banaszak, Z.: Automated guided vehicles fleet match-up sched-uling with production flow constraints. Eng. Appl. Artif. Intell. **30**, 49–62 (2014)
3. Bocewicz, G., Banaszak, Z., Pawlewski, P.: Multimodal cyclic processes scheduling in fractal structure networks environment. In: Proceedings of the 19th World Congress: The International Federation of Automatic Control, Cape Town, 8939–8946 (2014)
4. Courtat, T.: Walk on city maps—mathematical and physical phenomenology of the city, a geometrical approach. Modeling and simulation. Université Paris-Diderot—Paris VII (2012). https://tel.archives-ouvertes.fr/tel-00714310
5. Krenczyk, D., Kalinowski, K., Grabowik, C.: Integration production planning and scheduling systems for determination of transitional phases in repetitive production. Hybrid Artif. Intell. Syst. LNCS **7209**, 274–283 (2012)
6. Kelly, G., McCabe, H.: A survey of procedural techniques for city generation. ITB J. **14**, 87–130 (2006)
7. Levner, E., Kats, V., Alcaide, D., Pablo, L., Cheng, T.C.E.: Complexity of cyclic scheduling problems: A state-of-the-art survey. Comput. Ind. Eng. **59**(2), 352–361 (2010)
8. Levinson, D., Huang, A.: A positive theory of network connectivity. Environ. Plan. **39**, 308–325 (2012)
9. Moudani, W., Mora-Camino, F.: A dynamic approach for aircraft assignment and maintenance scheduling by airlines. J. Air Transp. Manag. **6**, 233–237 (2000)
10. Polak, M., Majdzik, P., Banaszak, Z., Wójcik, R.: The performance evaluation tool for automated prototyping of concurrent cyclic processes. Fundamenta Informaticae **60**(1–4), 269–289 (2004)

11. Sitek, P., Wikarek, J.: A hybrid framework for the modelling and optimisation of decision problems in sustainable supply chain management. Int. J. Prod. Res., 1–18 (2015). doi:10.1080/00207543.2015.1005762
12. Song, J.-S., Lee, T.-E.: Petri net modeling and scheduling for cyclic job shops with blocking. Comput. Ind. Eng. **34**(2), 281–295 (1998)
13. Von Kampmeyer, T.: Cyclic scheduling problems. Ph.D. Dissertation, Mathematik/Informatik, Universität Osnabrück (2006)

Concept of Indoor 3D-Route UAV Scheduling System

Yohanes Khosiawan, Izabela Nielsen, Ngoc Anh Dung Do and Bernardo Nugroho Yahya

Abstract The objective of the proposed concept is to develop a methodology to support Unmanned Aerial Vehicles (UAVs) operation with a path planning and scheduling system in 3D environments. The proposed 3D path-planning and scheduling allows the system to schedule UAVs routing to perform tasks in 3D indoor environment. On top of that, the multi-source productive best-first-search concept also supports efficient real-time scheduling in response to uncertain events. Without human intervention, the proposed work provides an automatic scheduling system for UAV routing problem in 3D indoor environment.

Keywords Real-time scheduling · Three-dimensional indoor scheduling

1 Introduction

Human-labor-free automated vehicles are being popularly utilized to perform various manufacturing tasks [6, 7] such as material moving, part-feeding, and storing mechanism. These automated vehicles include the earlier, widely used, Unmanned Ground Vehicles (UGVs)—operating on the ground—and the later newcomer Unmanned Aerial Vehicles (UAVs)—operating in the air.

Y. Khosiawan (✉) · I. Nielsen · N.A.D. Do
Mechanical and Manufacturing Engineering, Aalborg University, Aalborg, Denmark
e-mail: yok@m-tech.aau.dk

I. Nielsen
e-mail: izabela@m-tech.aau.dk

N.A.D. Do
e-mail: ngoc@m-tech.aau.dk

B.N. Yahya
Industrial and Management Engineering, Hankuk University of Foreign Studies, Yongin, South Korea
e-mail: bernardo@hufs.ac.kr

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_3

UAV has been the subject of immense interest in the recent years and has been developed into a mature technology to be applied in areas such as military, search and rescue, agriculture and environmental surveillance [2, 12, 23]. A specific task in agricultural usage for instance, it can be used for monitoring crops [12] located in various location across the greenhouse and using imaging device attached to it, UAV can visually monitor crop growth behavior, crop vitality, and crop stress at their early stage and get the user notified to take necessary actions. Furthermore, in indoor environment, not only monitoring, but also the earlier-mentioned material delivery [7] and part-feeding [6] tasks may also be done by UAV.

More benefits which is enabled in the UAV utilization compared to UGV is that UAV doesn't require any changes to the manufacturing environment, have a flexible moving track, and able to operates in an idle-space which are usually not used; the upper-air space. However, this convenience comes with a challenge. To utilize this flexible resource, there is a primary need to establish a coordinating and monitoring system for the UAV units to define their indoor 3D-environment-based route schedule in a safe (without collision) and effective (time-efficient) way. Specifically, this work deals with UAV trajectory operation, which requires an exact position in a particular 3D space to avoid collision with other UAV units or foreign objects. And this subject is getting popular in the market, indicated by the emerging 3D motion and depth sensing works [24, 26]. But when numerous operations are requested by a number of UAVs all the time, where each of them is requiring an immediate real-time result, it might lead to a late response due to long computational time. Not to mention, wireless connection used by the UAV is also contributing to the sluggish communication; which leads to a critical necessity of more simplified static pre-environment mapping to minimize data transmission on-the-fly.

In this study, a prior coordinate-based 3D indoor environment mapping is constructed. Consequently, the collision-free movement is pre-handled through a path planning which is built afterwards according to the environment mapping. And the proposed search concept will perform online scheduling in a time-efficient manner.

This paper is organized as follows. Section 2 provides the preliminaries which contains the required background knowledge to comprehend the problem. Section 3 describes the problem definition. Section 4 explains the proposed work. And Sect. 5 will give the conclusion of this work.

2 Preliminaries

2.1 Material Transportation Handling in Manufacturing Environment

In manufacturing environment, a lot of materials are required to be used in various activities. Those materials are going to be processed by machines in several different places. In relation with the number of transporter and available time, an efficient schedule shall be established.

As of now, UGV or also known as Automated Guided Vehicle (AGV) is a well-known mature solution for handling transportation network automation in the manufacturing environment [9]. This technology has been analyzed to be very tempting to adapt. It enables labor cost reduction, with a relatively short period of return on investment time [7]. On top of that, transportation around the hazardous area also could be avoided by human labor. With UGV, the required materials can be carried around the field, following pre-established guiding lines on the ground [7].

2.2 Vehicle Scheduling and Rescheduling

Vehicle scheduling [9] is an essential part when it comes to capacitated vehicle routing optimization. One purpose of this capacitated vehicle usage is to reduce operational costs. And in order to do that, timing of the vehicles' dispatches shall not only be time-efficient, but also distance-efficient so that the vehicles don't exhaustively moving around wasting their battery or fuel.

During the schedule execution, however, there might be uncertain events, e.g., machine breakdown and sudden battery depletion, which might occur and require a real-time scheduling system to adjust the schedule accordingly and keep the delay as minimum as possible.

Several works which focused on rescheduling [21] problem are two-phase heuristic [10] which has a scalability issue, and local rescheduling [20] which may only give local optimal solution as well. There are already various solutions can be found in the scientific literatures for capacitated routing problem (CVRP). But yet, various environments and other requirements call for a continuous further research in this area.

3 Problem Definition

In the open space of indoor environment, we face a challenge of having a flexible UAV positioning when it is running around, performing the manufacturing tasks. Figure 1 depicts the potential issue made by a UAV when it is moving forward in Y

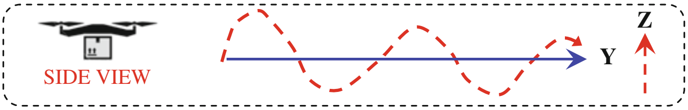

Fig. 1 Z-axis movement of UAV in 3D indoor environment

axis direction. Unlike UGV which moves on a fixed guiding track (with a fixed X and Y axis movement), UAV would have more degree of freedom in the allowed flying space. It would always be a flat distance from the Y-axis perspective. However, from the perspective of X and Z axis, the movement might be determined by the existence of obstacles, such as foreign objects and other UAVs in the 3D indoor environment.

Moreover, there might be a sudden UAV failure or machine breakdown. And instead of human bringing the carried package all the way back, there might be a possibility to hand it over to another nearby standing-by UAV beforehand, and only the broken UAV is carried out of the field. In other words, a real-time scheduling mechanism is needed. It doesn't stop only there, more hidden schedule nervousness [22] may exist before a reliable real-time scheduling system is established. The occurrences of such uncertain issues are circumstantial and causing uncertainty in the completion time of the schedule execution as well. In accordance with the aforementioned problems, some core problems are addressed as the following subsections.

3.1 Complex Environment Topology Mapping

In manufacturing environment, a complex environment is met so often, i.e., a corridor which is not having a uniform topology along the way. For instance, in a manufacturing environment, a particular corridor may have an irregular moving space due to the non-patterned formation of non-moving objects; e.g., ventilation pipes, tall machines.

And not only that, huge moving object, such as forklift, may also suddenly change the environment dramatically. A simply prior environment mapping could not solve this dynamic moving object problem completely. And rerouting on-the-fly would cause frequent delays and break the purpose of optimization from the schedule itself.

3.2 Real-Time Route Scheduling

It is important to properly assign and schedule tasks to different UGVs and UAVs in relation to the manufacturing/service needs. So far, scheduling method for fixed route is considered in many researches for UGV [6, 14, 15]. Some studies investigate the scheduling and rescheduling for a manufacturing system [1] with real-time mode [25]. While some of the others consider the rescheduling problem for UGV [20]. As for scheduling of multimodal transportation network is introduced in [3].

However, solution for real-time routing problem is missing for UAVs. Moreover, the solution to the splitting problem found in the travelling salesman problem with delivery and pick-up based on the customer's demand is only partially satisfied in [10]. Furthermore, the order splitting in this study is involving multiple UAVs utilization. And the goal here is to provide a real-time solution for putting these UAVs into use effectively; which quickly react to uncertain events [22], e.g., changes in production plan, machine and robot breakdown, without human intervention.

3.3 Collision-Free Three-Dimensional Path Routing

Path routing under capacity constraints has been studied for UGV, mobile robots, and trucks in [4, 11, 18, 19]. And time efficient scheduling using meta-heuristics or CSP is widely used [8, 16], where applications for UGV are found in [5, 13, 17]. Indeed, scheduling methods for fixed routes are considered in many researches for UGV. However, those expressed methods don't apply in UAV 3D routing. In order to avoid collision, which cause a system interruption, it is required to locate exactly a UAV unit in a particular 3D space. What being pursued in the proposed work is to develop an automatic 3D scheduling system for safe operation in indoor environments to facilitate deployment of multiple UAVs in a variety of application areas which still can't be seen today. Immediate application domains include lightweight [2] flexible logistics in the manufacturing process as well as aerial-sensor-based quality-control [23] for harsh condition of complex environment topology.

4 Approach

To be able to make a flight plan for performing tasks in manufacturing environment, it is required to generate a schedule which controls each UAV to fly on a certain route at a particular time. Also, of all things mentioned, a prior environment mapping is foremost job that should be carefully done to enable a robust path planning and highly optimized scheduling system. In this paper, concepts of approach towards the initial problem definitions are described.

4.1 Three-Dimensional-Coordinate Based Path Planning

In the initial stage of this work, the scheduling environment shall only contain 3 elements: static obstacles (wall, ventilation pipe, CCTV, lamp), feasible fields (allowed space for UAV flight), and UAVs. This means there exist dedicated areas for the UAVs to fly, where no foreign dynamic (moving) objects shall pass through; which is later called as UAV corridor.

A parameterized multiple 2D layers would be extracted from the 3D environment space. These layers will have distinct altitude in the space, and each of them contains vertices (depots) and lines (paths) which are located inside the feasible fields. An example of a UAV corridor which is later parameterized based on 2D layers for providing a feasible map for optimization process is depicted in Fig. 2.

A particular UAV will fly on the defined path in the map, and during the route calculation, occupied path at a certain time will be excluded. Thus, UAVs which are going to fly in the same corridor shall never occupy the same line; preventing the collision issue in advance. It also leads to an optimized scheduling system. Without any potential collision issue, there should be a minimum rescheduling due to line conflict; relatively small frequency of machine breakdown or unexpected battery depletion might still cause rescheduling (delay), though.

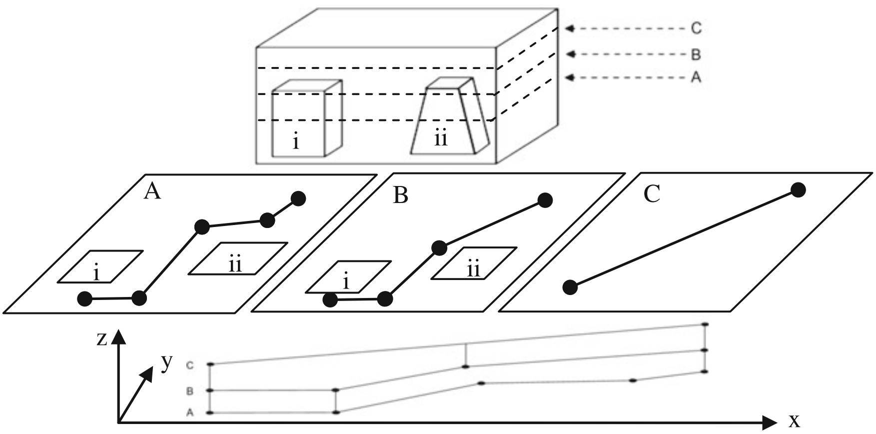

Fig. 2 Example of layer-based parameterization of 3D environment map data

Table 1 Sequence of tasks from several jobs of material transportation

Job	Sequence of tasks
1	M1-M3
2	M3-M2
3	M4-M1-M3-M2

Table 2 Processing time of machines involved in the jobs

Machine	M1	M2	M3	M4
Processing time	1	2	3	8

4.2 Multi-source Productive Best-First-Search Based Route Scheduling

Sequences of tasks from multiple jobs that will be handled in this study can be modeled into a graph. Table 1 depicts an example of collection of jobs in a manufacturing environment which requires materials being transported from a certain machine to another one in a particular sequence. Afterwards, starting task and ending task of a sequence will be called as source and sink respectively.

Followed by the processing time specifications depicted in Table 2, assuming all the tasks are having the same priority and the distances between machines are also considered, then these task sequences of jobs modeled as a diagram shown in Fig. 3.

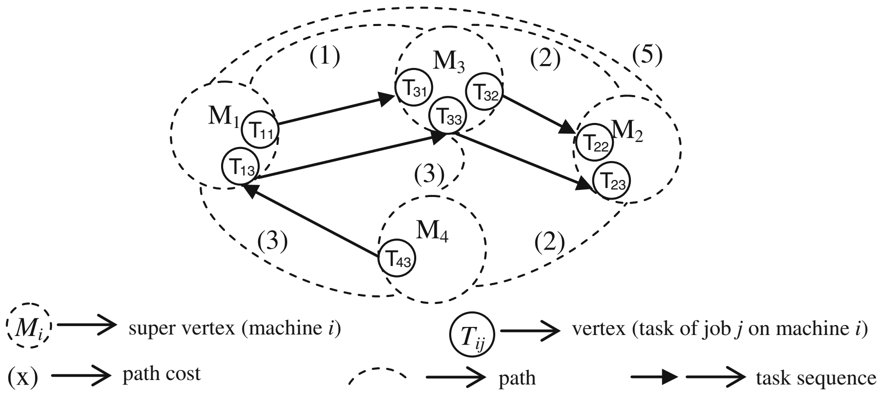

Fig. 3 Compound graph of jobs described in Table 1

In this study, multiple UAVs will be utilized and thus multiple sources are enabled to start simultaneously when possible. The execution of the tasks shall be run in an order according to the specified sequence. But which job to execute in advance and which one later is a fundamental issue here.

A particular UAV performing a certain job shall have a tendency to perform each of that job's tasks from the beginning till the end. It is caused by the zero traveling time for the UAV to just stand-by there for a while and continue the material's distribution to the next machine. In the end, it will contribute in relaxing the schedule nervousness. Thus, for the beginning, these jobs are pre-ordered without breaking them down as tasks. For that, Algorithm 1 is used to perform a preliminary scheduling, which is giving priority index to the jobs.

Algorithm 1 Priority index assignment to given jobs

```
Input: Arbitrary jobs (jobs)
Output: Priority-indexed jobs (pijobs)
for each job ∈ jobs do
  counter ← 0
  while pijobs has next element
    j ← pijobs.get (counter)
    if isEqual(j.sink, job.source) then
      k ← pijobs.get (counter+1)
      if isEqual(k.source, job.source) then
        if job.totalCost< k.totalCost then
          increment counter; break;
        endif
      else increment counter; break;
      endif
    else if isEqual(j.source, job.sink) then
      i ← pijobs.get (counter-1)
      if isEqual(i.sink, job.sink) then
        if job.totalCost< i.totalCost then break;
        endif
      else break
      endif
    endif
    increment counter
  end while
  pijobs.insertAtIndex(counter, job)
end for
```

The result of performing Algorithm 1 on the given jobs example in Table 1 is illustrated in Fig. 4.

In Algorithm 1, total cost of a job is summed up from the machine operational time and traveling time of all performed tasks. When there are multiple possibilities of pairing a sink with multiple sources (line 6–9) and a source with multiple sinks (line 14–16), the one with less total cost will be given higher priority to be made as a pair. After that, the system needs to select which sources will be used by the UAVs as their starting vertices. This selection will be prioritized on the source which doesn't have a pair. To break a tie, candidate with earlier priority-index is preferred to be set as a source. Consequently, with two UAV units for instance, the system shall start at multiple sources M_1 and M_4.

From here, multi-source productive best-first-search (PBFS) which consists of multiple PBFS running from multiple sources sharing their exploration results is run to traverse the graph depicted in Fig. 3. The rule specification for selecting the best vertex during the traversal is described in Algorithm 2. In this productive best-first-search algorithm, other than executing the next task in the sequence of a certain job, a UAV unit will try to deviate for a while, performing existing unhandled task from another task sequence at a particular time; given that he can come back in time—this will be explained more in later paragraph. In order to do this correctly, each traversal-step done by every UAV from multiple sources shall synchronously be aware of one global timeline. This synchronization of simultaneous multiple PBFS is achieved through a semaphore implementation. Only one traversal step of a source shall take place at every global iteration; which is secured by a global flag. Later on, during rescheduling due to uncertain events, executing multi-source PBFS on the remaining untraversed vertices shall consume minimized execution time as well.

Each available UAV shall start a single productive best-first-search from the designated starting vertices, while sharing the exploration result. This is to guarantee that there is no single task being performed by multiple UAVs. And every time a task is completed, the "source" label is moved to the next task in the sequence.

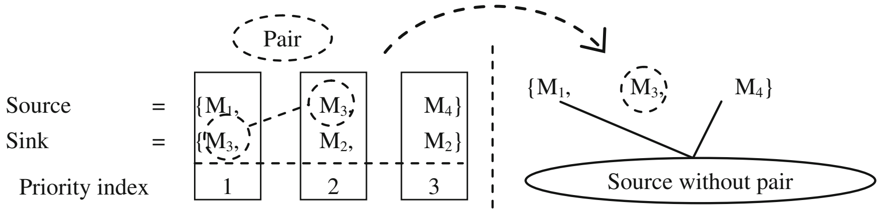

Fig. 4 Priority-indexed jobs

Algorithm 2 Productive Best First Search

```
Input: open ← global initial graph state
source ← non-paired source from priority-indexed jobs
closed ← global collection of traversed-vertices
globalTimeline ← initial time counter of all sources
while isNotEmpty(open)
  reportPersonalTimeCounter(globalTimeline)
  if isTheMinimumTimeCounter() then
    obtainFlag()
    best ← ∅
    cands ← getBestVertexCandidates(open)
    for each cand ∈ cands
      if isFromTheCurrentTaskSequence(cand) then
        if best = ∅ or best.path > cand.path then
          best ← cand endif
      else if best = ∅ or remainingIdleTime ≥
        2*cand.path then best ← cand
      endif
    end for
    add(closed, best)
    source ← best
    releaseFlag()
  else
    waitGlobalTimelineNotificationForNextIteration()
  endif
end while
```

One customization compared to the traditional best-first-search is the condition “best” in this context is not a mere path distance being seen here, but when the UAV tries to deviate to other job’s task sequence (when the current one is not finished yet), it shall see the cost c of that path as $2c$. And then recheck whether this $2c$ value is still the best option. This constraint is to guarantee that no UAV shall abandon any material in the middle of its task sequence, which potentially blocks others. In other words, UAV unit can be productive by handling other task during its idle time, but can come back in time and not abandoning its current job.

And at last, when a UAV finished its task sequence of a job, it shall look for another source vertex with Algorithm 1 and perform Algorithm 2 from there. One thing to be noted here is there is no isolated machine; this means a UAV may fly from and to any machine if required. And the goal is to get all vertices—not super-vertices—on the graph traversed, which means all tasks are executed.

5 Conclusion

A prior environment mapping, which outlines the static obstacles and free space in the field, allows the system to have input data of deterministic three dimensional flying paths. The utilization of layer based 3D path planning eliminates potential collision occurrences in advance since the system will schedule a UAV to fly through a non-occupied line. And using multi-source productive best-first-search algorithm, multiple UAVs will be utilized to perform given tasks simultaneously when possible. With the compound graph model of the tasks, regeneration of the schedule would be efficiently performed on untraversed vertices in a clear structure; which leads to a feasible real-time scheduling system. As a result, an early stage of real-time scheduling system could be arranged. The proposed approach will be implemented as a part of the planning and scheduling system for 3D-route UAV scheduling system in indoor environment.

Acknowledgment This work has partly been supported by Innovation Fund Denmark under project UAWorld; grant agreement number 9-2014-3.

References

1. Abumaizar, R.J., Svestka, J.A.: Rescheduling job shops under random disruptions. Int. J. Prod. Res. **35**(7), 2065–2082 (1997)
2. Bareth, G., Aasen, H., Bendig, J., Gnyp, M.L., Bolten, A., Jung, A., Michels, R., Soukkamaki, J.: Low-weight and UAV-based hyperspectral full-frame cameras for monitoring crops: spectral comparison with portable spectroradiometer measurements. Photogrammetrie Fernerkundung Geoinformation 69–79 (2015)
3. Bocewicz, G., Nielsen, I., Banaszak, Z.: Iterative multimodal processes scheduling. Annu. Rev. Control **38**(1), 113–122 (2014). doi:10.1016/j.arcontrol.2014.03.011
4. Corréa, A.I., Langevin, A., Rousseau, L.: Scheduling and routing of automated guided vehicles: a hybrid approach. Comput. Oper. Res. **34**, 1688–1707 (2007)
5. Dang, Q.V., Nielsen, I.: Simultaneous scheduling of machines and mobile robots. In: Corchado, J.M. (eds.) PAAMS 2013, CCIS, 365, pp. 118–128. Springer, Berlin (2013)
6. Dang, Q.V., Nielsen, I., Steger-Jensen, K., Golinska, P. (eds.) Scheduling a single mobile robot incorporated into production environment. In: EcoProduction and Logistics, pp. 185–201. Springer, Berlin (2013)
7. Davich, T.: Material handling solutions: a look into automated robotics. Wunsch Materials Handling Prize, Technical Communication Program—UW-Madison (2010)
8. Do, N.A.D., Nielsen, I., Chen, G., Nielsen, P.: A simulation-based genetic algorithm approach for reducing emissions from import container pick-up operation at container terminal. Ann. Oper. Res. 1–17 (2014)
9. Fazlollahtabar, H., Saidi-Mehrabad, M.: Methodologies to optimize automated guided vehicle scheduling and routing problems: a review study. J. Intell. Robot. Syst. **77**(3–4), 525–545 (2015)
10. Ganesh, K., Narendran, T.T.: TASTE: a two-phase heuristic to solve a routing problem with simultaneous delivery and pick-up. Int. J. Adv. Manuf. Technol. **37**(11–12), 1221–1231 (2008)

11. Golden, B.L., Raghavan, S., Wasil, E.A.: The Vehicle Routing Problem: Latest Advances and New Challenges, vol. 43. Springer, Berlin (2008)
12. Hampapur, A., Li, J., Pankanti, S., Otto, C.A.: Method for performing visual surveillance of moving objects by using unmanned aerial vehicle for civilian purpose, involves performing motion segmentation of images, and tracking moving objects to facilitate visual surveillance of objects. U.S. Patent No. US2013002866-A1, 3 Jan 2013
13. Hvilshøj, M., Bøgh, S., Nielsen, O.S., Madsen, O.: Autonomous industrial mobile manipulation (AIMM): past, present and future. Ind. Robot Int. J. **39**(2), 120–135 (2012)
14. Kats, V., Levner, E.: A faster algorithm for 2-cyclic robotic scheduling with a fixed robot route and interval processing times. Eur. J. Oper. Res. **209**(1), 51–56 (2011)
15. Möhring, R.H., Köhler, E., Gawrilow, E., Stenzel, B.: Conflict-free real-time AGV routing. In: Operations Research Proceedings, pp. 18–24. Springer, Berlin (2004)
16. Nielsen, I., Lim, M., Nielsen, P.: Optimizing supply chain waste management through the use of RFID technology. In: IEEE International Conference on RFID-Technology and Applications (RFID-TA), 2010, pp. 296–301 (2010)
17. Nielsen, I., Bocewicz, G., Do, N.A.D., Brunoe, T.D. et al. (eds.): Production and resource scheduling in mass customization with dependent setup consideration. In: LNPE, pp. 461–472. Springer, Switzerland (2014). doi:10.1007/978-3-319-04271-8_39
18. Nielsen, I., Dang, Q. V., Nielsen, P., Pawlewski, P. Scheduling of mobile robots with preemptive tasks. advances in intelligent systems and computing, pp. 19–27. Springer, Berlin (2014). doi:10.1007/978-3-319-07593-8_3
19. Nielsen, I., Dang, Q.V.: Bocewicz, G., Banaszak, Z.: A methodology for implementation of mobile robot in adaptive manufacturing environments. J. Intell. Manuf. doi:10.1007/s10845-015-1072-2
20. Nishi, T., Ando, M., Konishi, M.: Experimental studies on a local rescheduling procedure for dynamic routing of autonomous decentralized AGV systems. Robot. Comput. Integr. Manuf. **22**, 154–165 (2006)
21. Spliet, R., Gabor, A.F., Dekker, R.: The vehicle rescheduling problem. Comput. Oper. Res. **43**, 129–136 (2013)
22. Vieira, G.E., Herrmann, J.W., Lin, E.: Rescheduling manufacturing systems: a framework of strategies, policies, and methods. J. Sched. **6**(1), 39–62 (2003)
23. von Bueren, S.K., Burkart, A. Hueni, A., Rascher, U., Tuohy, M.P., Yule, I.J.: Deploying four optical UAV-based sensors over grassland: challenges and limitations. Biogeosciences **12**, 163–175 (2015)
24. Wang, C., Zhu, Z.Y., Chan, S.C., Shum, H.Y.: Real-time depth image acquisition and restoration for image based rendering and processing systems. J. Signal Process. Syst. Signal Image Video Technol. **79**(1), 1–18 (2015)
25. Yamamoto, M., Nof, S.Y.: Scheduling/rescheduling in the manufacturing operating system environment. Int. J. Prod. Res. **23**(4), 705–722 (1985)
26. Zhang, S., Wang, C., Chan, S.C.: A new high resolution depth map estimation system using stereo vision and kinect depth sensing. J. Signal Process. Syst. Signal Image Video Technol. **79**(1), 19–31 (2015)

Material Supply Scheduling for a Mobile Robot with Supply Quantity Consideration—A GA-based Approach

Izabela Nielsen, Ngoc Anh Dung Do, Peter Nielsen and Yohanes Khosiawan

Abstract Mobile robots have been used to transport materials in manufacturing and services. The cycle of material supply is dependent on not only processing rate but also the supply quantity. Due to the limit on carrying quantity of robot, the problem to determine the material supply quantity and material supply schedule without lack of materials for production or service processes becomes complicated. In this paper, the problem to schedule material supply and determine material supply quantity is formulated as a nonlinear program. A heuristic algorithm based on genetic algorithm is developed to solve the problem. The conducted numerical experiment shows the good performance of the proposed algorithm which can be implemented to solve large scale problem.

Keywords Robot scheduling · Material supply scheduling · Genetic algorithm

1 Introduction

In manufacturing and service systems, especially flexible manufacturing system, the automatic vehicles based material handling system is applied. The automatic vehicles such as automatic guided vehicles, mobile robots, and unmanned aerial vehicles play an important role in handling materials including raw materials, work-in-process parts, unfinished assembly parts and moving around the machines

I. Nielsen · N.A.D. Do (✉) · P. Nielsen · Y. Khosiawan
Department of Mechanical and Manufacturing Engineering,
Aalborg University, Aalborg, Denmark
e-mail: ngoc@m-tech.aau.dk

I. Nielsen
e-mail: izabela@m-tech.aau.dk

P. Nielsen
e-mail: peter@m-tech.aau.dk

Y. Khosiawan
e-mail: yok@m-tech.aau.dk

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_4

and stations to assure the smooth flow of materials, and hence, improve the productivity. Moreover, nowadays, customers prefer highly customized product and changing from traditional production policy where high-leveled customization are produced with high production cost to mass customization which can reduce production cost while keeping the same quality of product and on-time delivery to satisfy diversified demands will bring the success to the company [4, 6]. In these highly flexible manufacturing and service systems, automatic vehicles are broadly used. Moreover, automatic vehicles can be operated in dangerous and hazardous environment. The main role of automatic vehicle is deliver unfinished parts from warehouses and/or machines to other machines following the production process. In addition, more functions are designed on an automatic vehicle to assist the production process and this makes the schedule problem more complex. Thus, considering the problem to schedule an automatic vehicle to serve the production process is necessary.

This paper focuses on the material supply scheduling problem for a mobile robot, an automatic vehicle. The robot delivers materials from warehouse to the buffers of machines. The supply schedule must assure that there are no material shortages at all machines. To save the energy consumption which mainly occurs due to the moving activity of the robot, the supply schedule should minimize the travel distance of the robot. For a single machine, a large amount of supply quantity can reduce the number of visits, and hence, reduce the travel distance of robot. However, this cannot be implemented for the case with multiple machines while the carrying capacity of the robot is limited. If the mobile robot carries too many materials to supply a machine, it has no room to carry materials for other machines. The consequence is that the robot has to move longer distance because it has to back to the warehouse more frequently. Moreover the length of the cycle to supply a machine is proportional to the supply quantity and different cycle length leads to different supply schedule of the robot. Hence, the problem to find the optimal supply quantity for each machine and the supply schedule for the mobile robot is necessary and this is the novelty of this paper.

The problem to schedule the operations including material loading, moving among warehouse and machines and material unloading for a mobile robot cannot be treated as an Asymmetric Traveling Salesman Problem due to some additional constraints such as the carrying capacity of mobile robot and multiple visits. Some papers have been studied on the robot task-sequencing problem. Ascheuer et al. [1] considered the sequential ordering problem which is closed to the robot task-sequencing problem. Edan et al. [8] developed a near minimum task planning problem for a fruit harvesting robot with N fruit locations. Dang et al. [5] propose an MIP model to obtain the optimal feeding sequence of a mobile robot in a manufacturing cell. Some studies have done by Relich and Muszyński [12] and Do et al. [7] for planning and scheduling in manufacturing and services. There have been some papers solving the robot task-scheduling using heuristic algorithm such as nearest neighbor rule [9], closest insertion algorithm [2], and dispatching rules [15]. Other papers implemented metaheuristic to solve the combinatorial optimization problem, especially for robot task scheduling problem such as tabu search algorithm [10], neural network [11], and genetic algorithm [16]. Bocewicz et al. [3] considered the task scheduling problem for scheduling and rescheduling production process based on constraint satisfaction programming. Sitek [13, 14] implemented a hybrid approach including mathematical programming and constraint programming to schedule a supply chain system.

The materials can be unloaded to the buffers when the material levels at the buffers are lower or equal the minimum level. Furthermore, materials must be supplied before the material levels at the buffers reach zero to assure the machines have enough materials to work continuously. Therefore, there is a time window for material supply. Some papers focus on the practical problem of scheduling single robot with time windows. Moreover, the maximum supply quantity depends on both the limitation of storage capacity at the buffers and the carrying capacity of the robot. This makes the robot task scheduling becomes more practical and more complicated. An attempt to solve this complicated scheduling problem has been done in which (s, Q) policy is used to decide the supply quantity [6]. However, they assumed that the robot fills materials up to the maximum capacity of buffers. This is not true because if the mobile robot carries too many materials for a machine, it has no room to carry other materials for other machines due to the limit on carrying capacity. To serve the next machine, the robot has to go back to warehouse. The consequence is that the robot may not supply materials to the next machine within the required time window. Therefore, determining the material supply quantity is necessary in practice.

2 Problem Description and Solution Algorithm

2.1 Problem Formulation

In this paper, the considered system which is shown in Fig. 1 has a mobile robot to deliver the materials for a set of machines following the production process. There is buffer for materials at each machine with minimum and maximum levels of material storage. The supply schedule must assure that: (1) materials are delivered to a machine when the material level at the buffer of that machine is lower than the minimum level; (2) and there is no lack of materials at all machines. The robot should be scheduled to satisfy the supply constraint with minimum energy consumption. The mobile robot mainly consumes energy for its moving and this energy consumption is proportional with the moving distance. In a finite planning horizon, the robot can supply material to a machine many times. The number of supplies is inversely proportional to the units of material per supply which is simply called supply quantity in this paper. Increasing the supply quantity reduces the energy consumption for moving robot. However, the supply quantity is limited because of the capacity to carry materials and the storage capacity at the buffers. Moreover, in a multi-machine system, the supply quantity cannot be too large because this may

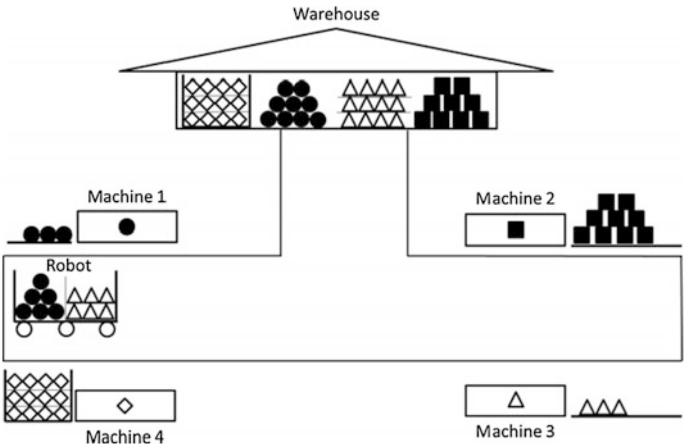

Fig. 1 The components of the considered production system

lead to infeasible solution. Assuming that two machines has two supply time windows which are very close. If we try to load as many materials as possible to supply only the first machine, then the robot cannot deliver material to the second machine on time because it has to back to warehouse, load materials, and move to the second machine. To overcome this situation, the supply quantities of two machines must be reduced so that they are assigned to a route. However, reducing supply quantity increases the number of supplies, and hence, increases the energy consumption. Thus, supply quantity is one of the decision variables of the problem.

Each machine can only process on one type of materials with constant processing time meaning that the demand rate on materials is also constant. The robot, with limit on carrying capacity, loads materials at a warehouse and deliver these materials to machines. When the robot arrives at a machine, material is unloaded to the machine buffer which has limit of storage capacity. These operations are repeated until the end of planning horizon and a replication can be called as a route because the robot will start at the warehouse and return to the warehouse after visiting some machines. The robot can visit one or more machines on a route. It is assumed that the mobile robot moves with constant speed. The supply quantity has to be assured that the material level at the buffer does not exceed maximum level after unloading material from the robot. To assure this condition, the supply quantity should be less than the difference between maximum level and minimum level of the buffer.

The machine consumes materials from its buffer for its process. The maximum and minimum levels of materials at the buffer are known in advance. The time window to supply material to a machine is determined from the time that the material at the buffer reaches the minimum level to the time that the material level at the buffer is zero without material supply. The material must be replenished before the material level at the buffer downs to zero meaning that the robot has to deliver material within the time window. With the constant processing rate and fixed supply quantity, the supply time window can be easily calculated which is shown in the section below. It is noted that the length of the supply time window depends on the demand rate on material and minimum level and be calculated by the following formula:

$$\text{Length of time window} = \text{processing time per material unit} * \text{minimum level}$$

The length of supply cycle depends on the demand rate and supply quantity. It is calculated as follows:

$$\text{Length of supply cycle} = \text{processing time per material unit} * \text{supply quantity}$$

Another aspect considered in this problem is the sequencing to supply materials. When the supply quantities of all machines are known, the supply time windows of machines are determined. The problem is similarly to a capacitated vehicle routing with time window problem (CVRPTWP), in which only one truck is used to deliver goods to customers [6]. However, different from CVRPTWP, the robot can arrives at a machine before the supply time window of that machine without penalty cost. The robot can wait at a machine without energy consumption, and therefore, the arrival time and the supply time at a machine may be different. It is noted that the time required for unloading materials to the buffer of a machine is assumed to be proportional to the unloading quantity. At the warehouse, there are operators put the materials into small load carriers. When the robot back to warehouse, it releases the empty small load carrier and takes the filled small load carriers to start a new supply. Hence, the duration for loading materials to robot can be omitted.

Finding a sequence of material supply to minimize the energy consumption is the second decision making of this problem. The sequence of material supply should minimize the total travel distance of the robot. It can be seen that there is a repetition of material supply at each machine. If we call Tmin is the smallest common multiple of supply cycle of all machines, the sequence of material supply of the robot will repeat after Tmin. Therefore, we can limit the planning horizon for determining the sequence of material supply of the robot to Tmin. The value of Tmin is dependent on the supply quantity of each machine because the supply cycle depends on the supply quantity.

In general, the materials processed on different machines are different. All required materials are located at the warehouse. Travel times from warehouse/machines to warehouse/machines of the robot are known in advanced. The initial material levels at the buffers of all machines are supposed at the maximum level.

The problem formulation is shown as below. Some notations are introduced to present the sets of machines and warehouse. The parameters include the processing time, material unloading rate, travel time among machines and warehouse, the maximum and minimum material levels at the buffers and the carrying capacity of the robot. Two main decision variables are the sequence of material supply and the quantity supply are introduced after the parameters.

Notation

M: set of machines, $M = \{1,2,\ldots, I\}$
N: set of location including machines and warehouse, $N = M \cup \{0\}$, 0 represents warehouse

Parameters

p_i: processing time on a material unit at machine i
c_i: material unloading rate (unit of time/part)
t_{ij}: the travel time from machine i (or warehouse) to machine j (or warehouse), $i, j \in N$
u_i: the maximum level at the buffer of machine i
v_i: the minimum level at the buffer of machine i
K: the carrying capacity of the robot
T: planning horizon

Variables

Q_i: supply quantity at machine i
S_{ih}: the starting time of h-th supply time window at machine i
E_{ih}: the ending time of h-th supply time window at machine i
F_{ih}: the time that robot starts the h-th supply at machine i

$$X_{jl}^{ih} = \begin{cases} 1 & \text{if robot moves from machine } i \text{ for } h\text{-th supply to machine } j \\ & \text{for } l\text{-th supply} \\ 0 & \text{otherwise} \end{cases}, \quad i,j \in N$$

If i equals zero in the variable X_{jl}^{ih}, it means that the robot starts from the warehouse. Similarly, if j equals zero, it means that the robot backs to the warehouse.

As mention above, the value of S_{ik} and E_{ik} can be determined when Q_i is known. The formula to calculate S_{ik} and E_{ik} are shown as follows

$$S_{ih} = \begin{cases} (u_i - v_i)p_i & h = 1 \\ S_{i,h-1} + Q_i p_i = S_{i1} + (h-1)Q_i p_i & 1 < h \leq \left\lfloor \frac{T-(u_i-v_i)p_i}{Q_i p_i} \right\rfloor \end{cases}$$

$$S_{ih} = S_{i1} + (h-1)Q_i p_i = [u_i - v_i + (h-1)Q_i]p_i \quad 1 \leq h \leq \left\lfloor \frac{T - (u_i - v_i)p_i}{Q_i p_i} \right\rfloor = H_i$$

$$E_{ih} = S_{ih} + v_i p_i = [u_i + (h-1)Q_i]p_i \quad 1 \leq h \leq h_i$$

It can be seen that the feeding cycle is $Q_i p_i$ for machine i.
The mathematical model for the problem can be formulated as follow
Objective function:

$$\text{Min} \sum_{i \in N} \sum_{1 \leq h \leq H_i} \sum_{j \in N} \sum_{1 \leq l \leq H_j} t_{ij} X_{jl}^{ih}$$

Subject to

$$\sum_{i \in N} \sum_{1 \leq h \leq H_i} X_{jl}^{ih} = 1 \quad \forall j \in N, l \in [1, H_j] \tag{1}$$

$$\sum_{j \in N} \sum_{1 \leq l \leq H_j} X_{jl}^{ih} = 1 \quad \forall i \in N, h \in [1, H_i] \tag{2}$$

$$\sum_{j \in N} \sum_{1 \leq l \leq H_j} X_{jl}^{ih} = \sum_{m \in N} \sum_{1 \leq n \leq H_m} X_{ih}^{mn} \quad \forall i \in M, h \in [1, H_i] \tag{3}$$

$$\sum_{i \in M} X_{i1}^{01} = 1 \tag{4}$$

$$S_{ih} \leq F_{ih} \leq E_{ih} \quad \forall i \in M, h \in [1, H_i] \tag{5}$$

$$F_{jl} \geq F_{ih} + c_i Q_i + t_{ij} X_{jl}^{ih} - L\left(1 - X_{jl}^{ih}\right) \quad \forall j \in N, l \in [1, H_j], i \in M, h \in [1, H_i], \tag{6}$$

$$F_{jl} \geq F_{0h} + t_{ij} X_{jl}^{0h} - L\left(1 - X_{jl}^{0h}\right) \quad \forall j \in N, l \in [1, H_j], h \in [1, H_i], \tag{7}$$

$$\sum_{\substack{i \in M, l \in [1, H_i], \\ F_{0h} \leq F_{il} \leq F_{0,h+1}}} Q_i \leq K \quad \forall h \in [1, H_i] \tag{8}$$

$$F_{ih} \leq F_{i,h+1} \quad \forall i \in N, h \in [1, H_i - 1] \tag{9}$$

$$Q_i \leq u_i - v_i \quad \forall i \in M \tag{10}$$

$$Q_i, S_{ih}, E_{ih}, F_{ih} \geq 0, X_{jl}^{ih} \in \{0, 1\} \quad \forall i, j, h, l$$

The objective is to minimize the travelling distance of the robot. Constraints (1), (2), and (3) are to assure a feasible route for the robot. Constraint (4) presents that the robot will start at the warehouse at the beginning. Constraint (5) assures that the machine can work continuously. Constraint (6) and (7) show the sequence of supply. It guarantees that the robot can visit the next machine when it finished supplying at the previous machine. Constraint (8) avoids the amount of carried materials over the carrying capacity of the robot. Constraint (9) assures that the next supply at a machine should happen after the current suppl. Constraint (10) is used to guarantee that the machine is not over supply.

2.2 Solution Algorithm

Constraint (8) to assure that the total supply quantities carried by the robot on a route is not over the carry capacity. However, the starting time of a route depends on the value of the supply quantity at each machine. This constraint is a nonlinear constraint and the above mathematical model is a nonlinear programming. Therefore, an algorithm should be developed to solve the problem.

When Q_i is fixed, it can be seen that the problem becomes similarly to a task sequencing problem of robot. Sih and Eih are fixed and they have a role of release time and due date. Then, the sequence to feed machines can be listed in the order of ending time of each machine. Therefore, with given Q_i, a heuristic algorithm is proposed to make the feeding sequence of robot and shown as follows

Step 1. With given Q_i calculate S_{ih} and E_{ih}.
Step 2. Sort E_{ih} in non-descending order. If $E_{ih} = E_{jl}$, i is listed before j if $S_{ih} \leq S_{jl}$; otherwise, j is listed before i. The list proposes the sequence of feeding.
Step 3. Consider two machines i and j which are next in the list. Suppose that i is listed before j

- If $F_{ih} + c_i Q_i + t_{i0} + t_{0j} > E_{jl}$, check the condition $Q_i + Q_j \leq K$
  - If the condition is satisfied, i and j are assigned to the same route
  - Otherwise, decrease Q_i and go back to Step 2
- If $S_{ih} > S_{jl}$:
  - Machine j should be feed before machine i when robot can arrive i before the ending time of i.
  - Otherwise, machine i should be feed before machine j with the condition that robot can arrive j before the ending time of j.

Step 4. Assign route where each route served one machine
Step 5. Merging two routes if $Q_i + Q_j \leq K$

Hence, an approach to solve the problem is to generate the value of Q_i and then to determine the feeding sequence for the robot. Genetic algorithm is one of the best approaches to search a large population of feasible solutions and to carry out the best solution. In this paper, a heuristic algorithm based on genetic algorithm is proposed to solve the problem. The main difference from genetic algorithm is that a heuristic algorithm for determining feeding sequence is included. Each chromosome will give a set of values of Q_i s and the developed algorithm is executed to find the feeding sequence. The fitness of a chromosome is evaluated based on the total travel distanced.

3 Numerical Experiment

The numerical experiment is conducted with a system with a mobile robot which feeds 4 machines. The maximum and minimum level at the buffer of each machine, the unloading rate, the processing time on an unfinished part at a machine, and the traveling time from warehouse and machines are shown in Fig. 2 and Table 1. The planning horizon is about 45 min. The carrying capacity will be 1500 parts. A pilot experiment is conducted to find the appropriate values for the parameters of genetic algorithms. The result from this pilot experiment shows that the population size, the maximum number of generations, and the mutation rate are 50, 40, and 0.01, respectively.

The result from the algorithm is present in Table 2 with the computation time less than 1 s. Figure 3 shows the result of robot moving and material unloading.

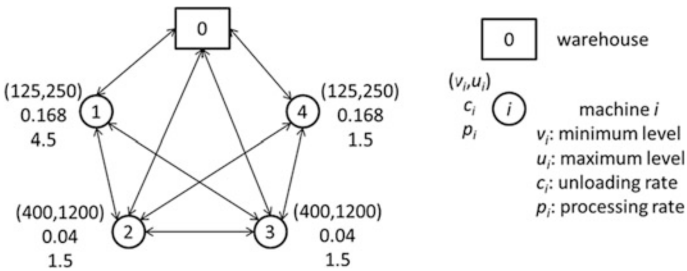

Fig. 2 Machine information for the experiment

Table 1 Traveling time of robots among warehouse and machine (in seconds)

	Warehouse	Machine 1	Machine 2	Machine 3	Machine 4
Warehouse	0	34	37	34	40
Machine 1	39	0	17	34	50
Machine 2	35	17	0	35	49
Machine 3	34	33	35	0	47
Machine 4	36	47	48	46	0

Table 2 The result of numerical experiment

	Machine/warehouse	Arrive at (seconds)	Unload at (seconds)	Unload amount (parts)
Route 1	4	40	(187.5; 375; 562.5; 750)	(125; 125; 125; 125)
0–4–1–4–1–4–0	1	818	818	125
	4	889	(937.5; 1125; 1312.5)	(125; 125; 125)
	1	1380.5	1380.5	125
	4	1451.5	(1500; 1687.5; 1875)	(125; 125; 125)
	0	1932		
Route 2				
0–2–0	2	1969	1969	1100
	0	2048		
Route 3				
0–3–1–4–0	3	2082	2082	1100
	1	2159	2159	125
	4	2230	2230	125
	0	2287		
Route 4				
0–4–1–4–0	4	2327	(2327; 2437.5)	(125; 125)
	1	2505.5	2505.5	125
	4	2576.5	2625	125
	0	2682		

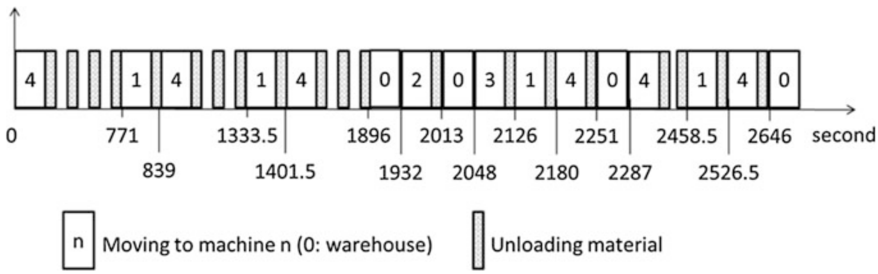

Fig. 3 Robot moving and material unloading

4 Conclusion

Using robot to handle materials in manufacturing and services is more and more broadly applied in real world. It is necessary to study on the material supply scheduling problem for robot to assure the smooth flow of material and increase the productivity as well as the flexibility of the manufacturing system and services. This paper focuses on the material supply scheduling for a manufacturing system including a mobile robot and multiple machines. Due to the carrying capacity of robot and storage capacity of machine buffers, material supply quantity is taken in account. The problem is formulated as a nonlinear programming, and a GA-based heuristic algorithm is developed. The numerical result shows that the proposed algorithm can be applied to solve the practical problem with large scale in short time. For further investigation, the different supply quantities on different visits at a machine should be taken in account. The problem is more practical if robot battery charging is considered. On the other hand, the solution quantity can be improved if metaheuristic can be applied on both determining the supply quantity at a machine and the supply sequencing. However, the computation time can be longer and this trade-off (solution quality vs. computation time) can be considered in future.

References

1. Ascheuer, N., Escudero, L.F., Grötschel, M., Stoer, M.: A cutting plane approach to the sequential ordering problem (with applications to job scheduling in manufacturing). SIAM J. Optim. **3**(1), 25–42 (1993)
2. Askin, R.G., Standridge, C.R.: Modeling and analysis of manufacturing systems, vol. 29. Wiley, New York (1993)
3. Bocewicz, G., Izabela, N., Zbigniew, B.: Iterative multimodal processes scheduling. Ann. Rev. Control **38**(1), 113–122 (2014)
4. Bzdyra, K., Banaszak, Z., Bocewicz, G.: Multiple project portfolio scheduling subject to mass customized service. In: Progress in Automation, Robotics and Measuring Techniques. Series: Advances in Intelligent Systems and Computing, vol. 350, pp. 11–21 (2015)

5. Dang, Q.V., Nielsen, I.E., Steger-Jensen, K.: Scheduling a single mobile robot for feeding tasks in a manufacturing cell. In: Proceedings of International Conference Advances Production Management System. Norway (2011)
6. Dang, Q.V., Nielsen, I., Steger-Jensen, K., Madsen, O.: Scheduling a single mobile robot for part-feeding tasks of production lines. J. Intell. Manuf. 1–17 (2013)
7. Do, N.A.D., Nielsen, I.E., Chen, G., Nielsen, P.: A simulation-based genetic algorithm approach for reducing emissions from import container pick-up operation at container terminal. Annals of Operations Research. (2014, Article in Press)
8. Edan, Y., Flash, T., Peiper, U.M., Shmulevich, I., Sarig, Y.: Near-minimum-time task planning for fruit-picking robots. Robot. Autom. IEEE Trans. **7**(1), 48–56 (1991)
9. Han, M.H., McGinnis, L.F., Shieh, J.S., White, J.A.: On sequencing retrievals in an automated storage/retrieval system. IIE Trans. **19**(1), 56–66 (1987)
10. Hurink, J., Knust, S.: A tabu search algorithm for scheduling a single robot in a job-shop environment. Discrete Appl. Math. **119**(1), 181–203 (2002)
11. Maimon, O., Braha, D., Seth, V.: A neural network approach for a robot task sequencing problem. Artif. Intell. Eng. **14**(2), 175–189 (2000)
12. Relich, M., Muszyński, W.: The use of intelligent systems for planning and scheduling of product development projects. Procedia Comput. Sci. **35**, 1586–1595 (2014)
13. Sitek, P., A hybrid CP/MP approach to supply chain modelling, optimization and analysis, Federated Conference on Computer Science and Information Systems (FedCSIS), 2014,1345–1352. DOI: 10.15439/2014F89
14. Sitek, P.: A concept of decision support in supply chain management—a hybrid approach. Beyond Databases Archit. Struct. Commun. Comput. Inf. Sci. **521**, 565–574 (2015)
15. Suárez, R., Rosell, J.: Feeding sequence selection in a manufacturing cell with four parallel machines. Robot. Comput. Integr. Manufact. **21**(3), 185–195 (2005)
16. Zacharia, P.T., Aspragathos, N.A.: Optimal robot task scheduling based on genetic algorithms. Robot. Comput. Integr. Manufact. **21**(1), 67–79 (2005)

An Integrated Declarative Approach to Decision Support for Scheduling Groups of Jobs

Jarosław Wikarek

Abstract This study deals with scheduling groups of jobs, their arrival and delivery, and individual processing of each of them. All jobs in a group should be delivered at the same time after processing. The author presents a novel way of modeling and decision support—an integrated declarative approach. This approach includes the modeling and solving of the problem in the integrated environment, composed of MP (Mathematical Programming) and CLP (Constraint Logic Programming). In addition, the paper proposes the transformation and linearization of the model in CLP/MP environment and implementation framework. Opportunity to ask questions and receive quick answers, taking into account any constraints, conditions and dynamics, constitutes an invaluable support to the manager.

Keywords Mathematical programming · Constraint logic programming · Optimization · Decision support systems · Scheduling · Group work · Soft constraints

1 Introduction

The proposed research problem—scheduling groups of jobs—finds many applications in industrial companies, including but not limited to food, ceramic tile, textile production industries, distributions, supply chain, installation of bulky equipment, manufacturing of complex devices, service of restaurants etc. It can be noticed in many production and logistic industries that have different customers. Assume that each customer has different orders. Each order has a different process function and resources, but all items ordered by a customer or group of customers should be delivered at the same time in one package to reduce the transportation costs, subsequent processing steps time and costs or/and assure proper quality of the product/service and customer satisfaction. The author presents a novel way of

J. Wikarek (✉)
Department of Control and Management Systems, Kielce University of Technology, Kielce, Poland
e-mail: j.wikarek@tu.kielce.pl

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_5

modeling and decision support in scheduling groups of jobs—an integrated declarative approach.

The remainder of the article is organized as follows. Section 2 presents a literature review and motivation. Problem statement, research methodology, mathematical model and contribution are provided in Sect. 3. Computational examples, tests of the implementation platform are presented in Sect. 4. Possible extensions of the proposed approach as well as the conclusions are included in Sect. 5.

2 Literature Review and Motivation

Scheduling methods for optimal and simultaneous provision of service to groups of customers are proposed most often in the flexible flow-shop system (FFS). In the FFS system, processing is divided into several stages with parallel resources at least in one stage. All of the tasks should pass through all stages in the same order [1, 2]. The exemplified objectives of the problem [2] are minimizing the total amount of time required to complete a group of jobs and minimizing the sum of differences between the completion time of a particular job in the group and the delivery time of this group containing that job (waiting period).

Flexible flow shop scheduling problems are usually solved by branch and bound algorithms [11] and by integer programming [10, 12]. A major disadvantage of the integer programming approaches to scheduling is the need for solving large mixed-integer programs to obtain meaningful optimal solutions [13].

Decomposition approaches divide the overall scheduling problem into segments that are considered consecutively neglecting the interdependencies of the different segments requiring strategies to effectively handle those. There are stage-oriented, job-oriented, and problem-oriented decomposition approaches [14].

The motivation was to develop a method that allows modeling and optimization decisions for scheduling groups of jobs with the same date of completion for various forms of organization (flow-shop, job-shop, open-shop etc.). Development of decision-making models, whose implementation using the proposed method will allow obtaining quick answers to key questions asked by a service manager. The method takes into account the dynamics resulting from the coming of new groups of orders while previous orders are handled.

3 Problem Statement and Methodology

The majority of models presented in the literature (Sect. 2) refer to a single problem and optimization according to the set criterion. Fewer studies are devoted to multiple-criteria optimization by operations research (OR) methods [2]. The constraint-based environment [3, 7, 8, 15, 16] offers a very good framework for representing the knowledge and information needed for the decision support in the scheduling problems. The central issue for a constraint-based environment is a constraint satisfaction problem (CSP). The classical CSP is a well-known paradigm that is suited to specify many kinds of real life and practical problems such as scheduling, planning, timetabling, routing and so on. CSP has been broadly investigated in Computer Science and Artificial Intelligence [3, 16]. The key idea of a CSP is to solve a problem by stating constraints representing requirements about the problem and then, finding solutions satisfying all the constraints.

Classical and pure CSPs are not suited in several real-life problems. For instance, CSPs are not able to model constraints that are preferences rather than hard requirements or to provide a not so-bad solution when the problem is over constrained. Many works in the literature [17] focus on generalizations of classical (hard) constraints to soft constraints. One of the environments that use CSPs is CLP. Declarative environments such as CLP facilitate problem modeling and introduction of logical and symbolic constraints [3, 16]. Unfortunately, high complexity of decision-making models and their integer nature contribute to poor efficiency of modeling in OR methods and inefficient optimization in CLP. Therefore, a new approach to modeling and solving these problems was developed [8]. A declarative environment was chosen as the best structure for this approach [4–7]. Mathematical programming environment was used for problem optimization [9]. This mixed and integrated approach [8, 18, 19] is the basis for the creation of the implementation environment to support managers. In addition to optimizing particular decision making problems connected with groups of orders, such environment allows asking various questions while processing the orders.

The main contribution on our part is the new method for the modeling and optimization of decision-making problems for handling orders in groups. It is based on the integration of CLP and MP environments (Sect. 3.3). In addition, the formal model for handling orders in groups with hard and soft constraints has been proposed.

Based on the proposed method and model, we designed the environment that allows to ask questions and get fast answers in the process of handling groups of orders. The algorithm presented in [8] enables dynamic use of this method.

3.1 Problem Description

This problem can be stated as follows (Fig. 1). Orders ($j = 1\ldots J$) enter the system in groups at different periods ($h = 1\ldots V$). Each order consists of operations ($o = 1\ldots O$) and should be processed with specific resources, including parallel resources ($k = 1\ldots K$). The orders ($j = 1\ldots J$) in each group should be delivered at the same time. It is assumed that all processors in the last stage are eligible to process all jobs. This assumption is valid due to the fact that processors in the last stage (packers in a factory, quality control, and waiters at restaurants) are the same in most application areas of the proposed problem. Special points at which orders are submitted and then delivered are introduced ($i = 1\ldots I$). The problem does not cover the

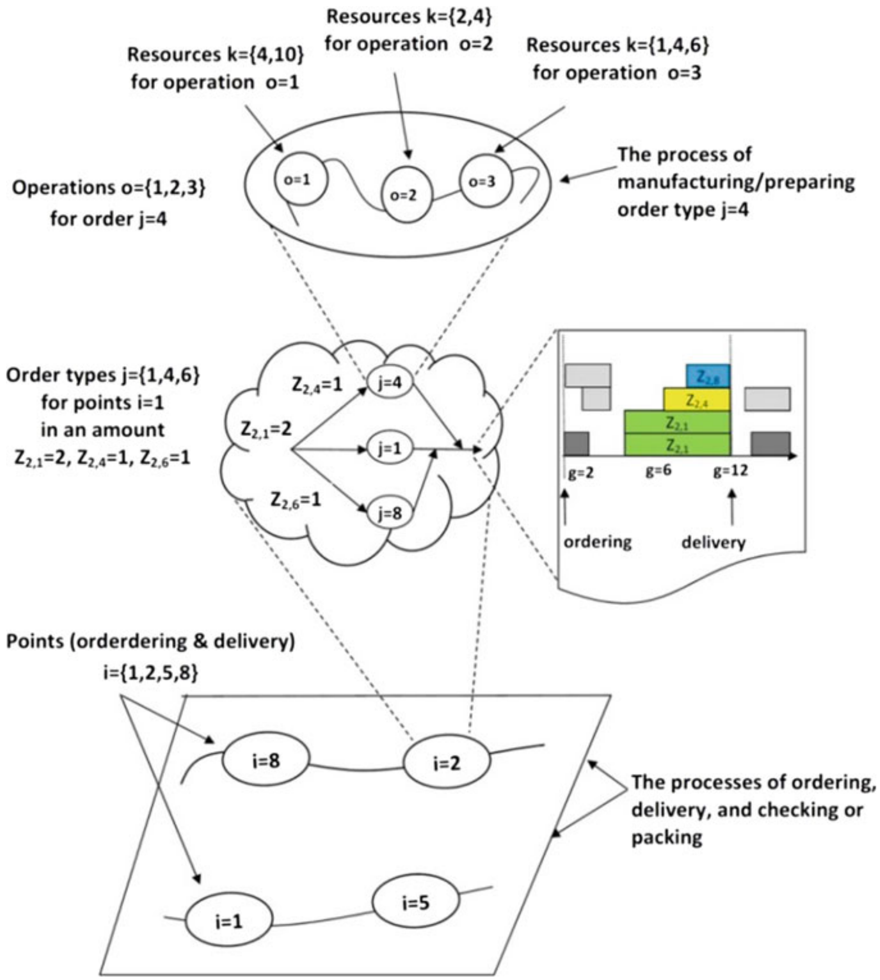

Fig. 1 Scheme for the problem of handling orders in groups

configuration of the points but relates to handling orders, as many orders may come from one customer/orders several items from the list of item/. Each order may be processed by a different resource set in any order.

Possible questions for this model are:

- What is the minimum average waiting time in each group? (Q1)
- Can the order be delivered within *g* period? (Q2)
- Which order cannot be delivered in each group of orders within *g* period? (Q3)
- Which order can be prepared in the number of M within *g* period for points *i*? (Q4)

- With the specified number of resources *rzg*/i.e. packers, waiters/, is it possible to deliver orders to all groups within N time? (Q5)
- Is it possible to deliver orders of all groups with resources d_1, d_2, …d_k within *g* period? (Q6)

3.2 Mathematical Model

A mathematical model is developed for the research problem. This model based on [8]. The sets, indices, parameters, decision variables are presented in Table 1.

Objective function
Minimization of average waiting time at each point *i* (1) is one of the objective functions that work well in case of such problems.

Constraints
Constraint (2) specifies the moment (period) from which resource k is needed to execute order j. Constraint (3) states no start is possible before orders appear. Constraint (4) ensures that the number of available resources k in period g is not exceeded. Constraint (5) provides resource occupancy for the time of the order execution. Resource k is used without interruption during the execution of order j from point i (6). Constraint (7) is for determining decision variable Y. Simultaneous completion of orders j from the given point is ensured by constraint (8).

To linearize this model, an ancillary variable was used, Wi, g = {0,1}, determined according to constraint (12) (where coefficients/factors ppg are determined by the CLP). Constraint (10) determine the end of the resource k occupancy. Constraint (11) is an auxiliary constraint, responsible for ending the execution of orders at point i, but only once. Constraint (12) specifies the number of different type of resources (waiters). Constraint (13) is responsible for the binarity of selected decision variables.

$$\mathrm{Fc} = \min \frac{1}{\mathrm{I}} \sum_{\mathrm{i}=1}^{\mathrm{I}} \mathrm{Tkp_i} \tag{1}$$

$$\begin{aligned} &\mathrm{S_{i,j,k}} + \mathrm{tpo}^{\mathrm{CLP}}_{\mathrm{j,k}} = \mathrm{Tkp}_i \quad \text{for i} = 1\ldots\mathrm{I},\ \mathrm{j} = 1\ldots\mathrm{J},\ \mathrm{z_{i,j}} > 0, \mathrm{r}^{\mathrm{CLP}}_{\mathrm{j,k}} > 0 \\ &\mathrm{S_{i,j,k}} = 0 \quad \text{for i} = 1\ldots\mathrm{I},\ \mathrm{j} = 1\ldots\mathrm{J},\ \mathrm{z_{i,j}} = 0 \end{aligned} \tag{2}$$

$$\mathrm{S_{i,j,k}} \geq \mathrm{Ts} \quad \text{for i} = 1\ldots\mathrm{I},\ \mathrm{j} = 1\ldots\mathrm{J},\ \mathrm{k} = 1\ldots\mathrm{K} \tag{3}$$

Table 1 Summary indices, parameters and decision variables

Sets	
Set of points	I
Set of orders	J
Set of resources	K
Number of periods	G
Number of periods in which orders can be entered	H
Indices	
Points	i = 1…I
Orders	j = 1…J
Resources	k = 1…K
Period	g = 1…G
Period in which orders can be entered	h = 1…H
Decision variables	
Number of periods *g* delivery of all orders for point *i*	Tkp_i
The number of period *g* in which resource *k* can be used for order *j* at point *i*	$S_{i,j,k}$
If the execution of order *j* for point *i* uses resource *k* in period *g* then $X_{i,j,k,g} = 1$, otherwise $X_{i,j,k,g} = 0$	$X_{i,j,k,g}$
If *g* is the last period in which resource *k* is used in the execution of order *j* for point *i* then $Y_{i,j,k,g} = 1$, otherwise $Y_{i,j,k,g} = 0$	$Y_{i,j,k,g}$
If *g* is the last period in which orders are executed for point *i* then $W_{i,g} = 1$, otherwise $W_{i,g} = 0$	$W_{i,g}$
Parameters	
Calculated number of period *g* for the start of demand for resource *k* and order *j* (*CLP*)	$tpo_{j,k}^{CLP}$
Calculated number of period *g* for the end of demand for resource *k* and order *j* (*CLP*)	$tko_{j,k}^{CLP}$
Number of *k* resources needed for execution of order *j*	$r_{j,k}^{CLP}$
Number used to convert periods to moments (for connecting index *g* with variable $S_{i,j,k}$, if $S_{i,j,k} = 7$ then index $g = 7$) (*CLP*)	pp_g^{CLP}
The number of available resources *k* in the period *g*	$dkg_{k,g}$
The number of resources *k* needed (occupancy) in period *g* resulting from the execution of orders currently being processed	$dkgo_{k,g}$
The number of resources of the second type (i.e. packers, waiters) available during period *g*	rzg_g
The number of resources of the second type (i.e. packers, waiters) in period *g* resulting from the execution of orders currently being processed	$rzgo_g$
Calculated number of period *g* for the end of demand for resource *k* and order *j* (*CLP*)	Ts
Inputs	
The number of orders *j* at point *i*	$z_{i,j}$

$$\begin{aligned}&\sum_{i=1}^{I}\sum_{j=1}^{J}\left(X_{i,j,k,g}\cdot r^{CLP}_{j,k}\cdot z_{i,j}\right)\le dkg_{k,g}\quad \text{for } k=1\ldots K, g=1\ldots G\\&\sum_{i=1}^{I}\sum_{j=1}^{J}\left(X_{i,j,k,g}\cdot r^{CLP}_{j,k}\cdot z_{i,j}\right)= dkgo_{k,g}\quad \text{for } k=1\ldots K, g=1\ldots G\end{aligned}\tag{4}$$

$$\begin{aligned}&\sum_{g}^{G}X_{i,j,k,g}=tpo^{CLP}_{j,k}-tko^{CLP}_{j,k}\quad \text{for } i=1\ldots I, j=1\ldots J, k=1\ldots K, z_{i,j}>0, r^{CLP}_{j,k}>0\\&X_{i,j,k,g}=0\quad \text{for } i=1\ldots I, j=1\ldots J, k=1\ldots K, z_{i,j}=0\\&X_{i,j,k,g}=0\quad \text{for } i=1\ldots I, j=1\ldots J, k=1\ldots K, r^{CLP}_{j,k}=0\end{aligned}\tag{5}$$

$$\begin{aligned}&X_{i,j,k,g-1}-X_{i,j,k,g}\le Y_{i,j,k,g-1}\quad \text{for } i=i\ldots I, j=1\ldots J, k=1\ldots K, g=2\ldots G, z_{i,j}>0, r^{CLP}_{j,k}>0\\&Y_{i,j,k,g}=0\quad \text{for } i=1\ldots I, j=1\ldots J, k=1\ldots K, g=G\\&Y_{i,j,k,g}=0\quad \text{for } i=1\ldots I, j=1\ldots J, k=1\ldots K, g=1\end{aligned}\tag{6}$$

$$\sum_{g}^{G}Y_{i,j,k,g}=1\quad \text{for } i=1\ldots I, j=1\ldots J, k=1\ldots K, z_{a,j}>0, r^{CLP}_{j,k}>0\tag{7}$$

$$\begin{aligned}Y_{i,j,k1,g}=Y_{i,j,k2,g}\ &\text{for } i=1\ldots I, j=1\ldots J, k1, k2=1\ldots K, g\\&=1\ldots G,\ z_{i,j}>0, r^{CLP}_{j,k}>0,\ tko^{CLP}_{j,k1}=0, tko^{CLP}_{j,k2}=0,\end{aligned}\tag{8}$$

$$Tkp_i=\sum_{g=1}^{G}pp^{CLP}_{g}W_{i,g}\quad \text{for } i=1\ldots I\tag{9}$$

$$\begin{aligned}Y_{i,j,k,g-tko^{CLP}_{j,k}}&=W_{i,g}\quad \text{for } i=1\ldots I, j=1\ldots J, k=1\ldots K, g\\&=tko^{CLP}_{j,k}{}_{j,k}\ldots LG, z_{i,j}\ge 0, r^{CLP}_{j,k}\ge 0\\Y_{i,j,k,g}&=W_{i,g+tko^{CLP}_{j,k}}\quad \text{for } i=1\ldots I, j=1\ldots J, k=1\ldots K,\\g&=1\ldots G-tko^{CLP}_{j,k}, z_{i,j}\ge 0, r^{CLP}_{j,k}\ge 0\end{aligned}\tag{10}$$

$$\sum_{g=1}^{G}W_{i,g}\le 1\quad \text{for } i=1\ldots I\tag{11}$$

$$\begin{aligned}&\sum_{i=1}^{I}W_{i,g}\le rzg_g\quad \text{for } g=1\ldots G\\&\sum_{i=1}^{I}W_{i,g}=rzgo_g\quad \text{for } g=1\ldots G\end{aligned}\tag{12}$$

$$
\begin{aligned}
X_{i,j,k,g} &= \{0,1\} \quad \text{for } i = 1 \ldots I, j = 1 \ldots J, k = 1 \ldots K, g = 1 \ldots G \\
Y_{i,j,k,g} &= \{0,1\} \quad \text{for } i = 1 \ldots I, j = 1 \ldots J, k = 1 \ldots K, g = 1 \ldots : G \\
W_{i,g} &= \{0,1\} \quad \text{for } i = 1 \ldots I, g = 1 \ldots G
\end{aligned} \tag{13}
$$

3.3 A Integrated Declarative Approach to Modeling, Solving and Optimization

The idea of integrating CLP and MP environments outlined in Sect. 3 idea was expanded into the framework, as in Fig. 2. It can be used to implement specific decision models (Sect. 3.2) as well as ask questions related to these models. The questions and the additional constraints are implemented in the form of CLP

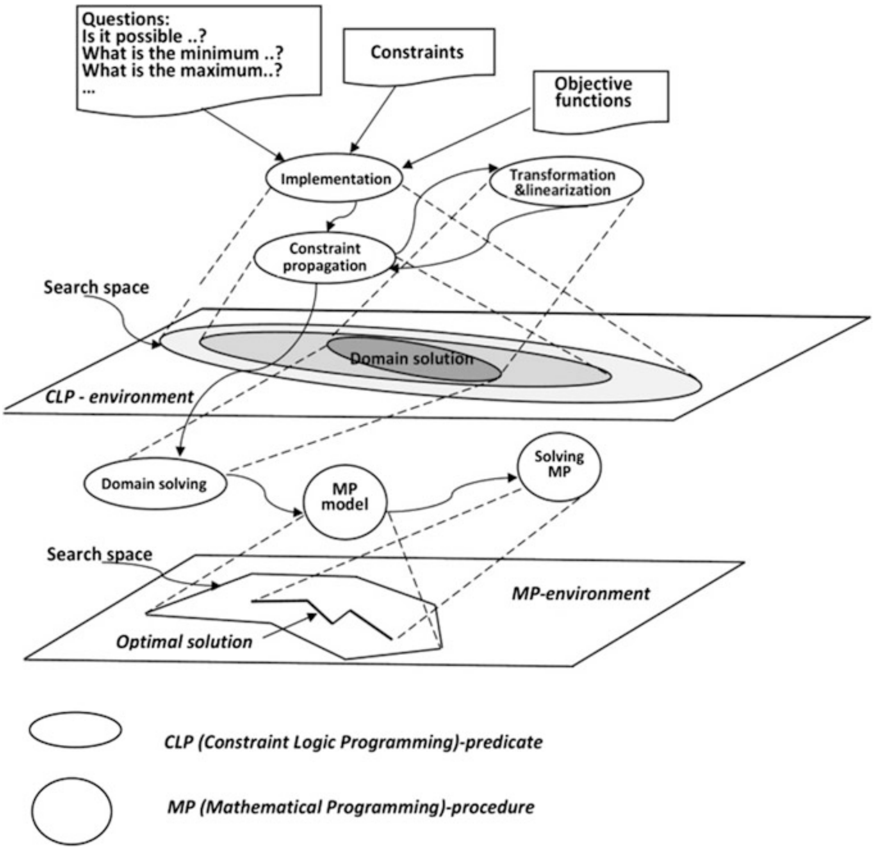

Fig. 2 The scheme of the implementation framework [8]

predicates. Examples of question that can be asked are presented in Sect. 3.1. The CLP environment is used to model the problem, linearize and transform it. It is also effective in terms of receiving answers to general questions, such as Is it possible…? Is it enough…?. For optimization and specific questions, such as What is the minimum/maximum…? What is the shortest time…?, after the initial "domain" solving, the problem is finally solved in the MP environment. In the case of decision problems (answers for general questions), obtained after transformation the linear model does not include the objective function. In this case, the MP environment is very fast in finding a feasible solution.

4 Numerical Experiments

In order to verify and evaluate the proposed approach, many numerical experiments were performed for the illustrative example. All the experiments relate to the system with ten points (i = 1…10), fifteen order types (j = 1…15), twelve resource types (k = 1…12), thirty time periods (g = 1…30) and seventeen orders zi, j. Computational experiments consisted in asking questions Q1…Q6 for the model (Sect. 3.2) implemented in the framework (Sect. 3.3) iteratively run with interactive algorithm. Orders are ordered in groups of h1 = 1, h2 = 5, h3 = 6, h4 = 12 periods. Table 2 shows the available number dk of each resource type k.

Figure 3a shows the implementation schedule of all orders for this example while minimizing the average waiting time-answer to question Q1 where all constraints are hard. The minimum waiting time for each order of period h1 is from Tpk = 5 to Tpk = 9, for the period h2 is from Tpk = 5 to Tpk = 11 for the period h3 is Tpk = 1 and for the period h4 is Tpk = 6. The makespan for all orders is 18 (Fig. 3a). Figure 4a shows the use of the resources (d6) needed to execute orders in the periods from g = 1 to g = 18. Figure 3b shows the implementation schedule-answer to question Q1 where the constraint (4) is soft constraint. It was allowed excess resources (k) by 50 %. In this case, the makespan for all orders is 17 (Fig. 3b). The minimum waiting time for each order of period h2 is from Tpk = 5 to Tpk = 9 (shorter) and for the period h4 is Tpk = 5 (shorter). Figure 4b shows the use of the resources (d6) needed to execute orders in the periods from g = 1 to g = 17.

The answer to question Q2 (Can the order be delivered within g period?) for ordering period h1 is NO for periods g = 4, g = 6, g = 9, but for period g = 11, it is YES. For the remaining periods h2, h3 and h4, the response can be read from the schedules (Figs. 3a, b).

Table 2 The number of different types of available resources

k	d_k	k	d_k	k	d_k	k	d_k
1	2	4	2	7	3	10	2
2	2	5	3	8	3	11	2
3	2	6	3	9	5	12	2

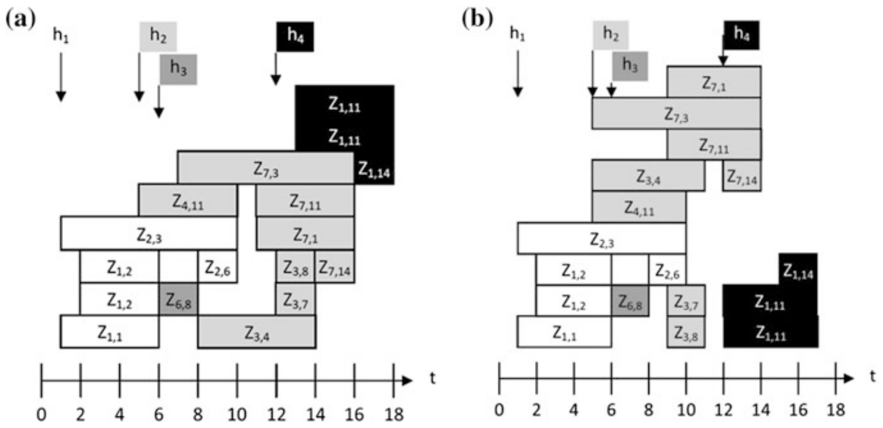

Fig. 3 **a** Gantt chart for illustrative example—hard constraints. **b** Gantt chart for illustrative example—hard and soft constraints

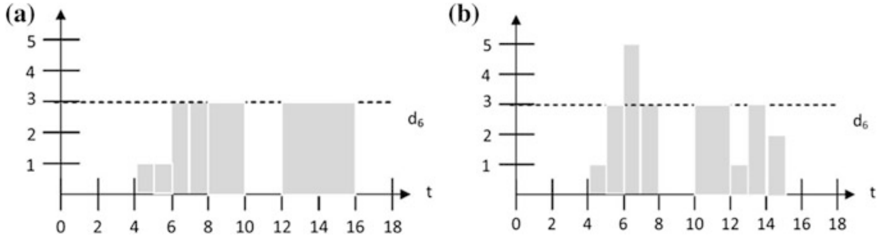

Fig. 4 **a** The use of the resource (d_6)—hard constraints. **b** The use of the resource (d_6)—hard and soft constraints

Table 3 A comparison of the effectiveness of the presented framework and mathematical programming (MP)

Question	V (V_{int})	C	T (s)
Q1 in MP, hard constraints	19,319 (18,940)	6,51,606	223
Q1 in framework, hard constraints	1134 (930)	83,299	31
Q1 in MP, hard & soft constraints	19,319 (18,940)	6,50,134	174
Q1 in framework, hard & soft constraints	1134 (930)	82,729	10

V variables, V_{int} integer variables, *C* constraints, *T* searching time

In the second stage of the research, effectiveness and efficiency of the proposed framework was evaluated in relation to the MP. For this purpose, question Q1 (with the largest computational requirements) and the model were implemented in both environments with hard and hard & soft constraints. The results are shown in Table 3.

5 Conclusion

In practice, such an approach to group order handling occurs in manufacturing, services, logistics and project management. The presented framework, which is an implementation of the proposed approach, enables effective planning, design and management of the processes by a manager. This allows the implementation of decision-making models with different objective functions and the introduction of the models already implemented with additional constraints. It also provides the opportunity to ask two types of questions and obtain answers. General questions may require domain solution, which in practice determines the availability of resources to execute orders. The wh-questions will in practice define the best, fastest, cheapest or the most expensive of the possible solutions. To obtain answers, optimization is necessary. The illustrative example shows only part of the framework's potential. Further work will consist in the implementation of more complex models such as a decision-making model covering operations. Introduction of precedence constraints to operations and orders, uncertainty, fuzzy logic [20] etc. is considered. New questions will be implemented to broaden the scope of decision support.

References

1. Ribas, I., Leisten, R., Framinan, J.M.: Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. Comput. Oper. Res. **37**, 1439–1454 (2010)
2. Tadayon, B., Salmasi, N.: A two-criteria objective function flexible flowshop scheduling problem with machine eligibility constraint. Int. J. Adv. Manufact. Technol. **64**(5–8), 1001–1015 (2013)
3. Apt, K., Wallace, M.: Constraint Logic Programming using Eclipse. Cambridge University Press, Cambridge (2006)
4. Sitek, P., Wikarek, J.: A Hybrid Approach to the Optimization of Multiechelon Systems Mathematical Problems in Engineering, Article ID 925675, Hindawi Publishing Corporation. doi:10.1155/2014/925675 (2014)
5. Sitek, P., Wikarek J.: A hybrid framework for the modelling and optimisation of decision problems in sustainable supply chain management. Int. J. Prod. Res. 1–18 doi:10.1080/00207543.2015.1005762 (2015)
6. Sitek, P., Wikarek, J.: Hybrid solution framework for supply chain problems. In: Distributed Computing and Artificial Intelligence (DCAI 2014), Book Series: Advances in Intelligent Systems and Computing, vol. 290, pp. 11–18 (2014)
7. Sitek, P., A hybrid CP/MP approach to supply chain modelling, optimization and analysis. In: Proceedings of the 2014 Federated Conference on Computer Science and Information Systems, pp. 1345–1352. doi:10.15439/2014F89 (2014)
8. Sitek, P., Wikarek, J., A novel approach to decision support and optimization of group job handling for multimodal processes in manufacturing and services. In: 15th IFAC/IEEE/IFIP/IFORS Symposium on Information Control Problems in Manufacturing, Ottawa, pp. 2183–2188 (2015)
9. Schrijver, A.: Theory of Linear and Integer Programming. Wiley, New York (1998)

10. Sawik, T.: Balancing and scheduling of surface mount technology lines. Int. J. Prod. Res. **40** (9), 1973–1991 (2002)
11. Carlier, J., Neron, E.: An exact method for solving the multi-processor flow-shop. RAIRO Oper. Res. **34**, 1–25 (2000)
12. Kurz, M.E., Askin, R.G.: Scheduling flexible flow lines with sequence-dependent setup times. Eur. J. Oper. Res. **159**(1), 66–82 (2004)
13. Jiang, J., Hsiao, W.C.: Mathematical programming for the scheduling problem with alternate process plans in FMS, Selected papers from the 16th annual conference on Computers and industrial engineering, pp. 15–18. Pergamon Press, Inc. Elmsford, NY, USA (1994)
14. Brah, S.A., Loo, L.L.: Heuristics for scheduling in a flow shop with multiple processors. Eur. J. Oper. Res. **113**(1), 113–122 (1999)
15. Bocewicz, G., Nielsen, I., Banaszak, Z.: Iterative multimodal processes scheduling. Ann. Rev. Control **38**(1), 113–132 (2014)
16. Rossi, F., Van Beek, P., Walsh, T.: Handbook of constraint programming. In Foundations of Artificial Intelligence. Elsevier Science Inc. New York, NY, USA (2006)
17. Buscemi, M.G., Montanari, U.: A survey of constraint-based programming paradigms. Comput. Sci. Rev. **2**(3), 137–141 (2008)
18. Milano, M., Wallace, M.: Integrating operations research in constraint programming. Ann. Oper. Res. **175**(1), 37–76 (2010)
19. Achterberg, T., Berthold, T., Koch, T., Wolter, K.: Constraint integer programming, a new approach to integrate CP and MIP. Lect. Notes Comput. Sci. **5015**, 6–20 (2008)
20. Relich, M., Muszynski, W.: The use of intelligent systems for planning and scheduling of product development projects. Procedia Comput. Sci. **35**, 1586–1595 (2014)

Lead Times and Order Sizes—A not so Simple Relationship

Peter Nielsen, Zbigniew Michna, Ngoc Anh Dung Do and Brian Bruhn Sørensen

Abstract Order quantity and lead time are two critical aspects of inventory management. In literature, lead time is considered as a constant or an uncontrollable stochastic variable. Theoretically, the order size has linear relationship with lead time meaning that order size increases when lead time increases. A data analysis from a Danish company reveals that the relationship between lead time and order size are not so simple. This paper focuses on analyzing the relationship between lead time and order size in order to answer two questions: (1) whether lead time is affected by order size or not; and (2) whether lead time and order size has linear relationship or not.

Keywords Inventory management · Lead time · Order quantity · Data analysis

1 Introduction

Lead time in inventory management is an undebated issue in manufacturing. Many studies are focused on how to order with lead time consideration. In traditional inventory problems, the lead time is assumed to be independent of the order

P. Nielsen (✉) · N.A.D. Do
Department of Mechanical and Manufacturing Engineering, Aalborg University, 9220 Aalborg, Denmark
e-mail: peter@m-tech.aau.dk

N.A.D. Do
e-mail: ngoc@m-tech.aau.dk

Z. Michna
Department of Mathematics and Cybernetics, Wroclaw University of Economics, Wroclaw, Poland
e-mail: zbigniew.michna@ue.wroc.pl

B.B. Sørensen
SCM Consultant, Copenhagen, Denmark
e-mail: BBS@B-B-S.DK

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_6

quantity [16]. The lead time is normally used to calculate both the re-order point (ROP) and the suitable order quantity. Many formulas are used to calculate ROP which assures there is no stock out during the lead time. The larger the lead time is, the higher the ROP is and the same for the order quantity. In some studies, the order quantity is assumed to be linear relation to lead time. The analysis on real data from a Danish company shows that there is a relation between lead time and order quantity but it is not a linear relation. This raises a question: is that true the lead time decides the order quantity? Or on another hand, the lead time affects to the order quantity or the order quantity affects to the lead time. This study will focus on analyzing the relationship between lead time and order quantity. The remainder of the paper is structured as follows. First a literature review is presented, followed by an analysis of actual lead time and order size relations. Finally conclusions and future research directions are presented.

2 Literature Review

Lead time and order quantity are two issues manufacturer's faced when controlling their supply chain. Many papers have been focused on how to determine the order quantity. Traditional inventory model determines the order quantity based on economic order quantity (EOQ) model and then, extends to the case with discount quantity [16]. Moon et al. [8] developed an EOQ model for ameliorating/deteriorating items under inflation and time discounting. Before that, Ray and Chaudhuri [13] worked on EOQ model with stock-dependent demand, shortage, inflation and time discounting. Then, Moon and Lee [9] considered the EOQ model with the effects of inflation and time-value of money. Chaudhry et al. [3] investigate on the problem to allocate order quantity among vendors. Dahel [4] develop a multi-objective mixed integer programming for the problem to select suppliers and order quantity allocation among these suppliers in volume discount environment. Jayaraman et al. [6] propose a comprehensive model for supplier selection and order quantity allocation. Nielsen et al. [11] find that order quantity and time are interdependent. These examples show that a lot of papers have been concentrated on determining order quantity in inventory management. However, in these studies, the authors normally omitted the lead time or assumed that lead time is constant. This assumption makes their studies less practical for real applications.

Some papers have included lead time in inventory management. Daya and Raouf [1] consider an inventory model where lead time is a decision variable. Chang and Chang [2] propose a mixed integer programming to solve the inventory problem with the consideration of variable lead time and price quantity discount. They assume that lead times can be reduced by adding extra crashing cost and this reduction will save the investment on safety stock. So and Zheng [15] consider the impact of lead time and forecast demand updating on order quantity variability.

Sirias and Mehra [14] implement a simulation approach for a two-echelon supply chain to compare two incentive systems: quantity discount versus lead time-dependent discount structures. Wang [17] evaluate the flexibility on order quantity and delivery lead time for a supply chain system. In these papers, lead time is considered as a factor which affects to the decision making on order quantity. Nielsen et al. [10] investigate the re-order point inventory management models sensitivity to demand distributions, demand dependencies and lead time distributions.

The above literature show few papers focus on the effect of lead time on order quantity. Likewise, few papers consider lead time, and typically treat lead time as an uncontrollable factor. No paper considers the effect of order quantity on lead time, so the relationship between lead times and order quantity is considered uni-directional—i.e. the order size depends on lead times, but lead times do not depend on order size. However, this seems counterintuitive as one would expect some sort of consequence from ordering large quantities. This paper analyzes the relationship between lead time and order quantity on both sides: the effect of lead time on order quantity and the effect of order quantity on lead time using data from a large Danish company.

3 Analysis

In the following a total of 78,009 orders for 25 different products from the same product family received by a manufacturing company over a period of three years are analyzed. The analysis contains a number of steps. First, χ^2 test for independence is used to determine if there is a link between order quantities and lead times. Following these t-tests are employed to investigate whether or not larger orders have larger average lead times than smaller orders. Following this the direct relationship between order sizes and lead times is investigated, where it is expected from theory that there is linear relationship. In Michna et al. [7], the authors propose that under stochastic lead times customers adapt order policies to reflect their current estimate of lead times. This seems intuitively appealing as one in this case would expect that if a customer is observing increasing lead times on average over a number of periods, the supplier will receive increasing order sizes with some time lag. An exact estimate of the time lag and the number of observations used to estimate lead times cannot be found from the data, so a number of combinations are tested. Having investigated this, the next point of interest is if the lead time distributions for different order sizes, but for the same product, exhibit the same type of behavior—i.e. have roughly the same shape. Figure 1 shows an overview of the analysis approach employed in the paper.

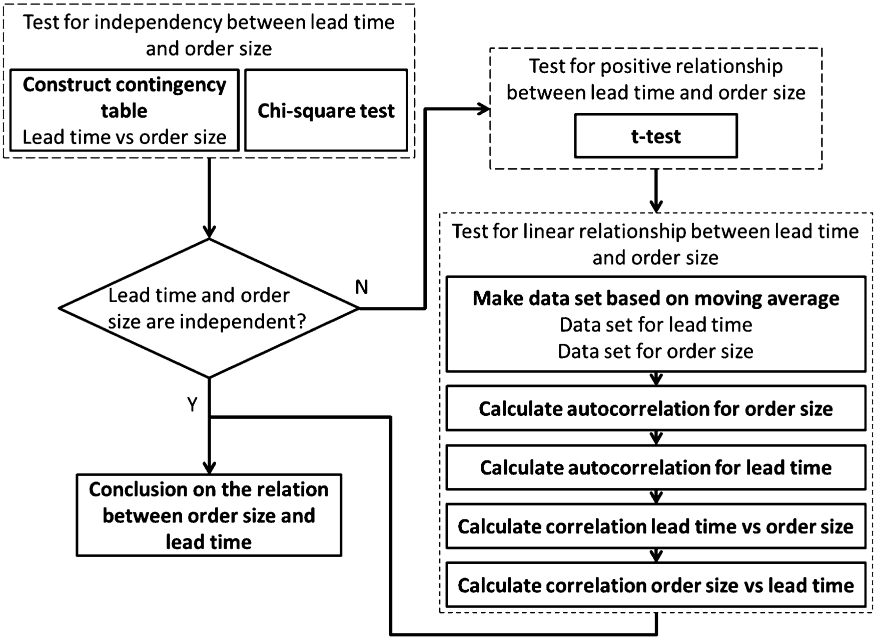

Fig. 1 Procedure for the analysis approach

For the following analysis some definitions are necessary:

Q_p^{1st}	The 1st quartile order size
Q_p^{3rd}	The 3rd quartile order size
$Q_{i,p}^{1st}$	The set of i orders for the pth product less than or equal to Q_p^{1st}
$Q_{j,p}^{3rd}$	The set of j orders for the pth product greater or equal $Q_{j,p}^{3rd}$
$LT_{i,p}^{1st}$	The lead time for the ith order for the pth product for the orders in $Q_{i,p}^{1st}$
$LT_{j,p}^{3rd}$	The lead time for the jth order for the pth product for the orders in $Q_{j,p}^{3rd}$
$\overline{LT}_p^{1st}$	The mean lead time for the orders contained in $Q_{i,p}^{1st}$
$\overline{LT}_p^{3rd}$	The mean lead time for the orders contained in $Q_{j,p}^{3rd}$

For the χ^2 test one would expect the same number of observations in all the four fields in the contingency table shown in Table 1.

Where lead times less than the 1st quartile are categorized as small and larger than the 3rd quartile are categorized as large and the same for order quantities.

From Table 2 it is clear that remarkably few of the tested products have lead time distributions were lead times and order sizes can be considered independent when using a χ^2 test for independence. Only three out of 25 products' lead time

Table 1 Contingency table concept

	Small orders	Large orders
Small lead times	No. observations	No. observations
Large lead times	No. observations	No. observations

Table 2 Overview of the p-values from

	p-Values from χ^2-test	p-Values from t-test	LT relationship
P1	0.000	0.000	2.1
P2	0.000	0.000	2.9
P3	0.000	0.000	1.8
P4	0.000	0.000	2.7
P5	0.000	0.000	2.0
P6	0.000	0.000	1.7
P7	0.000	0.000	1.4
P8	**0.329**	0.000	1.5
P9	0.000	0.000	1.7
P10	**0.332**	0.002	1.2
P11	0.000	0.000	1.6
P12	0.000	0.000	1.9
P13	0.000	0.000	1.7
P14	0.000	0.000	1.7
P15	0.000	0.000	2.5
P16	0.000	0.000	1.8
P17	**0.899**	**0.093**	1.1
P18	0.000	0.000	1.9
P19	0.000	0.000	1.8
P20	0.000	0.000	3.5
P21	0.000	**0.001**	1.2
P22	0.000	0.000	1.7
P23	0.000	**0.107**	1.2
P24	**0.008**	0.000	1.5
P25	0.000	0.000	2.1

distributions can be considered strictly independent—with one further with a p-value just below 0.01. So the conclusion supports the expectations that lead times and order sizes are related. The next step is now to determine whether there is a simple linear relationship (as tested by Pearson correlation) between the order sizes and the lead times as expected from theory.

The conclusion from the Pearson correlation tests is quite simple as seen in Fig. 2 where the right-hand axis is the number of observations. There is a significant positive linear correlation between order sizes and lead times for most products, but

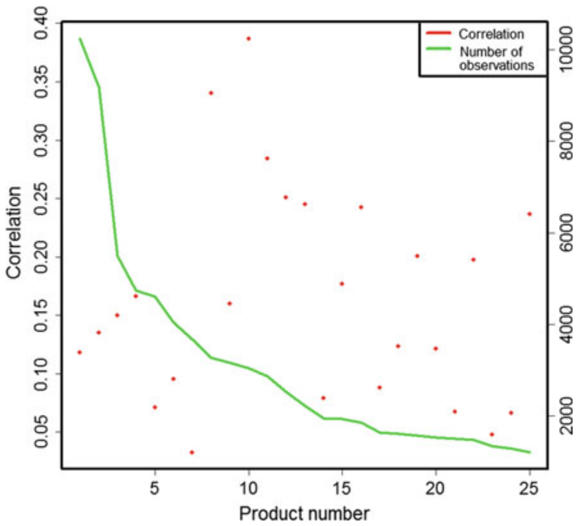

Fig. 2 Product number, number of observations and Pearson correlation

the strength of the correlation is so weak that it for the vast majority of the products has almost no explanatory power. It is also worth noting that the correlations are all positive as one would expect from theory. This is intuitive appealing as it implies that larger orders have larger lead times. Although this is not considered in either traditional inventory or supply chain management literature it seems that it (at least for this particular case) is an oversimplification of the connection between lead times and order sizes. The same oversimplification that can be said to be present when assuming that lead times are deterministic rather than stochastic in nature [12]. While there is a link between order sizes and lead times, it is not an unambiguous one. The strength of the link is too weak to be used to predict to any degree of certainty the lead time for any given order size. The reasons for this can be very varied, but a simple conclusion could be that the behavior is in fact highly stochastic in nature and stems from a number of order size dependent distributions.

This leads to the next step where the same quartiles of lead time distributions investigated in the χ^2 test are tested for similar mean using the student's t-test, with the null hypothesis that the lead time distributions $(LT_{i,p}^{1st}; LT_{j,p}^{3rd})$ have the same mean. In all but two cases the null hypothesis is rejected on a better than 0.01 level and in all but three cases it is rejected on the 0.001 level. The overwhelming conclusion from the analysis is that order sizes and lead times are dependent and that the lead time distributions for small orders have a significantly different mean than the mean lead times for large orders. The right hand column in Table 2 shows the $\frac{\overline{LT}_p^{3rd}}{\overline{LT}_p^{1st}}$ ratio for all products p. From this it is clear that the lead times for the orders contained in $Q_{j,p}^{3rd}$ are uniformly larger on average than the lead times for the orders contained in $Q_{i,p}^{1st}$ for all p. This ratio is on average 1.8 meaning that the lead times for $Q_{i,p}^{1st}$ on average are 1.8 times larger than the lead times for $Q_{j,p}^{3rd}$.

From literature it would be expected that if a customer observers increasing lead times (LT), the customer would place progressively larger orders (Q). In practice one would expect that a time series of lead times would be more strongly correlated with the observed order sizes if lagged. In fact one would suspect that the lagged relationship would have a stronger correlation than the unlagged one. If this is not the case one would suspect that the customers are not updating their lead time information and solely change order sizes to reflect changes in demand (or changes in e.g. quantity discount structures). Following this the next conclusion must be that order sizes and lead time should be strictly independent.

Let us assume that a customer uses a certain amount of past observations to estimate the following set of order sizes using e.g. a moving average. Strictly speaking we have no ability to guess how the customer estimates lead times, thus we can only test moving averages with different orders of smoothing and see if a pattern arises. Specifically one would assume that if the customers observe high lead times they will increase order sizes symmetrically that if they observe short lead times they will reduce order sizes.

The 25 selected products are compared using a centered moving average for 40 observations and Q is lagged with respect to LT with 0, 10, 20, 40, 60 and 80 observations. In this case it follows that both the lead time series and the order size series will exhibit strong autocorrelation. However, the true benchmark must be if the lagged observations are stronger correlated than the non-lagged observations—i.e. is the customers using historic observations of lead times to update their order policy? An example of an auto and cross correlation plot is seen in Fig. 3.

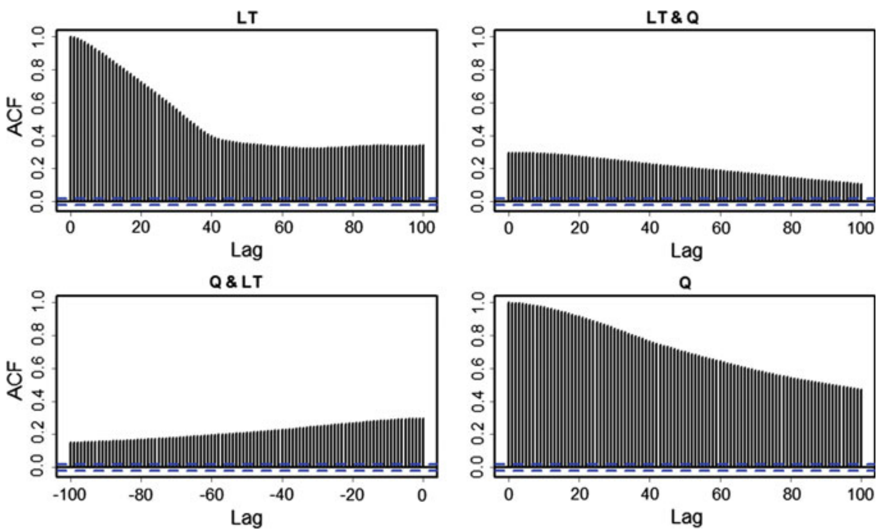

Fig. 3 An example of auto and cross correlation plots for product #1 using a 40 period moving average (MA)

As expected both the order size and the lead time by themselves exhibit strong autocorrelation, with the order size appearing to be more stable than the lead times. It is equally obvious that the strength in correlation between Q and LT is decreasing when Q is lagged compared to LT.

Figure 4 shows the strength of the correlation for the 25 products used in the study. The interpretation of the results depicted in Fig. 4 is not completely unambiguous. If changing lead times (increasing/decreasing) followed theory one would expect that on average the correlation is stronger (and positive) when orders sizes are lagged compared to LT. However, only 7 out of 25 products have this characteristic when the maximum lag of 80 is observed and only 5 out of 25 when lag 60 is considered. It is also worth noting that two of the products' lead times actually are negatively correlated when the moving averages. This makes very little sense as it implies that $LT \downarrow \rightarrow Q \uparrow$ which is counterintuitive unless the customers are not updating their lead time information and basing order sizes on other calculation methods. If order size calculation methods that do not use expected lead times are used, one would expect that lead times and order sizes in general should be independent. Based on this analysis it is tempting to conclude that not only does $LT \uparrow \rightarrow Q \uparrow$, but also $Q \uparrow \rightarrow LT \uparrow$. However, without knowing the demand faced by the customers there is no method for unequivocally concluding that lead times depend on order size.

To determine whether this simplified model could be relevant the next step is to establish whether the $LT^{1st}_{i,p}$ and $LT^{3rd}_{j,p}$ distributions exhibit similar behavior. Nielsen et al. [12] using partly the same dataset already established that the individual lead time distributions for small intervals can be considered i.i.d. This conclusion must now be somewhat modified to include the fact that within a given interval of order sizes the corresponding lead time distribution can be considered to be i.i.d. However, before reaching this conclusion it seems prudent to examine the characteristics of the lead time distributions. Are they in fact similar in shape, but have different means—i.e. is it possible to use a scaled version of the lead time distribution for all order sizes? When describing the characteristics of a distribution, normally one refers to the second, third and fourth moments of the distribution (variance, skewness and kurtosis respectively). Of these variance gives the largest

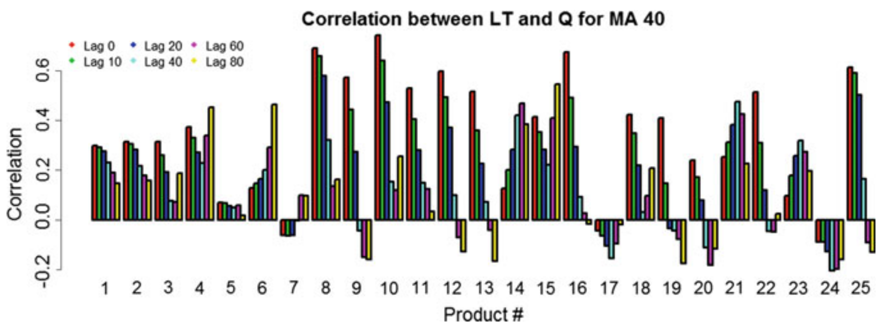

Fig. 4 Correlation between lead time and order size for all 25 products

problem in comparisons as both skewness and kurtosis are unitless. To achieve a unitless measure for comparison one typically uses the Coefficient of Variance (CV) which is simply the sample standard deviation of the distribution divided by the sample mean of the distribution. These are plotted for comparison purposes for all 25 investigated products for $LT_{i,p}^{1st}$ and $LT_{j,p}^{3rd}$ respectively. For all the characteristics (CV, skewness and kurtosis) the lower the values the less volatile the distribution is considered.

Again from the Figs. 5 and 6, it is possible to establish that for the vast majority of the products p there is a clear difference between the two distributions $LT_{i,p}^{1st}$ and

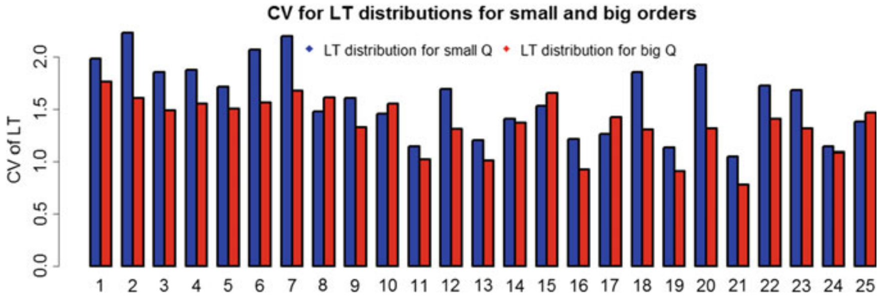

Fig. 5 Covariance of lead time distributions for small and big orders

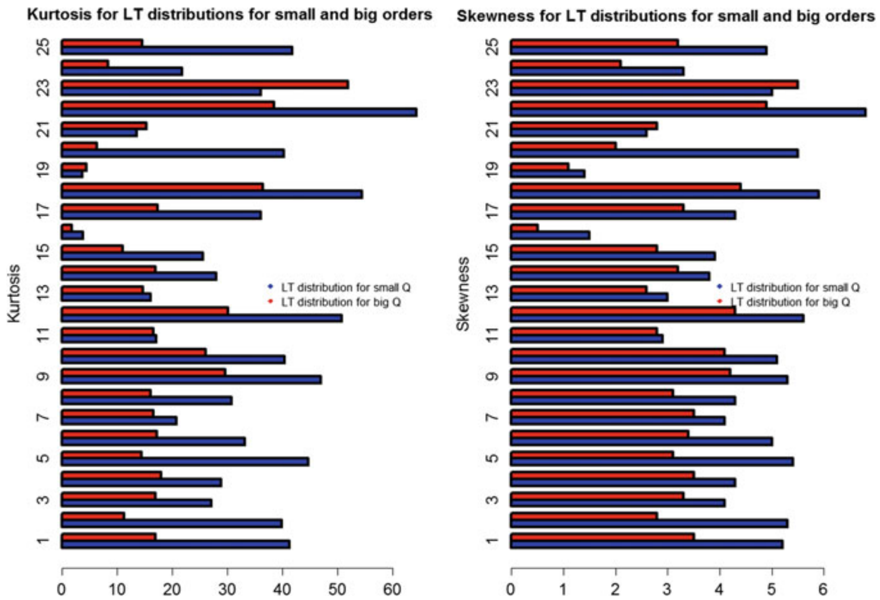

Fig. 6 Kurtosis (*left hand graph*) and skewness (*right hand graph*) of lead time distributions for small and big orders

$LT^{3rd}_{j,p}$. Specifically; in 20 out of 25 cases the CV is larger for $LT^{1st}_{i,p}$ than for $LT^{3rd}_{j,p}$, for 22 out of 25 the kurtosis is larger for $LT^{1st}_{i,p}$ than for $LT^{3rd}_{j,p}$ and for 23 out of 25 the skewness is larger for $LT^{1st}_{i,p}$ than for $LT^{3rd}_{j,p}$. So while the lead time on average is 1.8 times higher for $LT^{3rd}_{j,p}$ than for $LT^{1st}_{i,p}$ the $LT^{1st}_{i,p}$ distributions are on average more peaked and at the same time more skewed.

This is very interesting because from Do et al. [5] we know that the more extreme distributions have practical implications for the management of supply chains. More skewed lead time distributions results in longer time to reach a steady state of a supply chain. It is also worth noting that all the lead time distributions are positively (right skewed).

4 Conclusions and Future Research

This paper has focused on empirically establishing the link between order sizes and lead times. The methods main limitation in practice is the need for many observations. In most practical situations one can question whether it will be possible to have sufficient observations to estimate the lead time distributions for suitable intervals of order sizes. The implications for both inventory and supply chain management are significant. Current state of within the fields assumes that order sizes and lead times are strictly independent (with the typical added assumption that lead times are deterministic [12]. This is obviously an oversimplification of the relations between these variables. In traditional inventory management the following relationship between order sizes (Q), demand rate ($\widetilde{D}$) and average lead time ($\widetilde{LT}$) are assumed to hold true when demand and lead times are i.i.d. (or deterministic).

$$Q = k \times \widetilde{D} \times \widetilde{LT} \quad \text{where } k \text{ is a constant}$$

With this in mind the complexity that arises when Q and LT are in fact bidirectionally dependent is obvious. Increasing order sizes lead to increasing lead times and increasing lead times lead to increasing order sizes. From the analysis it is possible to conclude that order sizes and lead times are dependently distributed, but that the relationship is not strictly linear. It is also possible to conclude that increasing/decreasing lead times do not to a significant degree influence changes in order sizes. It is a catch 22 unless a suitable methodology for lead time distributions dependent on order size can be found and used. Regardless of this the results underline that current state is based on over simplistic models of lead times and one can only imagine how this impacts inventory and supply chain management in practice.

The contributions of this study are twofold. First, it shows that the assumption that the order size and lead time have a linear relation cannot be applied in any general case. This means that the relationship between order size and lead time should be investigated in inventory management. Second, this study proposes a simple method to analyze the dependence of order size and lead time. Future work will focus on how order size dependent lead time distributions influence supply chains and the bullwhip effect.

Acknowledgements The presented research is partly supported by the Polish National Centre of Science under the grant UMO-2012/07/B/HS4/00702.

References

1. Ben-Daya, M., Raouf, A.: Inventory models involving lead time as a decision variable. J. Oper. Res. Soc. **45**, 579–582 (1994)
2. Chang, C.T., Chang, S.C.: On the inventory model with variable lead time and price-quantity discount. J. Oper. Res. Soc. **52**, 1151–1158 (2001)
3. Chaudhry, S.S., Forst, F.G., Zydiak, J.L.: A multicriteria approach to allocating order quantity among. Prod. Inventory Manage. J. **32**(3), 82–86 (1991)
4. Dahel, N.E.: Vendor selection and order quantity allocation in volume discount environments. Supply Chain Manage.: Int. J. **8**(4), 335–342 (2003)
5. Do, N.A.D., Nielsen, P., Michna, Z., Nielsen, I.: Quantifying the bullwhip effect of multi-echelon system with stochastic dependent lead time, advances in production management systems. In: Innovative and Knowledge-Based Production Management in a Global-Local World, pp. 419–426 (2014)
6. Jayaraman, V., Srivastava, R., Benton, W.C.: Supplier selection and order quantity allocation: a comprehensive model. J. Supply Chain Manage. **35**(1), 50–58 (1999)
7. Michna, Z., Nielsen, I., Nielsen, P.: The bullwhip effect in supply chains with stochastic lead times. Math. Econ. **9**(16), 71–88 (2013)
8. Moon, I., Giri, B.C., Ko, B.: Economic order quantity models for ameliorating/deteriorating items under inflation and time discounting. Eur. J. Oper. Res. **162**(3), 773–785 (2005)
9. Moon, I., Lee, S.: The effects of inflation and time-value of money on an economic order quantity model with a random product life cycle. Eur. J. Oper. Res. **125**(3), 588–601 (2000)
10. Nielsen, P., Davoli, G., Nielsen, I., Rytter, N.G.: A design of experiments approach to investigating the sensitivity of the re-order point method. In: Proceedings of the International Conference on Advances in Production Management Systems 2012, Rhodes, Greece, 2012
11. Nielsen, P., Nielsen, I., Steger-Jensen, K.: Analyzing and evaluating product demand interdependencies. Comput. Ind. **61**, 869–876 (2010)
12. Nielsen, P., Michna, Z., Do, N.A.D.: An empirical investigation of lead time distributions, advances in production management systems. In: Innovative and Knowledge-Based Production Management in a Global-Local World, pp. 435–442 (2014)
13. Ray, J., Chaudhuri, K.S.: An EOQ model with stock-dependent demand, shortage, inflation and time discounting. Int. J. Prod. Econ. **53**(2), 171–180 (1997)
14. Sirias, D., Mehra, S.: Quantity discount versus lead time-dependent discount in an inter-organizational supply chain. Int. J. Prod. Res. **43**(16), 3481–3496 (2005)
15. So, K.C., Zheng, X.: Impact of supplier's lead time and forecast demand updating on retailer's order quantity variability in a two-level supply chain. Int. J. Prod. Econ. **86**(2), 169–179 (2003)
16. Stevenson, W.J.: Operations Management, 9th edn. McGraw-Hill, Boston (2007)
17. Wang, Y.C.: Evaluating flexibility on order quantity and delivery lead time for a supply chain system. Int. J. Syst. Sci. **39**(12), 1193–1202 (2008)

Part III
Systems Analysis and Modeling

Decisional DNA Based Conceptual Framework for Smart Manufacturing

Syed Imran Shafiq, Cesar Sanin, Edward Szczerbicki and Carlos Toro

Abstract This paper presents the conceptual framework for systematic knowledge representation, storage and reuse of manufacturing information in a production scenario. This knowledge structure is designed for three levels in a manufacturing set up viz. first at the engineering objects level, second at process and finally at factory level. Virtual engineering object (VEO) deals with knowledge at the individual object/component/machine level while Virtual engineering process (VEP) represents knowledge at the process/operations level. Implementation of VEO and VEP has been already been done. This article proposes the integrated concept and architecture at facility/factory level and we termed it as Virtual Engineering Factory (VEF). It provides access to the complete production history of the factory, which is useful for decision-making activities. Moreover, we propose combined architecture for the extraction of the knowledge from different levels of manufacturing through VEF, VEP and VEO.

Keywords Decisional DNA · Set of experience knowledge structure · Virtual engineering objects

S.I. Shafiq (✉) · C. Sanin
The University of Newcastle, University Drive, Callaghan, NSW 2308, Australia
e-mail: syedimran.shafiq@uon.edu.au

C. Sanin
e-mail: Cesar.Maldonadosanin@uon.edu.au

E. Szczerbicki
Gdansk University of Technology, Gdansk, Poland
e-mail: Edward.Szczerbicki@newcastle.edu.au

C. Toro
Vicomtech-IK4, San Sebastian, Spain
e-mail: ctoro@vicomtech.org

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_7

1 Introduction

Efforts are being made around the world to improve the productivity and efficiency in industrial manufacturing which can be achieved by integrating manufacturing with information and communication technology. The main objective behind this integration is to reap the benefits of the unprecedented advancement in the field of information and communication technologies [1, 2].

These ideas lead to the emergence of the new concept of Virtual Engineering Factory (VEF). The goal is the intelligent factory, which is characterized by adaptability, resource efficiency and the integration of customers and business partners in business and value processes.

Knowledge engineering plays an important role in virtual engineering systems as there is a need for a unified framework to represent the myriad types of data and application contexts in different physical domains, and interpret them under the appropriate contexts [3]. The concept of virtual systems has three broad levels of Virtual engineering object (VEO), Virtual engineering process (VEP) and Virtual Engineering Factory (VEF) as shown in Fig. 1. The concept of VEO and VEP has been developed and tested. This article not only focuses on the concept, architecture of VEF and also proposes the mechanism of interlinking the three levels.

The structure of this paper is as follows: Sect. 2 gives the overview and the central ides of the work. Moreover it deals with the basic concepts architecture and objectives of VEO, VEP and VEF. Section 3 discusses the integration of VEO, VEP and also gives framework to extract experience to reuse it for decision making. In Sect. 4 potential benefits of this work and conclusions are presented.

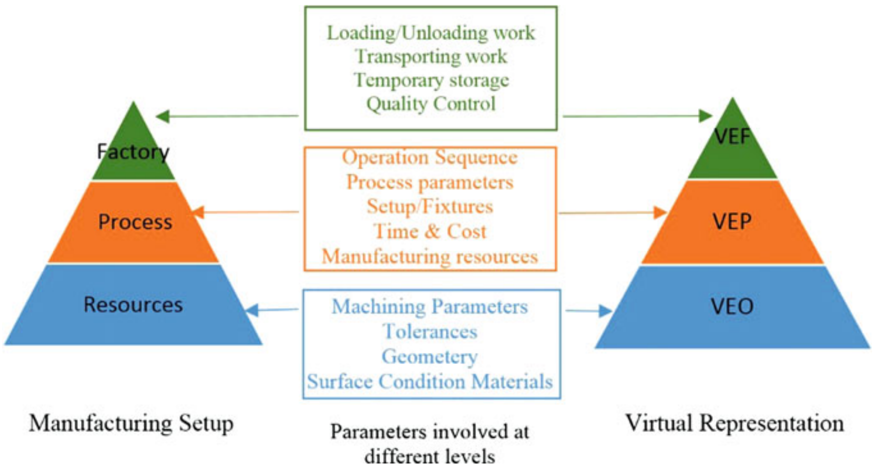

Fig. 1 Correlation of physical and the virtual world

2 VEO/VEP/VEF: Fusion of the Physical and Virtual World

The central idea of this work is to replicate the knowledge and experience of the manufacturing factory and to represent it virtually. In other words mechanism to store and reuse experience related with objects, process and factory working is developed. Physical manufacturing scenario can be classified into three groups/levels resources, process and factory. Knowledge representation of these levels is developed both at individual level and in conjunction with each other.

As depicted in Fig. 1. VEO is the representation at the individual object/resource/artefact level and is capable of representing all the information like machine level like machining parameters, tolerances, surface conditions etc.

VEP deals with the information at the process or shop floor level, like operation sequences, process parameters, time cost etc. While the VEF represents knowledge at the factory level encompassing or having links to VEO and VEP.

VEF stores the information and experience/formal decisions related with the various different aspects involved at the systems level like material handling, storage quality control, transportation etc.

Set of experience knowledge structure (SOEKS) and Decisional DNA (DDNA) a powerful knowledge representation technique is used as the technological base for this work. SOEKS-DDNA [4–6] is proposed as a unique and single structure for capturing, storing, improving and reusing decisional experience. Its name is a metaphor related to human DNA, and the way it transmits genetic information among individuals through time. Based on the literature review [7] it is evident that SOEKS-DDNA is a novel technique to reuse the experience and the formal decisions made in day-to-day activities. It can be implemented on various platforms (e.g., ontology, reflexive ontology, software based, fuzzy logic, etc.) in multi-domains, which makes it a general and universal approach.

Detail description of VEO, VEP and VEF is presented in the following sections.

2.1 Virtual Engineering Object (VEO)

A VEO [7–13] is knowledge representation of an engineering artefact, it has three features:

(i) the embedding of the decisional model expressed by the set of experience, (ii) a geometric representation, and (iii) the necessary means to relate such virtualization with the physical object being represented.

A VEO is a living representation of an object capable of capturing, adding, storing, improving, sharing and reusing knowledge through experience, in a way similar to an expert in that object. A VEO can encapsulate knowledge and experience of every important feature related with an engineering object. This can be achieved by gathering information from six different aspects (chromosomes) of an

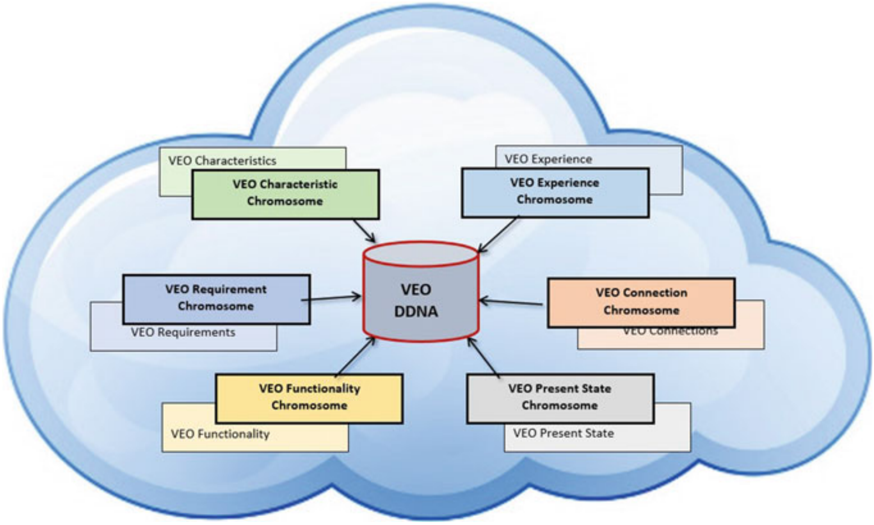

Fig. 2 VEO structure [7, 13]

object viz. Characteristics, Functionality, Requirements, Connections, Present State and Experience as illustrated in Fig. 2.

VEO is developed on the concept of cradle-to-grave approach, which means that the contextual information and decision making regarding an engineering object right for its inception until its useful life is stored or linked in it. The knowledge representation technique of Set of experience knowledge structure (SOEKS)-Decisional DNA (DDNA) is used for developing VEO as it provides dynamicity to overcome issues of representing complex and discrete objects.

The changing machining conditions such as spindle thermal deformation, tool failure, chatter, and work piece deformation induced by clamping force, cutting force, and material inner stress have significant impacts on machining quality and efficiency. Figure 1 exhibits that VEO will cater decision making regarding problems which may emerge during the machining process due to complex conditions at the machining level.

2.2 *Virtual Engineering Process (VEP)*

In manufacturing environment collection of components/tools/objects constitute a process and combination of process constitutes a system as depicted in Fig. 1. Following this pattern, virtual representation of artefacts in the form of VEO has already been achieved as discussed in Sect. 2.1.

Virtual engineering process (VEP) [14] is a knowledge representation of manufacturing process/process planning of artefact having all shop floor level

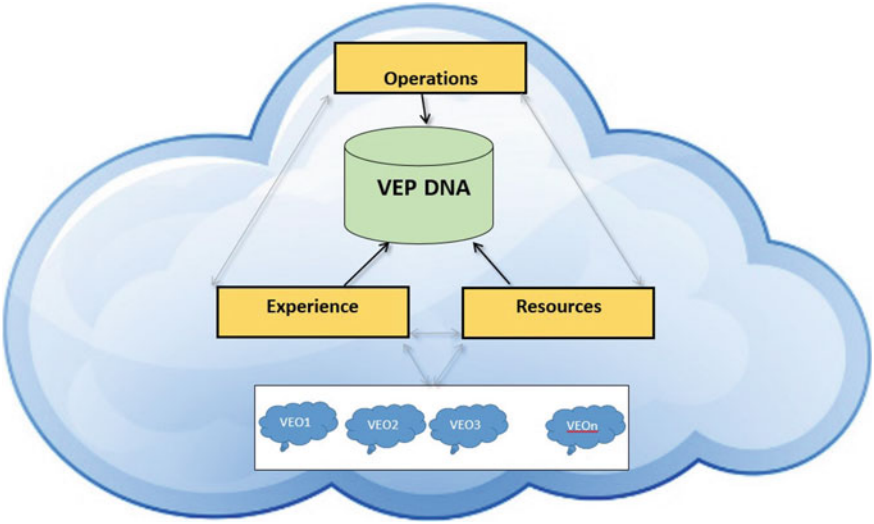

Fig. 3 VEP architecture [14]

information regarding operations required; their sequence and resources needed to manufacture it as shown in Fig. 3. VEP deals with the selection of necessary manufacturing operations and determination of their sequences, as well as the selection of manufacturing resources to "transform" a design model into a physical component economically and competitively.

Process planning is combination of information regarding the operation required, manufacturing sequence, and machines required. In addition to this, for VEP, information of all the VEO's of the resource associated with the process is also required. Therefore to encapsulate knowledge of the above mentioned areas the VEP is designed having following three main elements or modules:

(i) Operations: In this module of VEP all the information related with the operations that are required to manufacture an engineering object is stored. This includes knowledge in the form of SOEKS related to operation process and scheduling. Furthermore functional dependencies between operations are also part of operations. These are sub categorized and there interaction planning functions are given below:

- Scheduling route—based on global and local geometry.
- Processes—process capabilities, process cost.
- Process parameters—tolerance, surface finish, size, material type, quantity, urgency

(ii) Resources: Information based on the past experience about resources used to manufacture a component mentioned in operations module of VEP is stored here. The knowledge of the machine level stored in this section is as follows:

- Machine and tool selections—machine availability, cost machine capability, size, length, cut length, shank length, holder, materials, geometry, roughing and finishing
- Fixture selection—fixture element function, locating, supporting, clamping surfaces, stability

Furthermore as discussed in Sect. 2.1 the information of VEO categorized under characteristics, requirements, functionality, present state, connections and experience is also linked in this section.

(iii) Experience: In the experience module, links to the SOEKS of VEO's along with VEP having past formal decisions to manufacture engineering components are stored. They represent the links to SOE's based on past experience on that particular machine to perform given operation along with operational and routing parameter.

2.3 *Virtual Engineering Factory (VEF)*

In this section extension of VEO/VEP to the factory level is discussed and an integrated architecture covering all the aspect of a manufacturing unit is proposed. A manufacturing factory is collection of integrated equipment and human resources, whose function is to perform one or more processing and/or assembly operations on a starting raw material, part, or set of parts

The main components of a manufacturing system as identified in the literature can be broadly classified as:

- Production machines and tools
- Material handling and work positioning devices
- Computer systems
- Human resources are required either full-time or periodically to keep the system running

Based on the components and their functionality at the factory level, architecture of VEF is conceived. VEF have six elements each having links to involved VEP and VEO to represent the entire knowledge and experience of a manufacturing unit/factory. The arrangement of these six elements of VEF along VEO and VEP in cloud architecture is shown in Fig. 4. Elements of VEF are:

- VEF Loading/Unloading: Information related with loading and unloading work units at each station along with positioning of work units at each station is stores in this module.
- VEF Transportation: Transporting work units between stations in multi-station systems. Work units may flow through the same sequence of workstations or are moved through a variety of different station sequences, knowledge related with these are stored in here.

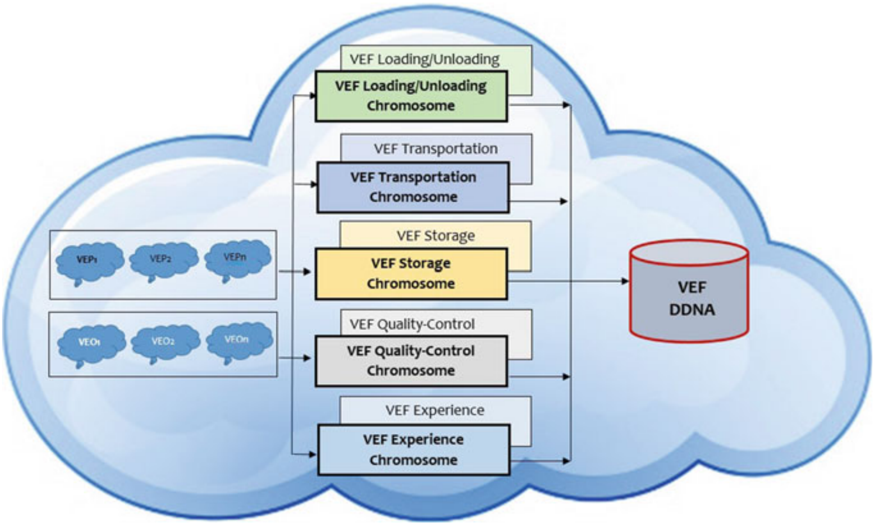

Fig. 4 VEF in a cloud architecture

- VEF Storage: This module of VEF will store all the knowledge related with the temporary storage of raw materials and work during the manufacturing process
- VEF Quality Control: What quality control strategy adopted, its implementation method and outcome is captured in this VEF module.
- VEF Experience: In this part of VEF the entire history of a formal decision taken at the factory level along with the links/information of the involved VEP and VEP in that decision are stored. In other word all the past experience is captured in this module.

Figure 5 illustrates the virtual representation of manufacturing unit having 4 machine machines, 3 automatic guided vehicles (AVGS's), loading/unloading

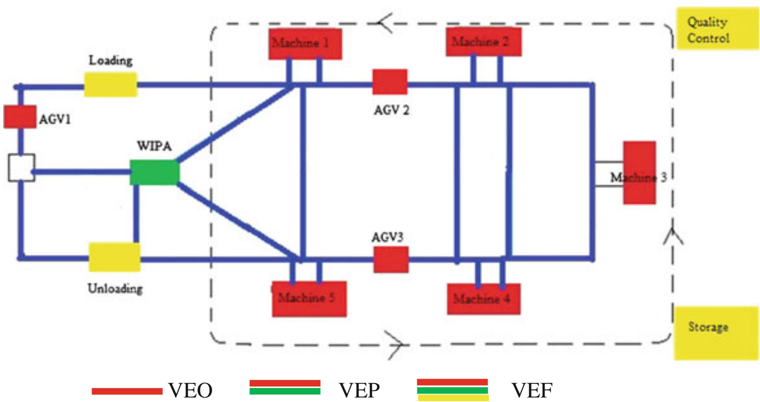

Fig. 5 Schematic diagram for knowledge representation of a virtual factory

stations, work in process assignment station, storage and quality control station. All the red stations are represented by the VEO architecture, green station (linked with involved red stations) represents the VEP design and the yellow station will have VEF model, encompassing VEO and VEO within itself.

3 Integration and Working Model for VEF, VEP and VEO

As discussed in the previous sections formal decisions are captured and stored in a structured manner at the object, process and system level. Figure 6 illustrates the flowchart after building of VEF SOEKS, VEF Chromosomes and VEF DNA. After

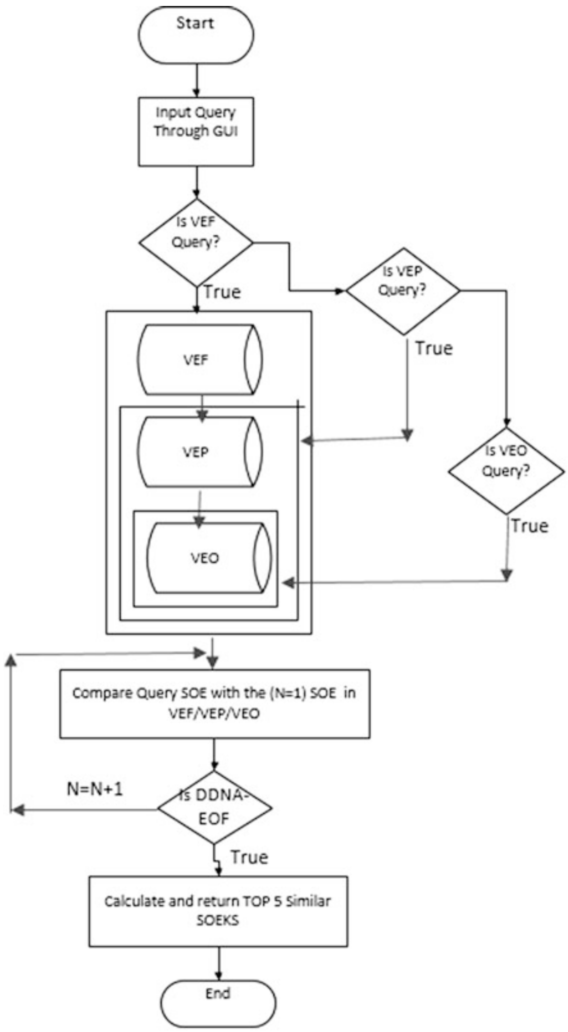

Fig. 6 Extracting knowledge from the experience at each level in a manufacturing setup

all the experiences in the batch are translated using the ideas described before, the inference process can executed to discover new rules according to the categories described above. Then, assuming the existence of more knowledge in the system, under specific conditions and after validation against other experiences, the inference process is able to determine that the values obtained.

When query is generated through a GUI, it is converted into a query SOEK programmatically. Depending upon whether it is related with the object, process or the factory level, program will keep on calculating the similarity of query SOEK with each of the SOEKS stored in the VEF-DDNA. Finally similarities calculated are sorted and the top 5 most similar SOEKS are returned.

4 Discussion

VEO provides a structure for parts involved in the manufacturing process to possess information on themselves and suitable means of communication. This VEO/VEP is to be embedded in the process as a whole and in extreme cases control not only their own logistical path through production, but rather the entire production workflow that concerns them.

VEO/VEP is to supply compressed information suitably derived from the complex interrelationships and communicated in a personalized manner as the basis for their intervention in the process. In this way, a new form of cooperation between machines and parts of machines arises. This will support both short term flexibility and medium-term transformability and thus improve the resilience of production.

Table 1 VEF features that can contribute in the design and implementation of a thinking factory

Key aspects	VEO/VEP/VEF features
Interoperability	Product self-awareness (history, status, location, delivery strategy and service)
	Throughout linking product virtual model and situational physical status
	Resource/energy efficiency and sustainable production
Virtualization	Empowering end users in the final product configuration
	Generation of production and manufacturing working options
	Accounting for time and cost
Decentralization	Analytics of production and manufacturing data
	User interface dynamic adaptation of information to user profile, devices, and context
Real-time capability	Emergence of new operational models
	Optimized decision making
Service orientation	Individualized product tracking and as underlying connection layer between factories and products
Modularity	Personalization and flexibility
	Dynamic resource visualization and creation of decisional footprints at the factory and machine levels

VEF which knowledge representation of entire factory and every component of the factory viz. machines, processes, storage, material handling etc. can be traced down through it. Table 1 illustrates the benefits of this integrated framework.

Thus it can be concluded that decisional DNA based knowledge representation of objects, process and system can play a vital role in designing and implementation of thinking factory.

References

1. Posada, J., et al.: Visual computing as a key enabling technology for Industrie 4.0 and industrial internet. Comput. Graphics Appli. IEEE **35**(2), 26–40 (2015)
2. Masood, T., et al.: Integrating through-life engineering service knowledge with product design and manufacture. Int. J. Comput. Integr. Manuf. **28**(1), 59–74 (2014)
3. Reed, N., et al.: Knowledge use in an advanced manufacturing environment. Des. Stud. **32**(3), 292–312 (2011)
4. Sanin, C., et al.: Decisional DNA: a multi-technology shareable knowledge structure for decisional experience. Neurocomputing **88**, 42–53 (2012)
5. Sanín, C., et al.: Application of a multi-domain knowledge structure: the decisional DNA. In: Nguyen, N., Szczerbicki, E. (eds.) Intelligent Systems for Knowledge Management, pp. 65–86. Springer, Berlin, Heidelberg (2009)
6. Sanín, C., et al.: Decisional DNA: the concept and its implementation platforms. Cybern. Syst. **43**(2), 67–80 (2012)
7. Shafiq, S.I., Sanin, C., Szczerbicki, E.: Set of experience knowledge structure (SOEKS) and decisional DNA (DDNA): past present and future. Cybern. Syst. **45**(02), 200–215 (2014)
8. Shafiq, S.I. et al.: Using Decisional DNA to Enhance Industrial and Manufacturing Design: Conceptual Approach in Information Systems Architecture and Technology. Wrocław University of Technology, Szklarska Poreba, Wrocław, Poland (2013)
9. Shafiq, S.I. et al.: Decisional DNA based framework for representing virtual engineering objects. In: Nguyen, N., et al. (eds.) Intelligent Information and Database Systems, pp. 422–431. Springer International Publishing (2014)
10. Shafiq, S.I., et al.: Implementing virtual engineering objects (VEO) with the set of experience knowledge structure (SOEKS). Procedia Comput. Sci. **35**, 644–652 (2014)
11. Shafiq, S.I., et al.: Virtual engineering objects (VEO): designing, developing and testing models. In: Grzech, L.B.A., Swiatek, J., Wilimowska, Z. (eds.) System Analysis Approach to the Design, Control and Decision Support, pp. 183–192. Wroclaw University of Technology Press, Wroclaw (2014)
12. Shafiq, S.I., et al.: Virtual Engineering Objects: Effective Way of Knowledge Representation and Decision Making. In: Barbucha, D., Nguyen, N.T., Batubara, J. (eds.) New Trends in Intelligent Information and Database Systems, pp. 261–270. Springer International Publishing (2015)
13. Shafiq, S.I., et al.: Virtual engineering object (VEO): toward experience-based design and manufacturing for Industry 4.0. Cybern. Syst. **46**(1–2), 35–50 (2015)
14. Shafiq, S.I., et al.: Virtual engineering object/virtual engineering process: a specialized form of cyber physical system for Industrie 4.0. In: 19th International Conference on Knowledge Based and Intelligent Information and Engineering Systems. Procedia Computer Science, Singapore (2015, in press)

Fuzzy Stage Dependent Travelling Salesman Problem with Networks as Nodes

Dorota Kuchta

Abstract A new generalisation of the travelling salesman problem is proposed. First of all, the nodes of the problem are allowed to be networks themselves, in which travelling salesman subproblems have to be solved. Then, a stage dependent problem is considered, in which the nodes have different inner travel times parameters in various stages of the travelling salesman route. Such a problem is considered in a fuzzy version, when the travel time parameters may be imprecise and variable, due, for example, to weather or traffic conditions. Corresponding integer linear programming models are formulated. An example is used to illustrate the proposed approach.

Keywords Travelling salesman · Fuzzy travel times

1 Introduction

The travelling salesman problem is a well known transport optimisation problem [1, 2, 3]. It involves finding the shortest (in terms of time or physical distance) or the cheapest way of visiting n sites (nodes of a network), whereby the salesperson leaves the first site and has to come back to it after visiting all the other sites, each of them exactly once. The solution to the problem requires finding the order of visiting the sites, thus a route whose total length or cost is the smallest possible. The total distance is the objective function of the problem. In this paper we consider the travelling salesman problem with the objective function being the distance measured in time units.

The travelling salesman problem finds application, e.g. in transport companies where the drivers have to visit a set of customers exactly once per day and have to come back to the company site.

D. Kuchta (✉)
Wroclaw University of Technology, Wroclaw, Poland
e-mail: Dorota.kuchta@pwr.edu.pl

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_8

In the basic formulation of the travelling salesman problem [3] fixed distances between sites are assumed. However, in reality, especially if time units constitute the measurement units, this assumption is often invalid. This is due to the dependence of the distance on various factors, like weather or traffic conditions (the reasons for the distances being variable are discussed more in detail in [4]). This problem has been taken into account by researchers in various ways.

First of all, fuzzy numbers [5] are used to represent distances instead of crisp numbers. Fuzzy numbers may represent distances which are variable and we cannot say what they will equal to in a selected moment—we can only say that they will be "about" a certain number. Fuzzy travelling salesman problems [2, 4, 6, 7, 8] give generally a solution, thus a route, whose length is also a fuzzy number.

A drawback of the simple fuzzy approach to the modelling of the variability of the distances is the fact that using fuzzy numbers we can express that a distance will be "about" a number, but we cannot introduce (at least using one fuzzy number per each distance) a dependence of this distance in time, e.g. on the time of day (peak hours, night hours, etc.). Thus, with a basic fuzzy formulation it is difficult to take into account that it is much better to arrive later to a certain site than to another, because this avoids having to wait in traffic jams. A trial to take this into account has been undertaken in [4], by eliminating the requirement that each node can be visited only once and by allowing the salesman to stay in each node for some time.

Researchers have been trying to incorporate this aspect in stage dependent [3] or time dependent [9, 10, 11] travelling salesman problem formulations. However, all the stage dependent or time dependent travelling salesman problem formulations known to the author do not allow a stage dependent or dependent fuzzy formulation. In other words, they assume that we know a crisp function where the time is the argument and the travel time the value. In the present paper we propose the first in the literature stage dependent fuzzy formulation of the travelling salesman problem, in which the travel time is a fuzzy function, with stages as arguments and fuzzy numbers as values. This makes it possible to model such situations where we say that in a certain moment or period the travelling time will be "about" one number and in another moment or period the travel time will be "about" another number. Such an approach reflects the reality; we cannot say exactly how long we will be travelling on a certain street or road, e.g. in rush hour or during night hours. However, we can estimate those quantities using the word "about", and the values given in both cases will be substantially different.

Another novelty of the model proposed in this paper is treating the sites to be visited, thus the nodes of a network, not as single points, but as sets of points, sites or nodes to be visited one by one. Again, this reflects the situation where a driver visits different cities or towns, but once he enters a city or town, he has another set of points (e.g. a set of customers) to visit. Thus, in each site there is a travelling salesman problem to be solved, and there is one metaproblem to be solved, which incorporates all the cities and towns at one time.

A tool which can be used to determine a solution of a travelling salesman problem is integer linear programming and the corresponding algorithms [3, 9, 10, 11]. In case there are many sites to visit and not much time to find a route, this approach may create problems, because the travelling salesman problem is an NP-hard problem [1] and its solution may take much time. Here, we assume the route planners have sufficient time and, for this reason, we use integer linear programming formulation. As a tool, we use an open source linear programming/mixed integer linear programming software (http://gusek.sourceforge.net/gusek.html).

The outline of the paper is as follows. In Sect. 2 we introduce basic information about fuzzy numbers. In Sect. 3 we introduce the basic travelling salesman problem and its linear integer programming formulation. In Sect. 4 we present a stage dependent travelling salesman problem and its linear integer programming formulation. In Sect. 5 starts the original part of the paper. In this section we propose a modification of the travelling salesman problem, which involves treating each node as a network of nodes. In Sect. 6 we propose a fuzzy version of the problem from Sect. 5. In Sect. 7 we propose a stage dependent version of the model from Sect. 6, followed by the conclusions. All the considerations are illustrated with a numerical example.

The words “travelling salesman” are abbreviated as TS.

2 Basic Information About Fuzzy Numbers

We limit ourselves to the simplest form of fuzzy number: triangular fuzzy numbers [5]. They are defined as triangles of real numbers (z_1, z_2, z_3) such that $z_1 \leq z_2 \leq z_3$ and in our case, additionally, $z_1 > 0$, and denoted as $\widetilde{z} = (z_1, z_2, z_3)$. Their name comes from the shape of their so called membership function, which expresses the possibility degree to which its argument will be the crisp value in reality.

The membership function for $z_1 = 4, z_2 = 5, z_3 = 7$ (thus the fuzzy number (4, 5, 7)) is as follows:

The yet unknown magnitude represented by the fuzzy number in Fig. 1 will be most likely around 5. Values smaller than 4 and greater than 7 are considered to be

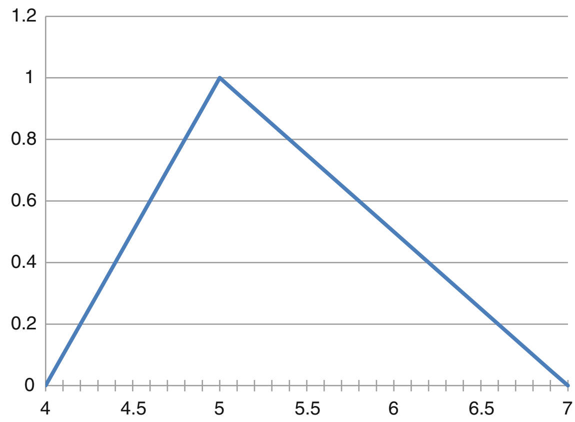

Fig. 1 Membership function of the triangular fuzzy number (4, 5, 7)

impossible. The closer a value is to 5, the higher the degree of possibility that this will be the actual value in reality.

It is important that fuzzy numbers are subject to analogous arithmetic operations as crisp numbers are. For example, addition of triangular fuzzy numbers $\widetilde{\mathrm{z}} = (\mathrm{z}_1, \mathrm{z}_2, \mathrm{z}_3)$ and $\widetilde{\mathrm{w}} = (\mathrm{w}_1, \mathrm{w}_2, \mathrm{w}_3)$ is defined as follows:

$$(\mathrm{z}_1, \mathrm{z}_2, \mathrm{z}_3) + (\mathrm{w}_1, \mathrm{w}_2, \mathrm{w}_3) = (\mathrm{z}_1 + \mathrm{w}_1,\ \mathrm{z}_2 + \mathrm{w}_2,\ \mathrm{z}_3 + \mathrm{w}_3) \tag{1}$$

Let us assume that a crisp number z is a special case of a triangular fuzzy number —it may be represented as (z, z, z).

It should be noted the serious problem of comparing or ranking fuzzy numbers (e.g. [12]). Fuzzy numbers are not always easy to compare. For example, fuzzy numbers $\widetilde{\mathrm{z}}$ = (2, 4, 8) and $\widetilde{\mathrm{w}}$ = (3, 6, 7) will be ranked differently if the parameters $\mathrm{z}_1, \mathrm{w}_1$ or $\mathrm{z}_2, \mathrm{w}_2$ are used as the ranking basis (we would have $\widetilde{\mathrm{z}} \leq \widetilde{\mathrm{w}}$ then) and if the parameters $\mathrm{z}_3, \mathrm{w}_3$ are used (the relation would be inversed in this case). There are many methods for ranking fuzzy numbers; their choice always depends on the preferences of the decision maker. For example, if the decision maker is a pessimist and the fuzzy numbers $\widetilde{\mathrm{z}}, \widetilde{\mathrm{w}}$ stand for travel times, he might use parameters $\mathrm{z}_3, \mathrm{w}_3$ as the basis for comparison. An optimist might take $\mathrm{z}_1, \mathrm{w}_1$, and someone more neutral, $\mathrm{z}_2, \mathrm{w}_2$. Defuzzification methods can be also used (the authors of [4] apply them to the fuzzy travelling salesman problem).

3 Basic Travelling Salesman Problem and Its Integer Linear Programming Formulation

There are n sites which have to be visited. The travel time between the sites $i$and $j, i,j = 1, \ldots n,\ i \neq j$ is denoted as c_{ij}. Let us define the following binary decision variables (for $i,j = 1, \ldots n,\ i \neq j$):

$$x_{ij} = \begin{cases} 1 \text{ if the TS should travel from site } i \text{ directly to site } j \\ 0 \text{ otherwise} \end{cases} \tag{2}$$

The following linear programming model [3] will yield a solution of the TS problem:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \rightarrow min \tag{3}$$

$$\sum_{i=1}^{n} x_{ij} = 1,\ \sum_{j=1}^{n} x_{ij} = 1 \tag{4}$$

Table 1 Travel length between the sites in Example 1

Sites	1	2	3	4	5
1	x	3	2	4	5
2	3	x	8	2	1
3	2	8	x	3	5
4	4	2	3	x	6
5	5	1	5	6	x

$$\sum_{i,j \in M} x_{ij} \leq |M| - 1 \tag{5}$$

for each subset M of the set $\{2, 3, \ldots n\}$ such that $2 \leq |M| < n$ where $|M|$ stands for the number of elements in M.

The objective function (3) represents the minimisation of the route total length. Constraints (4) assure that each site is visited exactly once. Constraints (5) guarantee that any site already visited (apart from the 1st) will not be visited again.

Let us consider the following:

Example 1 We have five sites to be visited. The travel time between each couple is represented in Table 1:

For example, the travel time between the 2nd and the 3rd site is eight time units.

Solving problem (2)–(5) for the data from Table 1, we get the following decision variable values: $x_{15} = x_{52} = x_{24} = x_{43} = x_{31} = 1$; the other decision variables are equal to 0. The objective function value is 13. This solution means that the shortest route of the TP will take 13 time units and this minimal time can be achieved if the TP travels from site 1 to site 5, then to site 2, then to site 4, then to 3 and, finally, back to 1.

In the next section we present such a formulation of the TP problem in which the stage a site is visited will be represented by an extra decision variable and thus can be taken into account in further modifications. To explain what is meant by the word "stage", let us note that in Example 1 site 5 is visited immediately after the TP leaves the starting point, thus in the 1st stage, but node 3 in the 4th stage.

4 Basic Stage Dependent Travelling Salesman Problem and Its Integer Linear Programming Formulation

Here, we are dealing with the same problem as in the last section. The formulation presented below is one taken from [3] and slightly modified by the author of the present paper.

The decision variables (2) are retained. Also, the objective function (3) and constraints (4) remain a part of the model. However, we have to introduce additional decision variables (for $i, j = 1, \ldots n, i \neq j$, t = 1,…, n):

$$y_{ij}^t = \begin{cases} 1 \text{ if the TP travels from site } i \text{ to site } j \text{ in stage } t \\ 0 \text{ otherwise} \end{cases} \tag{6}$$

$$x_i^t = \begin{cases} 1 \text{ if theTP enters site } i \text{ in stage } t \\ 0 \text{ otherwise} \end{cases} \tag{7}$$

Additional constraints are also necessary:

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{t=1}^{n} y_{ij}^t = 1 \tag{8}$$

$$\sum_{t=2}^{n}\sum_{j=1}^{n} t y_{ij}^t - \sum_{t=1}^{n}\sum_{k=1}^{n} t y_{ki}^t = 1 \text{ for each } i \neq 1 \tag{9}$$

$$\sum_{j=1}^{n} y_{1j}^1 = 1, \sum_{i=1}^{n} y_{i1}^n = 1 \tag{10}$$

$$x_1^t = 1 \;\text{ for } t = n \text{ and 0 otherwise} \tag{11}$$

$$x_i^n = 0 \text{ for each } \; i \neq n \tag{12}$$

$$x_j^t = \sum_{j=1}^{n} y_{ij}^t \text{ for each } j \neq 1 \text{ and each } t \neq n \tag{13}$$

$$x_{ij} = \sum_{t=1}^{n} y_{ij}^t \text{ for each couple } i, j \tag{14}$$

Constraint (8) ensures that each site is entered in exactly one stage. Constraints (9) guarantee that if a site (apart from the 1st) is entered at stage t, it is left at stage t + 1. Constraints (10) ensure that the 1st site is left in the 1st stage and reentered in the last stage. Constraints (11) and (12) guarantee the same as constraint (10), but using the other set of decision variables (we have certain redundancies in model (2)–(4), (8)–(14), but they facilitate the interpretation of the results). Constraint (13) links together decision variables (6) and (7) and constraint (14) does the same with decision variables (2) and (6).

Applying model (2)–(4), (8)–(14) to Example 1, we get the same values of decision variables (2) and the objective function as presented in Sect. 3, and the following values of decision variables (6) and (7): $y_{15}^1 = y_{52}^2 = y_{24}^3 = y_{43}^4 = y_{31}^5 = 1$, $x_5^1 = x_2^2 = x_4^3 = x_3^4 = x_1^5 = 1$; the other decision variables are equal to 0.

The formulation of the TS problem presented in this section allows making visible and using the information of the stage of the route in which each of the sites is visited.

5 Travelling Salesman Problem with Networks as Nodes

In this section we assume that each of the sites of the basic TS problem from Sect. 3 consists itself of several sites which have to be visited. Thus, for each site $i = 1, \ldots, n$ we have a TS problem with k_i sites inside the i-th site and travel time between them equal to c^i_{pr}, where $p, r = 1, \ldots, k_i, p \neq r$ represent the sites/points inside the sites of the original TS problem from Sect. 3.

In order to solve such a TP problem, it is of course enough to solve n local TP problems formulated as in Sect. 3, for each site and additionally the original TS problem (from Sect. 3) where only the distances between the sites (and not those inside the sites) are taken into account. The total travel time of the TS will be the sum of the n + 1 optimal routes' lengths: n inside the sites and 1 between the sites.

Example 2 Let us assume the same data as in Example 1. Table 1 represents the travel times between the sites.

Now, the sites themselves are networks of points to be visited. Let us assume that $k_i = 4$for $i = 1, 2, 3, 4, 5$. This is illustrated in Fig. 2:

Let us assume that we have solved the TS problem for each individual site. We do not give here the values $c^i_{pr}, i = 1, \ldots, 5;\ \ p, r = 1, \ldots 4$ or the exact optimal route within each site. We simply give the lengths of the optimal route within each site, $c^i_{min}, i = 1, \ \ldots, \ 5$: 5 for site 1, 3 for site 2, 5 for site 3, 7 for site 4 and 3 for site 5. This means that the TS has to spend at least five time units inside site 1, visiting

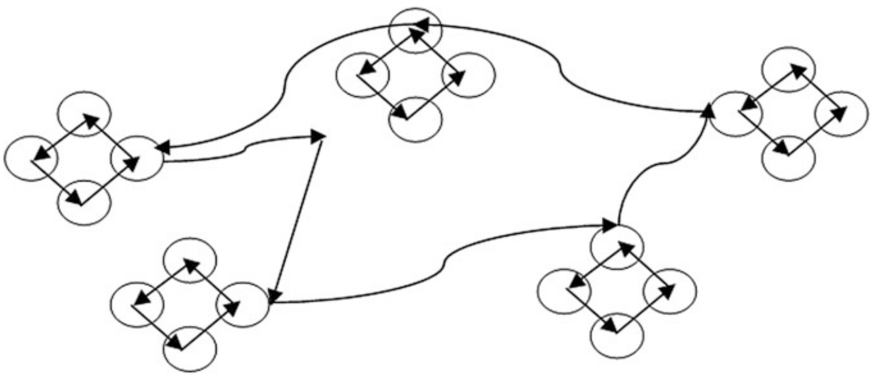

Fig. 2 Illustration of Example 2—five sites to be visited, each of them containing five subsites (an example of a route)

the four points in the site, at least three time units inside site 2, visiting its internal points, etc.

Then, the optimal route of the TS leads him from the 1st site, through the 5th, 2nd, 4th and 3rd back to the 1st, and it takes him 13 + 5 + 3 + 3 + 7 + 5 = 36 time units.

6 Fuzzy Travelling Salesman Problem with Networks as Nodes

The simplest fuzzy TS problem formulation, for the problem considered in Sect. 5, admits fuzzy numbers instead of the crisp parameters c_{ij} and c^i_{pr}, $i,j = 1,\ldots n, p, r = 1,\ldots,k_i$. Various fuzzy number types may be considered (e.g. in [4] trapezoidal fuzzy numbers are assumed), but for most practical cases triangular fuzzy numbers will be sufficient. Thus, we have parameters $\widetilde{c}_{ij}$ and $\widetilde{c}^i_{pr}$, $i,j = 1,\ldots n, p, r = 1,\ldots,k_i$, which are triangular fuzzy numbers, as defined in Sect. 2. They may also be crisp numbers, which we assumed to be a special case of triangular fuzzy numbers.

We have thus:

$$\widetilde{c}_{ij} = \left(c_{ij,1}, c_{ij,2}, c_{ij,3}\right) \tag{15}$$

$$\widetilde{c}^i_{pr} = \left(c^i_{pr,1}, c^i_{pr,2}, c^i_{pr,3}\right) \tag{16}$$

where the parameters and indices fulfill all the aforementioned relations.

Any route of the TS in such a problem has a fuzzy length, being the sum of the fuzzy lengths of the paths covered by the TS (defined according to (1)). The problem is a comparison of fuzzy numbers and is not always unequivocal (Sect. 2). Therefore, it may not be clear which route is the shortest. One of many possible approaches to solving such a problem is to consider the pessimistic, neutral and optimistic case, as illustrated in Sect. 2. Three TS problems could thus be solved with networks as nodes: respectively, for the pessimistic, neutral and optimistic travel times values. If the route obtained in all three cases is the same, the solution is unequivocal, otherwise the decision maker has to make a choice.

Example 3 Let us consider Example 2 where the distances between the sites (Table 1) remain crisp, but the distances in the sites, between the subsites, are fuzzy. This is a realistic assumption: the sites may represent cities, where traffic problems make the travel times variable and uncertain, thus fuzzy, but on the routes between the cities the travel times are fixed. We assume in the 1st site the travel times are fixed too.

Again, we do not give the values $\widetilde{c}^{i}_{pr}$, $i = 1, \ldots n, p, r = 1, \ldots, k_i$, but the lengths of the optimal routes $\widetilde{c}^{i}_{min}$(we assume that the corresponding TS problems for each site have already been solved). Thus, in site one the TS has to spend five units, in the 2nd site (2, 3, 4) time units, in the 3rd site (3, 5, 6) time units, in the 4th site (6, 7, 8) time units and in the 5th site (2, 3, 4) time units. Then, in all three cases (pessimistic, neutral and optimistic) we get the same optimal global route (we do not know whether the route in the sites is the same in all three cases; it does not have to be) whose length is fuzzy and is equal to 13 + 5 + (2, 3 ,4) + (2, 3, 4) + (6, 7, 8) + (3, 5, 6) = (36, 41, 45). Thus, in the worst case the TS would have to spend 45 units visiting all the places he has to visit, in the best case 36 time units and in the most possible case he would need to spend 41 time units travelling.

7 Stage Dependent Fuzzy Travelling Salesman Problem with Networks as Nodes

This section contains the main contribution of the present paper. It is assumed, as in Sect. 6 (and in [4, 6, 8]), that the travel times are not known exactly and thus are represented as fuzzy numbers, but, additionally, we admit the phenomenon that the fuzzy travel times estimations depend on time. We divide the day into L periods (like morning rush hours, morning normal hours, afternoon rush hours, evening hours, night hours) and for each of the periods we allow for another fuzzy travel time estimation. For each period s = 1, … L we admit other possible travel time estimations $\widetilde{c}_{ij,s}$ and $\widetilde{c}^{i}_{pr,s}$, $i, j = 1, \ldots n, p, r = 1, \ldots, k_i$, s = 1, … L. Then, for each stage of the TS route we give the information about which period of the day it will be accomplished. For example, several earlier stages might be accomplished in morning rush hour, several later stages in morning normal hours, etc. We model this information by dividing the set $\{1, 2, \ldots, t\}$ into L disjoint sets $U^s, s = 1, \ldots, L, \bigcup_{s=1}^{L} U^s = \{1, 2, \ldots, T\}$ (some of them may be empty), such that if $t \in U^s$ and the route from site i to site j is covered in the t-th stage, fuzzy estimation $\widetilde{c}_{ij,s}$ and $\widetilde{c}^{j}_{pr,s}$ are used.

Then, we propose to solve three TS problems (corresponding, respectively, to the optimistic, neutral and pessimistic case). Each would have the same decision variables and constraints (3)–(4), (8)–(14) (as in the problem in Sect. 4), but the objective functions would be as follows:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{s=1}^{L} \sum_{t \in U_s} c_{ij,s,1} y^{t}_{ij} + \sum_{j=1}^{n} \sum_{s=1}^{L} \sum_{t \in U_s} c^{j}_{min,s,1} x^{t}_{i} \rightarrow min \tag{17}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{s=1}^{L}\sum_{t\in U_s} c_{ij,s,2}y_{ij}^{t} + \sum_{j=1}^{n}\sum_{s=1}^{L}\sum_{t\in U_s} c_{min,s,2}^{j}x_i^t \rightarrow min \tag{18}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{s=1}^{L}\sum_{t\in U_s} c_{ij,s,3}y_{ij}^{t} + \sum_{j=1}^{n}\sum_{s=1}^{L}\sum_{t\in U_s} c_{min,s,3}^{j}x_i^t \rightarrow min \tag{19}$$

where $c_{min,s,1}^{j}, c_{min,s,2}^{j}, c_{min,s,3}^{j}$ are the values of the shortest routes in the jth site for the optimistic, neutral and pessimistic case, respectively, for s = 1, …, L.

Objective function (17) refers to the optimistic case, (18) to the neutral case and (19) to the pessimistic case.

Of course, the routes found in the three problems may be different. Thus, the decision maker has to make the final decision.

Example 4 Let us consider Example 3 with one modification. We assume two periods of the day: non-rush hour and rush hour (L = 2). The fuzzy evaluations of travel times in sites 2, 3, 4 and 5 are different in both periods of time and of course the shortest routes for the pessimistic, neutral and optimistic case. In Table 2 $c_{min,s,1}^{j}, c_{min,s,2}^{j}, c_{min,s,3}^{j}$ for s = 1, 2 and j = 2, 3, 4, 5 are given:

In Example 3 we used only the non-rush hour estimations. It is also worth noting that in site 2 and 3 the estimation for the pessimistic case is not very different in the non-rush and rush hours, but the estimations for the optimistic and neutral case are. In sites 3 and 4 the rush hours do not introduce a substantial change in none of the three estimations.

We also assume that the first two stages of the TS route will be accomplished in the non-rush hours and the last three in the rush hours. We have thus $U^1 = \{1,2\}$, $U^2 = \{3,4,5\}$.

Solving the problem with the objective functions (17), (18), (19) we get the following solutions:

- for the optimistic and neutral case: route 1 → 3→2 → 4→5 → 1 of the total length of 93 (optimistic case) and 115 (neutral case) time units, while the length of the travel between the sites is equal to 23 time units (the rest are travels in the sites);
- for the pessimistic case: route 1 → 3→4 → 5→2 → 1 of the total length of 195 time units, while the length of the travel between the sites is equal to 15 time units (the rest are travels in the sites).

Table 2 Optimal lengths of TS routes in the nodes for two-day periods for the pessimistic, neutral and optimistic case

	s = 1			s = 2		
	$c_{min,1,1}^{j}$	$c_{min,1,2}^{j}$	$c_{min,1,3}^{j}$	$c_{min,2,1}^{j}$	$c_{min,2,2}^{j}$	$c_{min,2,3}^{j}$
j = 2	8	9	40	30	40	50
j = 3	7	8	40	30	40	50
j = 4	20	30	40	25	35	45
j = 5	20	30	40	25	35	45

It is evident that the introduction of various estimations, depending on the period of the day, changes the situation in the case when the estimations of the travel time do not depend on the period of the day. In Example 4 there are two routes, but neither of them is that obtained in Example 1 or 3. Now, it is better to take a longer route between the sites, in order to spend less time on the way, and travelling in the sites included.

The decision maker has to choose one of the routes obtained in this example, basing his decision on his preferences and other criteria.

8 Summary

In this paper a generalisation of the stage dependent travelling salesman problem has been proposed. First, fuzziness was introduced into the stage dependent travelling salesman problem—to the author's knowledge, this has not been done previously in the literature. Furthermore, for the first time the nodes of the travelling salesman problem were considered to be networks themselves, where the travelling salesman has also to make a travelling salesman route.

The proposed problem allows taking into account the uncertainty, imprecision and variability of travelling times due to traffic problems or weather conditions, both on roads linking individual cities and in the cities themselves. Further research is needed to verify the model in practice, for real world data. Also various defuzzy fication methods may be considered, in cooperation with the decision maker.

References

1. Korte, B., Vygen, J.: Combinatorial Optimization, Theory and Algorithms. Algorithms and Combinatorics, 21st edn. Springer, Berlin, Heidelberg, New York, Tokyo, Paris, Milan (2012)
2. Michalewicz, Z., Fogel, D.B.: How to Solve It: Modern Heuristics. Springer, Berlin, Heidelberg, New York, Tokyo, Paris, Milan (2010)
3. Orman, A.J., Williams, H.P.: A survey of different integer programming formulations of the travelling salesman problem. In: Kontoghiorghes, E.J., Gatu, C. (eds.) Econometric and Financial Analysis. Advances in Computational Management Science, pp. 93–108. Springer, Berlin, Germany (2006)
4. Botzheim, J., Földesi P., Kóczy T.L.: Solution for fuzzy road transport traveling salesman problem using eugenic bacterial memetic algorithm. In: Proceedings of IFSA-EUSFLAT, pp. 1667–1672 (2009)
5. Zadeh, L.A.: A computational approach to fuzzy quantifiers in natural languages. Comput. Math Appl. **9**(1), 149–184 (1983)
6. Changdar, C., Mahapatra, G.S., Kumar Pal, R.: An efficient genetic algorithm for multi-objective solid travelling salesman problem under fuzziness. Swarm Evol. Comput. **15**(0), 27–37 (2014)
7. Gupta, A., Kumar, A., Sharma, M.K.: Applications of fuzzy linear programming with generalized *LR* flat fuzzy parameter. Fuzzy Inf. Eng. **5**(4), 475–492 (2013)

8. Isleyen, S.K., Kesen, S.E., Baykoç, Ö.F.: From analytical perspective to heuristic approach: travelling salesman problem with discrete fuzzy travel times. Gazi Univ. J. Sci. **22**(4), 267–276 (2009)
9. Cordeau, J., Ghiani, G., Guerriero, E.: Analysis and branch-and-cut algorithm for the time-dependent travelling salesman problem. Transp. Sci. **48**(1), 46–58 (2014)
10. Gouveia, L., Voss, S.: A classification of formulations for the (time-dependent) traveling salesman problem. Eur. J. Oper. Res. **83**(1), 69–82 (1995)
11. Li, F., Golden, B., Wasil, E.: Solving the time dependent traveling salesman problem (2005)
12. Bortolan, G., Degani, R.: A review of some methods for ranking fuzzy subsets. Fuzzy Sets Syst. **15**(1), 1–19 (1985)

Aspects of Structure Selection and Parameters Tuning of Control Systems Using Hybrid Genetic-Fruit Fly Algorithm

Jacek Szczypta and Krystian Łapa

Abstract In this paper a new approach for automatic design of control systems is presented. Typical control system design is difficult and time consuming. The approach proposed in this paper allows to automate this process by means of hybrid genetic-fruit fly algorithm. Genetic algorithm is used for controller structure selection, while fruit fly algorithm is used for controller parameters tuning. Proposed approach was tested on a problem of control of double spring-mass-damp object. Proposed approach allows to perform the process of control system design easier and faster.

Keywords Optimization · Evolutionary algorithm · Fruit fly algorithm

1 Introduction

Control systems design is a typical optimization problem [22, 23]. Control systems design is a difficult and time consuming task.

Usually, is based mostly on tuning of parameters of a structure which was previously selected by human expert. The tuning method might be based on analytical approach [2], based on set-point step-response [10] or based on delay and phase margin [25]. When process of parameters tuning does not give satisfactory results, human expert selects another structure and performs the tuning process once again. The structure change might require change of tuning method.

In the literature many approaches to automatization of this daunting task can be found. Those approaches are often based on computational intelligence methods

J. Szczypta (✉) · K. Łapa
Institute of Computational Intelligence, Czestochowa University of Technology, Częstochowa, Poland
e-mail: jacek.szczypta@iisi.pcz.pl

K. Łapa
e-mail: krystian.lapa@iisi.pcz.pl

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_9

such as: classifiers [8], neural networks [6, 12, 32], neuro-fuzzy systems [14, 16, 17, 28–32] and population-based algorithms [3, 5, 6].

In this paper is presented a new method for automatic design of entire control system: the selection of the control system structure and tuning of control system parameters. Proposed method is based on a PID (Proportional-Integral-Derivative) correction terms [1]. PID-based controllers fulfill the needs of most of control systems and in practice are used the most often. Moreover, in design process is used precise model of controlled object.

In our previous paper the problem of automatic design including structure selection and parameter tuning of control system based on PID using hybrid population-based algorithm was addressed [20]. Structure selection was performed by means of genetic algorithm, while parameter tuning was performed using one of the following optimization techniques: Evolutionary Algorithm, Bat Algorithm, Gravitational Algorithm and Imperialist Competitive Algorithm.

In this paper for parameter tuning is used Fruit Fly Optimization Algorithm (FOA). FOA is a novel population-based optimization technique proposed by Pan [13]. This new optimization algorithm can be characterized as simple computational process and easy to transform concepts to program code. Proposed in this paper method allows to perform difficult and time consuming task of controller design easier and faster.

This paper is organized into four sections. Section 2 presents a detailed description of the proposed approach. In Sect. 3 simulation results are drawn. Conclusions are presented in Sect. 4.

2 Proposed Approach to Control System Design

The approach proposed in this paper is based on generalized controller structure, hybrid computational intelligence algorithm, knowledge about controlled object and conditions in which control process is performed.

Idea of generalized controller structure (Fig. 1a) is a result of generalization of typical controllers used in practice: PI controller [12, 19, 27], PD controller [7, 20], cascaded PID-based controller [4, 33] and controller with feed-forward signals [10, 25]. The generalized controller is a network of interconnected Controller Blocks (CB) coupled with feedback and feedforward signals. Feedback signal is denoted as fb_{n}, where $n = 1, \ldots, N$, feedforward signal is denoted as ff_{m}, where $m = 1, \ldots, M$. CB consists of proportional term (P), integral term (I) and derivative term (D) (Fig. 1b). $\mathrm{CB}_{\mathrm{in}}$ and $\mathrm{CB}_{\mathrm{out}}$ denote CB input and CB output signal respectively. The generalized controller structure is the initial controller structure in controller design process. In Fig. 1 the connections that can be established or lost during design process were marked with dashed line.

The hybrid computational intelligence algorithm is a description of a twofold process of control system design: selection of control system structure and tuning of control system parameters. Selection of the control system structure is performed

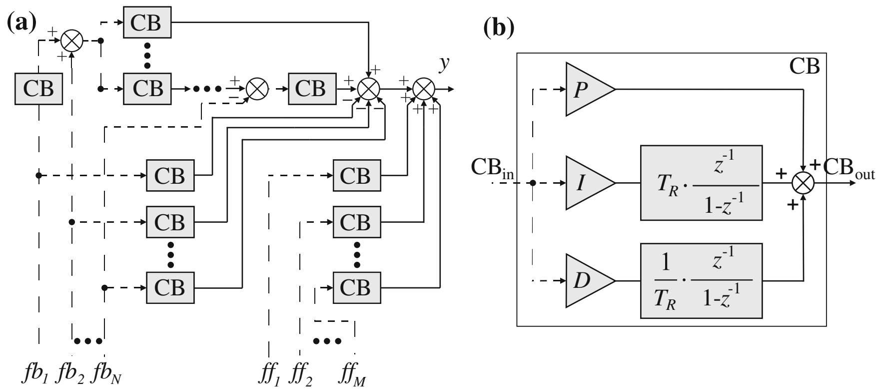

Fig. 1 Generalized controller structure: **a** control system and **b** CB definition

using genetic algorithm [15]. Tuning of the control system parameters is performed using fruit fly optimization algorithm [13]. Hybrid population-based algorithm is described in Sect. 2.2. Selection of the control system structure and tuning of parameters is performed concurrently during process of execution of hybrid population-based algorithm.

Knowledge about controlled object allows to model different states of controlled object like disabled state or damaged state.

The conditions in which the control process is performed are addressing discretization of controller output signal and feedback signals. Another phenomenon, which has to be taken into account is limitation of caused by specification of physical elements, which are used in control process as actuators.

2.1 Coding of the Structure and Parameters

In method proposed controller with its structure and parameters is encoded in a single individual $\mathbf{X}_{ch}$. The individual $\mathbf{X}_{ch}$ is described as follows:

$$\mathbf{X}_{ch} = \left\{\mathbf{X}_{ch}^{\mathrm{par}}, \mathbf{X}_{ch}^{\mathrm{red}}\right\}, \tag{1}$$

where $\mathbf{X}_{ch}^{\mathrm{par}}$ is an individual encoding correction term parameters, $\mathbf{X}_{ch}^{\mathrm{red}}$ is an individual encoding connections in general structure of the control system presented in Fig. 1. The individual $\mathbf{X}_{ch}^{\mathrm{par}}$ is described as follows:

$$\mathbf{X}_{ch}^{\mathrm{par}} = \left(X_{ch,1}^{\mathrm{par}}, X_{ch,2}^{\mathrm{par}}, \ldots, X_{ch,L}^{\mathrm{par}}\right), \tag{2}$$

$$\mathbf{X}_{ch}^{\mathrm{par}} = (P_1, I_1, D_1, P_2, I_2, D_2, \ldots), \tag{3}$$

where P_1, I_1, D_1, P_2, I_2, D_2, … denote control system parameter values, ch = 1, …, Ch denotes index of the individual in the population, Ch denotes a number of individuals in the population, L denotes length of the individual $\mathbf{X}_{ch}^{par}$.

The individual $\mathbf{X}_{ch}^{red}$ is described as follows:

$$\mathbf{X}_{ch}^{red} = \left(X_{ch,1}^{red}, X_{ch,2}^{red}, \ldots, X_{ch,L^{red}}^{red} \right), \tag{4}$$

where every gene $X_{ch,g}^{red} \in \{0,1\}$, ch = 1, …, Ch, g = 1, …, L, decides if relevant part of the control system occurs in control process (relevant gene $X_{ch,g}^{red} = 1$), L denotes length of the individual $\mathbf{X}_{ch}^{red}$.

2.2 Structure Selection and Parameters Tuning

Structure selection and parameters tuning is performed concurrently by hybrid genetic-fruit fly algorithm.

Selection of the structure encoded in $\mathbf{X}_{ch}^{red}$ is performed by means of genetic algorithm. Genetic algorithm is a well-known method [16, 22] and is based on natural selection, crossover, mutation and inheritance process.

Tuning of parameters encoded in $\mathbf{X}_{ch}^{par}$ is performed by means of Fruit Fly Algorithm. The Fruit Fly Optimization Algorithm is a method for global optimization based on the foraging behavior of the fruit fly. The fruit fly itself is superior to other species in sensing, perception, osphresis and vision. The osphresis organs of fruit flies can find all kinds of scents floating in the air and can sense food located 40 km away. Fruit flies use their sensitive vision to locate food and accompanying flies. Fruit fly in case of successful localization of the food, flies in that direction (see Fig. 2).

The main steps of hybrid genetic-fruit fly algorithm are as follows:

Step 1. Initialize individual's population. Setting maximum number of generations and population size. Setting crossover and mutation probability.
Step 2. Evaluate individual's population. Aim of this step is to evaluate, in the sense of selected criterion, the control system coded in every individual.
Step 3. Check hybrid algorithm stop condition. When the algorithm reaches the maximum number of iterations, the algorithm returns information about the best, in the sense of selected criterion, individual in population and exits. Otherwise, goes to step 4.
Step 4. Select individuals for evolutionary operations. Select individuals using roulette wheel. More in [21].

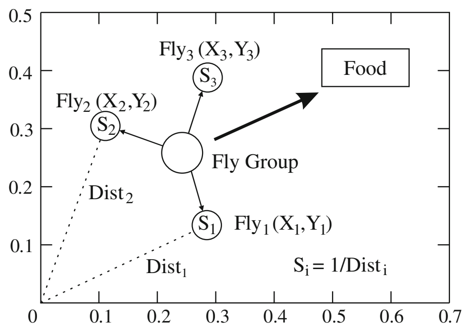

Fig. 2 The body look of the fruit fly and group food searching of fruit fly

Step 5. Apply crossover operator. More in [21].
Step 6. Apply mutation operator. More in [21].
Step 7. Use osphresis for foraging: randomly generate several fruit flies around the fruit fly group to construct a population.
Step 8. Use vision for the foraging: find the best fruit fly with the maximum smell concentration value and let the fruit fly group fly towards the best one.
Step 9. Individuals repair. Aim of the repair is to correct values of the genes to preserve parameters value in acceptable range.
Step 10. Generate offspring population and go to step 2.

2.3 Individuals Evaluation

The definition of fitness function in most of the control systems does not depend on algorithm but on problem considered. In case of selection of structure and tuning of parameters of control system, fitness function can consist the following elements: RMSE error, oscillations of the controller output signal, controller complexity and overshoot of the controlled signal. High number of the controller output signal oscillations is a negative phenomenon, because it can cause huge changes of the controller output signal and, as a result, excessive use of mechanical control parts. Overshoot is a very important issue, because is not acceptable in many industrial applications. The individual evaluation function is described as follows:

$$ff(\mathbf{X}_{ch}) = \frac{1}{RMSE_{ch} + c_{ch} \cdot w_c + os_{ch} \cdot w_{os} + ov_{ch} \cdot w_{ov}}, \tag{5}$$

where $c_{ch} > 0$ denotes the complexity of the controller structure and it is calculated by the formula:

$$c_{ch} = \sum_{g=1}^{L} \mathbf{X}_{ch,g}^{\text{red}}, \tag{6}$$

$w_c \in [0, 1]$ denotes a weight factor for the complexity of the controller structure, $os_{ch} > 0$ denotes oscillation factor of controller output signal (in simulations its value is calculated automatically), $w_{os} \in [0, 1]$ denotes a weight for the oscillation factor, $ov_{ch} > 0$ denotes value of the greatest overshoot of the controlled signal and finally $w_{ov} \in [0, 1]$ denotes a weight for the overshoot factor. RMSE function of the individual ch is described by the following formula:

$$RMSE_{ch} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \varepsilon_{ch,i}^2} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \left(s_{ch,i}^* - s_{ch,i}^1\right)^2}, \tag{7}$$

where $i = 1, \ldots, N$, denotes sample index, $N = T_L/T_S$ denotes the number of samples, $\varepsilon_{ch,i}$ denotes controller tracking error for the sample i, $s_{ch,i}^*$ denotes the reference value of the controlled signal value for the sample i, $s_{ch,i}^1$ denotes real value of the controlled signal for the sample i. In method proposed in this paper the function described by Formula (5) is maximized.

3 Simulations Results

In simulations a problem of design of controller of double spring-mass-damp object was considered (see Fig. 3). More details about this problem can be found in [11, 22]. Initial controller structure is presented in Fig. 4.

Controlled object parameters values were set as follows: spring constant $k = 10$ N/m, friction coefficient $\mu = 0.5$, masses $m_1 = m_2 = 0.2$ kg. Initial values of: s^1, v^1, s^2 and v^2 were set to zero. Simulation time length T_L was set to 10 s, trapezoid reference signal s^* is presented in Fig. 5a, sinuous reference signal s^* is presented in Fig. 5b. Ranges for genes coding controller parameters were set as follows: $P = [0, 20]$, $I = [0, 50]$, $D = [0, 5]$. Controller output signal y was limited to

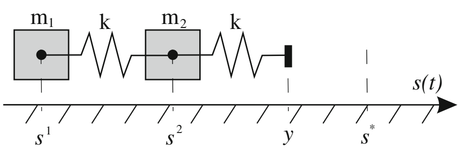

Fig. 3 Simulated spring-mass-damp object

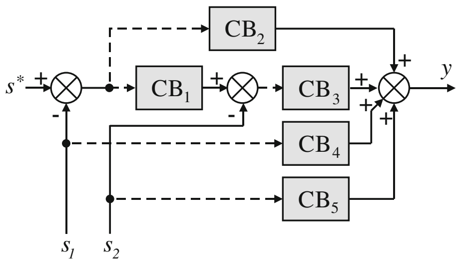

Fig. 4 Initial controller structure

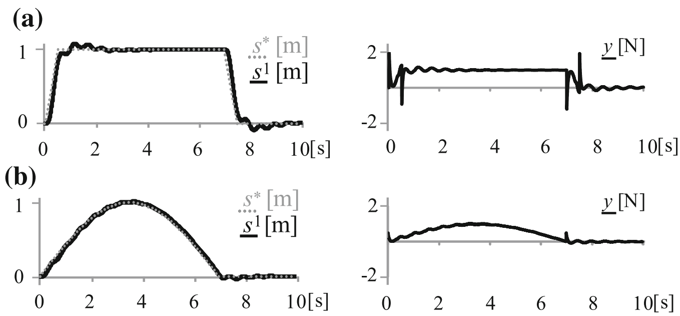

Fig. 5 Controller performance using: **a** trapezoid signal and **b** sinuous signal

the range of [−2, +2] N. Quantization resolution of the controller output signal as well as of the position sensor for s^1 and s^2 was set to 5 decimal places. Time step in the simulation was set to $T_S = 0.1$ ms, while control step was set to $T_R = 2$ ms.

The authorial environment (implemented in C# language) was used for simulations. Algorithms parameters used in simulations were set as follows: the number of individual in the population was set to 100, the number of algorithm generations (iterations) performed was set to 2000, the crossover probability was set to pc = 0.8, the mutation probability was set to pm = 0.1. Assumed weights of the fitness function are: wc = 0.001, wz = 0.01, wov = 0.0001.

In process of controller design using hybrid genetic-fruit fly algorithm trapezoid reference signal was used. Further, controller using sinuous reference signal was tested. In Fig. 5 performance of the controller obtained using hybrid genetic-fruit fly algorithm is presented. The performance is shown using controlled signal and controller output signal is shown. In Table 1 detailed information regarding fitness function and its components is shown.

Table 1 Fitness function of the best individual

Name	$RMSE_{ch}$	$c_{ch} \cdot w_c$	$os_{ch} \cdot w_{os}$	$ov_{ch} \cdot w_{ov}$	$ff(\mathbf{X}_{ch})$
Value	0.0703	0.005	0.087	<0.0001	6.1611

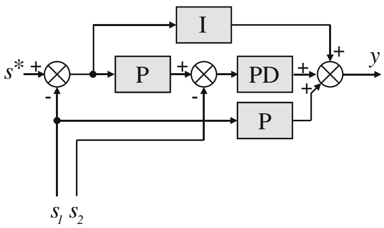

Fig. 6 Controller structure obtained in process of automatic design

Remarks related to results of the simulations can be summarized as follows: (a) Controlled signal value and controller output signal is acceptable (see Fig. 4). Controlled signal overshoot is small and is free from static error. Oscillations of controller output signal are small and diminishing in time. (b) Presented in Fig. 6 control system structure is a combination of cascaded PID controller with I-term branch from position error and feed-forward branch from signal s^1. Obtaining a similar structure using typical algorithms of control system design would be difficult and time consuming.

4 Conclusions

In this paper a new approach for automatic design of control systems is presented. In this approach a combination of genetic algorithm and fruit fly algorithm is used. In simulation results of non-complex control system characterized by sufficient control accuracy with acceptably small oscillations of controller output signal was presented. In simulations correctness of proposed in this paper approach was proved. Proposed approach allows to perform the process of control system design easier and faster.

Acknowledgments The project was financed by the National Science Centre (Poland) on the basis of the decision number DEC-2012/05/B/ST7/02138.

References

1. Astrom, K.J., Hagglund, T.: PID Controllers: Theory, Design, and Tuning. Instrument Society of America, Research Triangle Park (1995)
2. Bagheri, P., Khaki-Sedigh, A.: An analytical tuning approach to multivariable model predictive controllers. J. Process Control **24**, 41–54 (2014)
3. Cazarez-Castro, N.R., Aguilar, L.T., Castillo, O.: Fuzzy logic control with genetic membership function parameters optimization for the output regulation of a servomechanism with nonlinear backlash. Expert Syst. Appl. **37**, 4368–4378 (2010)

4. Dash, P., Saikia, L.C., Sinha, N.: Automatic generation control of multi area thermal system using Bat algorithm optimized PD–PID cascade controller. Int. J. Electr. Power Energy Syst. **68**, 364–372 (2015)
5. Dziwiński, P., Bartczuk, Ł., Przybył, A., Avedyan E.D.: A new algorithm for identification of significant operating points using swarm intelligence. Lecture Notes in Computer Science, vol. 8468, pp. 349–362. Springer, Berlin (2014)
6. Dziwiński, P., Bartczuk, L., Starczewski, J.T.: Fully controllable ant colony system for text data clustering. Artif. Intell. Soft Comput. **7269**, 199–205 (2012)
7. Fang, M.-C., Lin, Y.-H., Wang, B.-J.: Applying the PD controller on the roll reduction and track keeping for the ship advancing in waves. Ocean Eng. **54**(2012), 13–25 (2012)
8. Gabryel, M., Korytkowski, M., Scherer, R., Rutkowski, L.: Object detection by simple fuzzy classifiers generated by boosting. Artif. Intell. Soft Comput. 540–547 (2013)
9. Guzmán, J.L., Hägglund, T., Veronesi, M., Visioli, A.: Performance indices for feedforward control. J. Process Control **26**, 26–34 (2015)
10. Jeng, J.-C.: Simultaneous closed-loop tuning of cascade controllers based directly on set-point step-response data. J. Process Control **24**, 652–662 (2014)
11. Kotteeswaran, R., Sivakumar, L.: Performance evaluation of optimal PI controller for ALSTOM gasifier during coal quality variations. J. Process Control **24**, 27–36 (2014)
12. Olaofe, Z.O.: A 5-day wind speed & power forecasts using a layer recurrent neural network (LRNN). Sustain. Energ. Technol. Assessments **6**, 1–24 (2014)
13. Pan, W.-T.: A new Fruit Fly Optimization Algorithm: taking the financial distress model as an example. Knowl.-Based Syst. **26**, 69–74 (2012)
14. Przybył, A., Cpałka, K.: A new method to construct of interpretable models of dynamic systems. Lecture Notes in Artificial Intelligence, vol. 7268, pp. 697–705. Springer, Berlin (2012)
15. Rutkowski, L.: Computational Intelligence. Springer, Berlin (2008)
16. Scherer, R., Rutkowski, L.: Relational equations initializing neuro-fuzzy, system. In: 10th Zittau Fuzzy Colloquium, Zittau, Germany, pp. 18–22 (2002)
17. Scherer, R.: Neuro-fuzzy systems with relation matrix. Artificial Intelligence and Soft Computing, pp. 210–215 (2010)
18. Spinner, T., Srinivasan, B., Rengaswamy, R.: Data-based automated diagnosis and iterative retuning of proportional-integral (PI) controllers. Control Eng. Pract. **29**, 23–41 (2014)
19. Su, Y., Zheng, Ch.: Velocity-free saturated PD controller for asymptotic stabilization of spacecraft. Aerosp. Sci. Technol. **39**, pp. 6–12 (2014)
20. Szczypta, J., Łapa, K., Shao, Z.: Aspects of the selection of the structure and parameters of controllers using selected population based algorithms. In: Artificial Intelligence and Soft Computing. Lecture Notes in Computer Science, vol. 8467, pp. 440–454 (2014)
21. Szczypta, J., Przybył, A., Cpałka, K.: Some aspects of evolutionary designing optimal controllers. Lecture Notes in Artificial Intelligence, vol. 7895, pp. 91–100. Springer, Berlin (2013)
22. Theodoridis, D.C., Boutalis, Y.S., Christodoulou, M.A.: Robustifying analysis of the direct adaptive control of unknown multivariable nonlinear systems based on a new neuro-fuzzy method. J. Artif. Intell. Soft Comput. Res. **1**(1), 59–79 (2011)
23. Tran, V.N., Brdys, M.A.: Optimizing control by robustly feasible model predictive control and application to drinking water distribution systems. J. Artif. Intell. Soft Comput. Res. **1**(1), 43–57 (2011)
24. Vilanova, R., Arrieta, O., Ponsa, P.: IMC based feedforward controller framework for disturbance attenuation on uncertain systems. ISA Trans. **48**, 439–448 (2009)
25. Wang, L., Xiao, K., de Lillo, L., Empringham, L., Wheeler P.: PI controller relay auto-tuning using delay and phase margin in PMSM drives. Chin. J. Aeronaut. **27**, 1527–1537 (2014)
26. Zaky, M.S.: A self-tuning PI controller for the speed control of electrical motor drives. Electr. Power Syst. Res. **119**, 293–303 (2015)

27. Zalasiński, M., Cpałka, K., Er, M.J.: New method for dynamic signature verification using hybrid partitioning. In: Rutkowski, L., Korytkowski, M., Scherer, R., et al. (eds.) ICAISC 2014, Part II. LNCS, vol. 8468, pp. 216–230. Springer, Heidelberg (2014)
28. Zalasiński, M., Cpałka, K., Hayashi, Y.: New method for dynamic signature verification based on global features. In: Rutkowski, L., Korytkowski, M., Scherer, R., et al. (eds.) ICAISC 2014, Part II. LNCS, vol. 8468, pp. 231–245. Springer, Heidelberg (2014)
29. Zalasiński, M., Cpałka, K.: New approach for the on-line signature verification based on method of horizontal partitioning. Lecture Notes in Artificial Intelligence, vol. 7895, pp. 342–350. Springer, Berlin (2013)
30. Zalasiński, M., Cpałka, K.: Novel algorithm for the on-line signature verification. Lecture Notes in Artificial Intelligence, vol. 7268, pp. 362–367. Springer, Berlin (2012)
31. Zalasiński, M., Cpałka, K.: Novel algorithm for the on-line signature verification using selected discretization points groups. In: Rutkowski, L., Korytkowski, M., Scherer, R., et al. (eds.) ICAISC 2014, Part I. LNCS, vol. 7894, pp. 493–502. Springer, Heidelberg (2013)
32. Zhang, W., Wang, J., Zhao, Z., Tian, M.: Short-term wind speed forecasting based on a hybrid model. Appl. Soft Comput. **13**, 3225–3233 (2013)
33. Zou, H., Li, H.: Tuning of PI–PD controller using extended non-minimal state space model predictive control for the stabilized gasoline vapor pressure in a stabilized tower. Chemometr. Intell. Lab. Syst. **142**, 1–8 (2015)

On the Application of a Hybrid Genetic-Firework Algorithm for Controllers Structure and Parameters Selection

Krystian Łapa and Krzysztof Cpałka

Abstract An approach proposed in this paper uses a new hybrid population-based algorithm. This algorithm is a fusion between genetic algorithm and firework algorithm. Proposed approach aims on solving complex optimization problems in which not only structure parameters of the solution have to be selected, but also the mentioned structure. Proposed approach is based on multiple linear correction terms PID connected using proposed dynamic structure. In simulations a problem of selecting structure and its parameters for automatic control was used. For system evaluation a weighted multi-objective fitness function was used, which can consider elements connected to the simulation problems taken into consideration, such as: RMSE error, oscillations of the controller output signal, controller complexity and overshoot of the control signal.

Keywords Hybrid population-based algorithm · Selecting structure · Controller

1 Introduction

Population-based algorithms belong to heuristic algorithms and they are used mostly for solving optimization problems. They are different than traditional optimization methods because: (a) they do not process parameters of the problem, but encoded version of these parameters, (b) they search problem parameters not on the basis of one solution but they use a population of solutions, (c) they use objective function directly, not its derivatives, (d) they use probabilistic, not deterministic

K. Łapa (✉) · K. Cpałka
Institute of Computational Intelligence, Częstochowa University of Technology, Częstochowa, Poland
e-mail: krystian.lapa@iisi.pcz.pl

K. Cpałka
e-mail: krzysztof.cpalka@iisi.pcz.pl

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_10

selection of rules. Due to that they have an advantage in comparison with other kind of approaches such as analytical methods, randomization methods, etc. (see e.g. [4, 10]).

Among population-based algorithms a different approaches can be found (see Fig. 1): inspired by nature, inspired by ecology (e.g. Biogeography-Based Optimization, see [11]) or based on social evolution (e.g. Imperialist Competitive Algorithm, see [1]) etc. It is worth to mention that all population-based algorithms benefit from the evolutionary principle of survival of individuals (solutions of problem specially encoded by parameters) and use mechanisms of exploration and exploitation (defined by specified algorithm) of search space to improve fitness of individuals. The aim of population-based algorithms is to minimize or maximize values of fitness function adopted to the considered problem. Algorithms stop working when a stop condition is achieved (e.g. when value of fitness function reaches assumed value).

Most of the presented in the literature population-based algorithms cannot be used directly to solve a group of complex optimization problems (which are important from a practical point of view). Those problems concern to find both the structure and the structure parameters of solution (which is needed e.g. in neural networks, fuzzy systems, biometric systems and controllers—considered in this paper) [3, 7, 14]. Majority of population-based algorithms are focused on searching parameters of structures defined by user. For more elastic solutions (considering searching both the structure and the structure parameters) hybrid population-based algorithms can be used. In this type of algorithms the structure of the problem can be encoded for example in binary parameters and these parameters can be tuned using genetic algorithm operators. The parameters of the structure can be tuned using any population-based algorithm designed for optimization. The development of hybrid algorithm requires a proper synchronization of its algorithms, components and a proper balance between exploration and exploitation of searching space ensured.

In this paper a new hybrid population-based algorithm is presented. It is based on fusion between firework algorithm (see e.g. [13]) and genetic algorithm (see e.g. [2]). This algorithm was tested on selection of the structure and the structure parameters for controller based on linear correction terms (a controller with an object are a control system). In the literature many approaches for automatization of

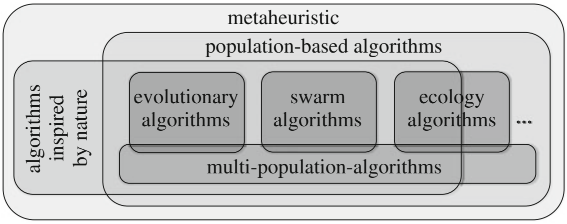

Fig. 1 Division of the population-based algorithms

this process can be found (see e.g. [5]). However, they are based mostly on selecting parameters of controllers with structures experimentally specified by experts. Thus, new and original element of the paper is (a) the proposed hybrid algorithm and (b) a way of its use for the automatic selection of the structure and parameters of the controller in automatic control system.

This paper is organized into four sections. Section 2 presents a description of problem of selecting the structure and structure parameters of controllers. In Sect. 3 a proposed hybrid population-based algorithm is presented. In Sect. 4 simulation results are drawn. Conclusions are presented in Sect. 5.

2 Description of Problem of Selecting the Structure and Structure Parameters of Controllers

In this paper a problem of selecting the structure and structure parameters based on linear correction terms is considered. Linear correction terms correspond to the needs of most automation systems (see e.g. [8]) and they are the most frequently used in practice (see e.g. [9]). Controllers which base on linear correction terms can consist of many Control Blocks (CB) (Fig. 2a). Each of CB can consist of correction terms such as: proportional (P), differential (I) and derivative (D). Moreover, each correction term consist of real number parameter: for P it is a reinforcement parameter K_p, for I it is a time constant T_i, for D it is a time constant T_d. An adequate cooperation of CB (including proper structure of the controller) should be ensured to achieve a proper quality of control.

Proposed method is based on the idea of using general structure of controller which can be modified during training process (it can be reduced or revived). The process of modification of the structure aims to obtain the most simple structure (thanks to properly defined fitness function considering, among others, complexity of the solution), which possibly best suits the control criteria. The main (general) structure of controller of MISO type (multiple input, single output) is presented in Fig. 3. It consists of a subset of selected CB blocks presented in Fig. 2a. The control signal from the block CB may be described as follows:

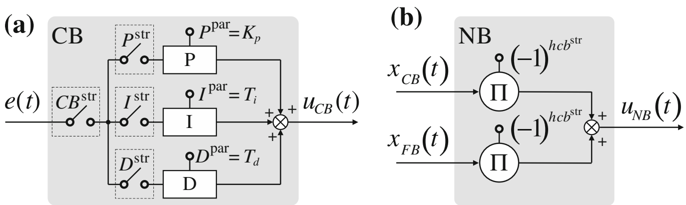

Fig. 2 Main structure of: **a** control block (CB) and **b** node block (NB)

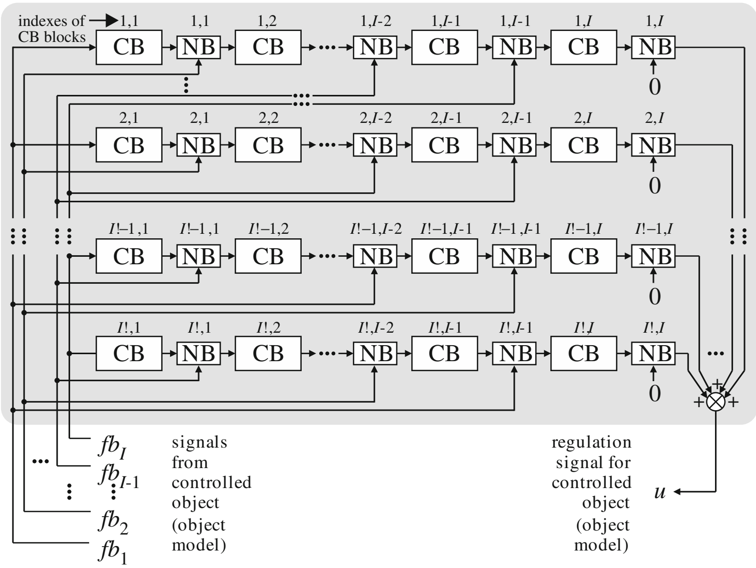

Fig. 3 The generalized structure of the controller based on a linear correction terms, designed by automatically evolutionary reduction

$$u_{CB}(t) = P^{\mathrm{str}} \cdot K_p \cdot e(t) + I^{\mathrm{str}} \cdot \frac{1}{T_i} \cdot \int_0^t e(t)dt + D^{\mathrm{str}} \cdot T_d \cdot \frac{de(t)}{dt}, \tag{1}$$

where $e(t)$ stands for input signal attached to CB block, $u_{CB}(t)$ stands for CB block output signal. Each correction term in (1) is marked symbolically by *key* $P^{\mathrm{str}} \in \{0, 1\}$, $I^{\mathrm{str}} \in \{0, 1\}$ or $D^{\mathrm{str}} \in \{0, 1\}$. The status of the *keys* corresponds to the occurrence of a merger or a break in the circuit (see Fig. 3). In the proposed method the key values are selected evolutionarily. Moreover, structure of the controller also contains Node Blocks (NB) (Fig. 2b) described as follows:

$$u_{NB}(t) = (-1)^{hcb^{\mathrm{str}}} \cdot x_{CB}(t) + (-1)^{hfb^{\mathrm{str}}} \cdot x_{fb}(t), \tag{2}$$

where is a type of feedback of signal connected to the output of the corresponding block CB (if then it is a positive feedback, if then it is a negative feedback) and a type of feedback of signal connected to the corresponding input of controller structure.

The aim of the proposed method is to select structure and parameters related to the structure. Selection of the structure is based on modification (reduction/addition) of correction terms and selection of feedbacks occurring in the controller. The selection process promotes these solutions, in which the number of attached keys is as small as possible.

3 Hybrid Genetic-Firework Population-Based Algorithm

Proposed hybrid genetic-firework population-based algorithm is a fusion between genetic algorithm and firework algorithm. The aim of genetic algorithm part is to select the structure of the controller, the aim of firework algorithm is to select the parameters of the controller. Both algorithms work simultaneously. The well-known idea of genetic algorithm is based on evolution of species (see e.g. [2]), the idea of firework algorithm is based on the behaviour of fireworks and their sparks (see e.g. [13]). In the firework algorithm each *firework* and *spark* are individuals representing single solutions for considered problem. Exploding of *firework* generates specified number of *sparks* around it and covers considered search space. After creation of *sparks*, all individuals are evaluated and best solutions are selected for next step of the algorithm (these solutions became new *fireworks*). Repeating such actions a certain number of times (resulting from the assumed number of steps of the algorithm) gives a real chance to get close to the optimal solution for the considered problem.

The next part of this section describes: **(a)** encoding method for *fireworks* and *sparks* (Sect. 3.1), **(b)** evaluation method for *fireworks* and *sparks* (Sect. 3.2) and **(c)** evolution of the *fireworks* and *sparks* (Sect. 3.3).

3.1 Encoding of the Structure and Parameters

The solutions encoded in population are marked as $\mathbf{X}_j$, $j = 1, \ldots, N$, where N stands for the number of individuals in population (each individual represent single controller). Each individual consists of two parts of parameters: $\mathbf{X}_j^{\mathrm{str}}$ and $\mathbf{X}_j^{\mathrm{par}}$ $\left(\mathbf{X}_j = \left\{\mathbf{X}_j^{\mathrm{str}}, \mathbf{X}_j^{\mathrm{par}}\right\}\right)$. The part $\mathbf{X}_j^{\mathrm{str}}$ is used to encode structure of the controller, and it is expressed as follows::

$$\mathbf{X}_j^{\mathrm{str}} = \begin{bmatrix} CB_{j,1,1}^{\mathrm{str}}, P_{j,1,1}^{\mathrm{str}}, I_{j,1,1}^{\mathrm{str}}, D_{j,1,1}^{\mathrm{str}}, hcb_{j,1,1}^{\mathrm{str}}, hfb_{j,1,1}^{\mathrm{str}}, \ldots, \\ CB_{j,1,I}^{\mathrm{str}}, P_{j,1,I}^{\mathrm{str}}, I_{j,1,I}^{\mathrm{str}}, D_{j,1,I}^{\mathrm{str}}, hcb_{j,1,I}^{\mathrm{str}}, \ldots, \\ CB_{j,I!,1}^{\mathrm{str}}, P_{j,I!,1}^{\mathrm{str}}, I_{j,I!,1}^{\mathrm{str}}, D_{j,I!,1}^{\mathrm{str}}, hcb_{j,I!,1}^{\mathrm{str}}, hfb_{j,I!,1}^{\mathrm{str}}, \ldots, \\ CB_{j,I!,I}^{\mathrm{str}}, P_{j,I!,I}^{\mathrm{str}}, I_{j,I!,I}^{\mathrm{str}}, D_{j,I!,I}^{\mathrm{str}}, hcb_{j,I!,I}^{\mathrm{str}} \end{bmatrix} = \left[X_{j,1}^{\mathrm{str}}, \ldots, X_{j,L^{\mathrm{str}}}^{\mathrm{str}}\right], \quad (3)$$

where each parameter $\mathbf{X}_{j,g}^{\mathrm{str}}$, $g = 1, \ldots, L^{\mathrm{str}}$, encodes information about corresponding *key* (CB^{str}, P^{str}, I^{str}, D^{str}, hcb^{str} or hfb^{str}) of controller structure, $L^{\mathrm{str}} = 6 \cdot I! \cdot I - I!$ stands for the number of parameters of the individual $\mathbf{X}_j^{\mathrm{str}}$ (in the practice controllers use small amount of inputs which translate into processable L^{str}). The part $\mathbf{X}_j^{\mathrm{par}}$ encodes parameters of controller, and it is defined as follows:

$$\mathbf{X}_j^{\text{par}} = \begin{bmatrix} P_{j,1,1}^{\text{par}}, I_{j,1,1}^{\text{par}}, D_{j,1,1}^{\text{par}}, \ldots, \\ P_{j,1,I}^{\text{par}}, I_{j,1,I}^{\text{par}}, D_{j,1,I}^{\text{par}}, \ldots \\ P_{j,I!,1}^{\text{par}}, I_{j,I!,1}^{\text{par}}, D_{j,I!,1}^{\text{par}}, \ldots, \\ P_{j,I!,I}^{\text{par}}, I_{j,I!,I}^{\text{par}}, D_{j,I!,I}^{\text{par}} \end{bmatrix} = \left[X_{j,1}^{\text{par}}, \ldots, X_{j,L^{\text{par}}}^{\text{par}}\right], \tag{4}$$

where each parameter $X_{j,g}^{\text{par}}$, $g = 1, \ldots, L^{\text{par}}$, encodes information about real number parameter K_p, T_i or T_d of a CB block of the controller, $L^{\text{par}} = 3 \cdot I! \cdot I$ stands for the number of parameters of individual $\mathbf{X}_j^{\text{par}}$.

3.2 Individuals Evaluation

The way of defining fitness function in most of the control systems does not depend on algorithm but on considered problem. In case of selecting structure and parameters of controller, fitness function can consist the following elements: RMSE error, oscillations of the controller output signal, controller complexity and overshoot of the control signal. This is a very important issue, because e.g. **(a)** High number of the controller output signal oscillations tends to induce an excessive use of mechanical control parts and may cause often big changes of the controller output signal value. **(b)** The overshoot of the control signal is not acceptable in many industrial applications. The individual evaluation function (maximization problem) is described as follows:

$$ff\left(\mathbf{X}_j\right) = \left(RMSE_j + c_j \cdot w_c + os_j \cdot w_{os} + ov_j \cdot w_{ov}\right)^{-1} \tag{5}$$

where $w_c \in [0, 1]$ denotes a weight factor for the complexity of the controller structure, $c_j > 0$ denotes the complexity of the controller structure described by the formula:

$$c_j = \sum_{g=1}^{L^{\text{str}}} X_{j,g}^{\text{str}}, \tag{6}$$

$w_{ov} \in [0, 1]$ denotes a weight for the overshoot factor, $ov_j \geq 0$ denotes value of the greatest overshoot of the controlled s^1 signal, $w_{os} \in [0, 1]$ denotes a weight for the oscillations factor and finally $os_j \geq 0$ denotes oscillation count of controller output signal calculated as follows:

$$os_j = \sum_{o=1}^{O-1} \left|r_o - r_{o+1}\right|, \tag{7}$$

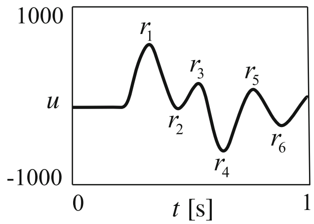

Fig. 4 Minima and maxima of the output signal u

where r_o are (sorted by time value) minima and maxima of the signal u according to Fig. 4. It is worth noting that the function of the form (7) should also count the oscillations with minimum amplitude as an undesirable phenomenon in control systems.

3.3 Evolution Process

The hybrid genetic-firework algorithm proposed in this paper works according to the following steps:

Step 1. Initialization of *fireworks* $\mathbf{X}_j$, $j = 1, \ldots, N$. Individuals in this algorithm are called *fireworks* and *sparks*. Each parameter of *firework* $\mathbf{X}_j^{\mathrm{str}}$ (Sect. 3.1) is chosen randomly from the set $\{0, 1\}$, each parameter $\mathbf{X}_j^{\mathrm{par}}$ (coding parameters of controllers) is generated randomly from ranges related to the considered problem.

Step 2. Evaluation of the initialized population. Each *firework* is evaluated by the fitness function defined in Sect. 3.2.

Step 3. Exploding of the *fireworks*. Each *firework* explodes and generates specified number of *sparks*. This number is calculated for each individual on the basis of fitness function value of *fireworks* (thanks to that more fittest *fireworks* get more *sparks* and vice versa):

$$\hat{s}_j = m \cdot \frac{y_{\max} - ff\left(\mathbf{X}_j\right) + \check{n}}{\sum_{k=1}^{n}\left(y_{\max} - ff\left(\mathbf{X}_k\right)\right) + \check{n}}, \tag{8}$$

where m is a parameter controlling total number of *sparks*, $y_{\max}$ is a best value of fitness function of *fireworks*, $\check{n}$ is a smallest constant in the computer (which prevents division by zero). This number ($\hat{s}_j$) is then reduced (projected) to the range $[b \cdot m, a \cdot m]$, where parameters of the algorithm a and b should satisfy the condition $a < b < 1$. Limiting the number of sparks is performed in order to: (a) prevent the domination of the entire population by fireworks outstanding good value of the evaluation function, (b) allocate sparks even to the fireworks which in the current step have a bad value of evaluation function. The locations of *sparks* are obtained by mimicking the *firework* explosion process. Each *spark* is a clone of *firework* (it gets the same parameters and the same structure as *firework*) with specified number of parameters modified randomly in a calculated range (range stands 'amplitude of explosion' and it is calculated individually for each *firework*). The value of amplitude of explosion for individual $\mathbf{X_j}$ depends on the fitness function value:

$$A_j = \hat{A} \cdot \frac{ff(\mathbf{X}_j) - y_{\min} + \check{n}}{\sum_{k=1}^{n} (ff(\mathbf{X}_k) - y_{\min}) + \check{n}}, \tag{9}$$

where $\hat{A}$ is a parameter controlling maximum range of *sparks*, $y_{\min}$ is the worst value of fitness function of *fireworks*. Thanks to that, fittest *fireworks* generate *sparks* close to them (exploitation) and vice versa (exploration). In this step additional number of *sparks* is generated on the basis of randomly chosen *fireworks* with modification of specified number of parameters using Gaussian explosion (to maintain diversity of population).

Step 4. Structure modification. In this step a structure of *sparks* generated in Step 3 is modified by mutation operator (known from genetic algorithm). For each *spark* a number from unit interval is generated randomly. If this number is smaller than mutation probability $p_m \in (0, 1)$, *spark* structure will be modified as follows: for each structure parameter a number from unit interval is generated randomly, if this number is smaller than mutation probability, value of the structure parameter is changed to the opposite value (from 0 to 1 and vice versa).

Step 5. Evaluation of *sparks*. In this step each *spark* generated in Step 3 and modified in Step 4 is evaluated by fitness function defined in Sect. 3.2.

Step 6. Selection of new population. New population obtains one of the actually best *firework* and $N - 1$ *sparks* chosen from *sparks* generated in Step 3 and modified in Step 4. In the original version of the algorithm process of selecting individuals takes into account only their diversity: which gives more chances to choose (by the method of roulette wheel) for those individuals with greater distance from the others. It promotes a fuller exploration of search space, however, can lead to degeneration of the population. For this reason, the proposed approach also takes into account the value of fitness function. Thus, the probability of selecting an individual $\mathbf{X}_j$ is defined as follows:

$$p(\mathbf{X}_j) = \frac{\sum_{k=1}^{K} \|\mathbf{X}_j - \mathbf{X}_k\|}{ff(\mathbf{X}_j)}. \tag{10}$$

Step 7. Replacement of the old population by population generated in the previous step. All individuals from new population are treated as *fireworks*. In this step a stop condition is checked. This condition affects the number of iterations of the algorithm. If this condition is not met, then algorithm goes back to Step 3.

Detailed information on the used algorithms can be found e.g. in [2, 13].

4 Simulations Results

In our simulations a problem of designing controller structure and tuning parameters for double spring-mass-damp object was considered (see Fig. 5). The purpose of the controller was the generation of such a control signal (acting on masses and), to adjust in the best way a position of the mass to the reference position. The motion equations for the mass m_1 (for position s^1, velocity v^1 and acceleration a^1) are described as follows:

$$s_n^1 = s_{n-1}^1 + v_{n-1}^1 \cdot T + \left(a_{n-1}^1 \cdot T^2\right) \cdot 0.5, \tag{11}$$

$$v_n^1 = v_{n-1}^1 + a_{n-1}^1 \cdot T, \tag{12}$$

$$a_n^1 = \left(\left(s_n^2 - s_n^1\right) \cdot k - v_n^1 \cdot \mu\right) \cdot m_1^{-1}, \tag{13}$$

where n and $n - 1$ denotes current and previous simulation step respectively, k is spring constant. Analogically, for mass m_2, the motion equations (for position s^2, velocity v^2 and acceleration a^2) have the following form:

$$s_n^2 = s_{n-1}^2 + v_{n-1}^2 \cdot T + \left(a_{n-1}^2 \cdot T^2\right) \cdot 0.5, \tag{14}$$

$$v_n^2 = v_{n-1}^2 + a_{n-1}^2 \cdot T, \tag{15}$$

$$a_n^2 = \left(\left(u - s_n^2\right) \cdot k - v_n^2 \cdot \mu\right) \cdot m_1^{-1}, \tag{16}$$

where u is controller output signal and μ is coefficient of kinetic friction. Remarks about considered model can be summarized as follows: **(a)** Object parameters values were set as follows: spring constant k was set to 10 N/m, coefficient of friction $\mu = 0.5$, masses $m_1 = m_2 = 0.2$ kg. Initial values of: s^1, v^1, s^2 and v^2 were set to zero (which means that the masses were in the rest state at their initial positions). **(b)** Signals fb_i used in structure were set to: s^1, s^2, $s^* - s^1$ respectively. **(c)** Simulation length was set to 10 s, a shape of the reference signal s^* (trapezoid) is presented in Fig. 6, a shape of test signal s^* (sinuous) is also presented in Fig. 6. **(d)** Search range for genes encoding controller parameters were set as follows: $P = [0, 20]$, $I = [0, 50]$, $D = [0, 5]$. **(e)** Output signal of the controller was limited to the range $u \in (-2, +2)$. **(f)** Quantization resolution for the output signal u of the controller as well as for the position sensor for s^1 and s^2 was set to 10 bit. **(g)** Time

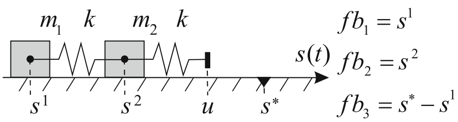

Fig. 5 Simulated spring-mass-damp object

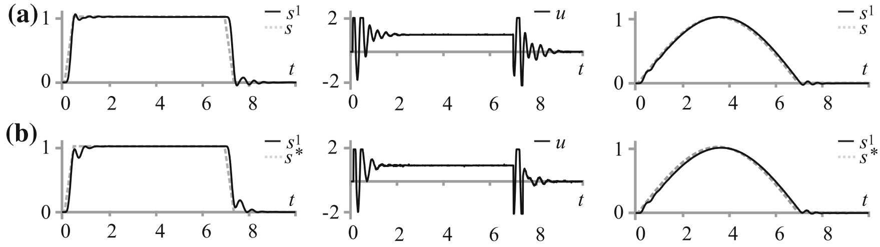

Fig. 6 Signal values s^1, $s*$ and output signal u for case: **a** FAGA-1, **b** FAGA-2

step in the simulation was equal to $T = 0.1$ ms, while interval between subsequent controller activations were set to twenty simulation steps.

The authorial environment (implemented in C++) was used for simulations. Parameters of the algorithms for the simulations were determined as follows: the number of individual (*fireworks*) N was set to 10, the number of *sparks* was set to $m = 100$ [13], the number of additional *sparks* was set to 10, bounds parameters for number of *sparks*: $a = 0.04$ and $b = 0.80$, maximum amplitude of explosion was set to $\hat{A} = 0.5$, the algorithm performs 1000 steps (generations), the mutation probability was set as $p_m = 0.3$. In our simulations, RMSE error function of the individual was described by the following formula:

$$RMSE_j = \sqrt{\frac{1}{Z}\sum_{n=1}^{Z}\varepsilon_{j,n}^2} = \sqrt{\frac{1}{Z}\sum_{n=1}^{Z}\left(s_{j,n}^{*} - s_{j,n}^{1}\right)^2}, \tag{17}$$

where $n = 1, \ldots, Z$, denotes sample index, Z denotes the number of samples, $\varepsilon_{j,n}$ denotes controller tracking error for the sample $s_{j,n}^{*}$ denotes the value of the reference signal of the controlled value for the sample n, $s_{j,n}^{1}$ denotes its current value for the sample n. Moreover, the following weights of fitness function components were used: $w_{os} = 0.0100$ and $w_{ov} = 0.0001$. At the same time two cases associated with different weight values w_c were considered:

Case 1. This case (marked further as FAGA-1) aims to obtain high accuracy of the control system: $w_c = 0.0010$.

Case 2. This case (marked further as FAGA-2) aims to obtain low complexity of the control system: $w_c = 0.0040$.

The conclusions of the simulations can be summarized as follows:

- The controllers obtained for considered in simulations problem using hybrid genetic-firework algorithms perform well (see Fig. 7): the level of oscillations (in case FAGA-1 a best value for oscillations was achieved—see Table 2) and overshooting is low, the accuracy of the system is more than satisfactory. This applies to situations, where testing of the controller was made using signal $s*$ with trapezoidal shape (used in a training phase) and also when signal $s*$ have sinusoidal shape (used only in test phase).

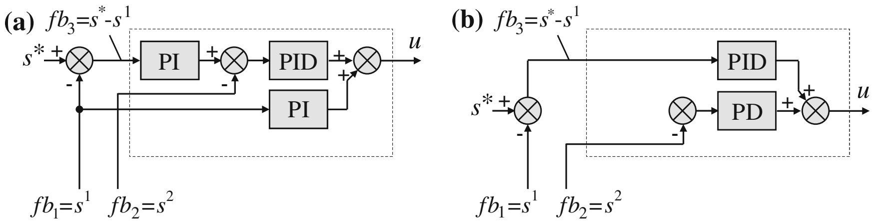

Fig. 7 Structure of the controller obtained for case: **a** FAGA-1, **b** FAGA-2 (structures denoted by *dashed line* were formed by reduction of the structure shown in Fig. 3)

– The structure of controllers for both cases considered in simulations was chosen according to assumptions. For case FAGA-1 the structure is more complex than for case FAGA-2 (see Fig. 7), but it works more accurate (see Fig. 6a, b). It is worth to mention that, in our previous work (see [12]) different methods for automatic selection the structure and the structure parameters were used, however we did not achieved as simplest structure as in case FAGA-2 (see Table 1) with comparable quality of control (see Table 2).
– In our previous works problem of selecting both the structure and the structure parameters was considered (see [6, 12]). Those papers contains methods based on simple algorithms (e.g. genetic algorithm, evolutionary strategy (μ, λ)) and on standard population-based algorithms (e.g. gravitation algorithm, firefly algorithm). Results obtained in this paper differ than results from other papers (see Tables 1 and 2). Accuracy obtained for case FAGA-1 is close to the best accuracy achieved in previous researches (see Table 2). At the same time, the structure obtained in case FAGA-2 is the simplest (in comparison to other results) and preforms appropriate well.

Table 1 Number of correction terms (*nterms*) for best individuals of population

nterms	FAGA-1	FAGA-2	Our previous results [12]
P	3	**2**	2–5
I	3	**1**	1–3
D	**1**	2	1–3
All	7	**5**	6–10

Table 2 Values of the fitness function (5) for best individuals of population

Name	FAGA-1	FAGA-2	Our previous results [12]
RMSE	0.0547	0.0623	0.0502–0.1790
$c_j \cdot w_c$	0.0070	0.0200	0.0060–0.0100
$os_j \cdot w_{os}$	0.0101	0.0113	0.0123–0.0350
$ov_j \cdot w_{ov}$	0.0004	0.0001	0.0001–0.0014
ff	13.8504	10.6723	4.9140–13.9082

5 Conclusions

In this paper a new hybrid genetic-firework algorithm is proposed. This algorithm can be used to solve complex optimization problems in which: both the structure and the structure parameters of the controller have to be found and various criteria for selection have to be taken in consideration (e.g. related to the complexity, accuracy, etc.). The algorithm was used for automatic selection of the controller, but can also be used e.g. for selection of structure and structure parameters of another kind of computational intelligence algorithm (e.g. neural network, decision tree, neuro-fuzzy systems, etc.). Results received in the simulations are positive and obtained structures of the controller are simple.

Acknowledgments The authors would like to thank the reviewers for very helpful suggestions and comments in the revision process.

The project was financed by the National Science Centre (Poland) on the basis of the decision number DEC-2012/05/B/ST7/02138.

References

1. Atashpaz-Gargari, E., Lucas, C.: Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition. IEEE Cong. Evol. Comput. **7**, 4661–4666 (2007)
2. Binitha, S., Siva, S.S.: A survey of bio-inspired optimization algorithms. Int. J. Soft Comput. Eng. (IJSCE) **2**(2) (2012)
3. Cpałka, K., Łapa, K., Przybył, A., Zalasiński, M.: A new method for designing neuro-fuzzy systems for nonlinear modelling with interpretability aspects. Neurocomputing **135**, 203–217 (2014)
4. Khajehzadeh, M., Taha, M.R., El-Shafie, A., Eslami, M.: A survey on meta-heuristic global optimization algorithms. Res. J. Sci. Eng. Technol. **3**(6), 569–578 (2011)
5. Li, W.: Design of PID controller based on an expert system. Int. J. Comput. Consum. Control (IJ3C) **3**(1), 31–40 (2014)
6. Łapa, K., Przybył, A., Cpałka, K. A new approach to designing interpretable models of dynamic systems. Lecture Notes in Artificial Intelligence, vol. 7895, pp. 523–534. Springer, Berlin (2013)
7. Łapa, K., Zalasiński, M., Cpałka, K.: A new method for designing and complexity reduction of neuro-fuzzy systems for nonlinear modelling. Lecture Notes in Artificial Intelligence, vol. 7894, pp. 329–344. Springer, Berlin (2013)
8. Malhotra, R., Sodh, R.:. Boiler flow control using PID and fuzzy logic controller. IJCSET **1** (6), 315–331 (2011)
9. Perng, J.-W., Chen, G.-Y., Hsieh, S.-C.: Optimal PID controller design based on PSO-RBFNN for wind turbine systems. Energies **7**, 191–209 (2014)
10. Rutkowski, L. Computational Intelligence. Springer, Heidelberg (2008)
11. Simon, D.: Biogeography-based optimization. IEEE Trans. Evol. Comput. **12**, 702–713 (2008)
12. Szczypta, J., Łapa, K., Shao, Z. Aspects of the selection of the structure and parameters of controllers using selected population based algorithms. In: Artificial Intelligence and Soft Computing. Lecture Notes in Computer Science, vol. 8467, pp. 440–454 (2014)

13. Tan, Y., Shi, Y., Tan, K.C. (Eds.): Fireworks Algorithm for Optimization, ICSI 2010, Part I, LNCS 6145, pp. 355–364 (2010)
14. Zalasiński, M., Łapa, K., Cpałka, K. New algorithm for evolutionary selection of the dynamic signature global features. Lecture Notes in Artificial Intelligence, vol. 7895, pp. 113–121. Springer, (2013)

Gene Expression Programming in Correction Modelling of Nonlinear Dynamic Objects

Łukasz Bartczuk

Abstract In this paper we shown the applying of gene expression programming algorithm to correction modelling of non-linear dynamic objects. The correction modelling is the non-linear modelling method based on equivalent linearization technique that allows to incorporate in modelling process the known linear model of the same or similar object or phenomenon. The usefulness of the proposed method will be shown on a practical example of the continuous stirred tank reactor modelling.

Keywords Nonlinear modeling · Dynamic objects · Gene expression programming

1 Introduction

Mathematical modelling is important and well accepted engineering technique (see e.g. [1, 2]). A simulations of various objects or phenomena allow us to better understand them because make it possible to observe internal state of object and its response to a given input signals. They are especially useful when examination of an object or phenomenon is not possible (e.g. it can lead to damage of the object or the object does not even exists yet) and allow to develop more accurate control or failure detection systems. In the literature we can find many different techniques to build mathematical models. Some of them are based on analytical methods (see e.g. [2, 3]) and the others are based on computational methods like e.g. neural networks (see e.g. [4, 5]), fuzzy systems (see e.g. [6–15]) or population based algorithms (see e.g. [16–20]). Since in the real world objects usually have non-linear characteristic so their mathematical models most often are defined as a system of non-linear

Ł. Bartczuk (✉)
Częstochowa University of Technology, Institute of Computational Intelligence, Częstochowa, Poland
e-mail: lukasz.bartczuk@iisi.pcz.pl

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_11

differential equations binding variables that describe the state of objects (the state variables model) (see e.g. [1]):

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}, \mathbf{u}), \tag{1}$$

where $\mathbf{x}$ and $\mathbf{u}$ are vectors of state variables and input signals.

However such models are more difficult to analyse and computing, so in practice they are often approximated by a linear models. This allow us to use well-established method of control theory but it can affect the accuracy of the real object mapping.

The Eq. (1) can be rewritten in form that allows for the separation of its non-linear part:

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \eta g(\mathbf{x}, \mathbf{u}), \tag{2}$$

where: $\mathbf{A}$, $\mathbf{B}$ are system and input matrices respectively, $g(\mathbf{x}, \mathbf{u})$ is a separate non-linear part of the system and η is the influence factor of the whole system non-linearities. If we assume that η is small and the system is weakly non-linear, then its linear approximation about equilibrium point will be useful in some strictly defined range. However, it should be noted that if actual operating point goes beyond the defined range, the accuracy of this model significantly decreases.

The applying of equivalent linearization techniques (see e.g. [21]) allows us to present Eq. (2) as follows:

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}, \mathbf{u}) = \mathbf{A_{eq}}\mathbf{x} + \mathbf{B_{eq}}\mathbf{u} + e(\mathbf{x}, \mathbf{u}), \tag{3}$$

where $\mathbf{A_{eq}} = \mathbf{A} + \mathbf{P_A}$ and $\mathbf{B_{eq}} = \mathbf{B} + \mathbf{P_B}$, and $e(\mathbf{x}, \mathbf{u})$ is an error term. $\mathbf{P_A}$ and $\mathbf{P_B}$ are the correction matrices whose purpose is non-linear modelling of the relationship between the known linear model and unknown non-linear model that is constructed. This separation of the system matrix $\mathbf{A}$ and the input matrix $\mathbf{B}$ allows us to incorporate some linear model of the same or similar phenomena or object into nonlinear modelling. In order to solve the accuracy decreasing problem when operating point is changing we assume that the values of the coefficients of corrections matrices are estimated for each new operating point. In our earlier papers [5, 6] this coefficients were generated by neural-fuzzy systems. In this paper we show how the genetic programming paradigm can be used to discover the functional dependencies that allows to generate an adequate values of correction matrices $\mathbf{P_A}$ and $\mathbf{P_B}$.

This paper is organized as follows: the next section contains a short description of applying the genetic programming paradigm in correctional modelling of non-linear dynamic objects. In Sect. 3 the experimental results of the application of the proposed method to modelling a continuous stirred-tank reactor are presented. The final conclusions are included in Sect. 4.

2 Genetic Programming in Nonlinear Correction Modelling of Dynamic Objects

There are many methods of the artificial or computational intelligence which are inspired by nature i.e. neural networks (see e.g. [4, 5, 22]), bat algorithm (see e.g. [23]), swarm intelligence (see e.g. [16]) and others. Among them, especially interesting are genetic and evolutionary algorithms (see e.g. [24, 25]). These methods are inspired by biological evolution and allow to solve optimization problems. One of their advantages is the ability to process multiple solutions simultaneously. In genetic algorithms, each solution is encoded in the form of linear chromosomes (individuals). In each iteration of algorithm, new solutions are formed by recombining previous solutions and only the best of them (in the sense of selected fitness function) are passed to the next iteration.

Genetic programming (see e.g. [26, 27, 28]) adopts this concept to automatically creation of computer programs that solve the considered problem. In this case individuals represent programs which are usually described in the form of a tree. Each node of the tree represents non-terminal symbols (functions) or terminal symbols (constants and inputs parameters). Set of possible functions consists of arithmetic operators, mathematical and logic functions, and should be selected carefully, according to the domain of the problem being solved. It should be noted that genetic programming is successfully applied to solve the problems of the symbolic regression and mathematical modelling (see e.g. [5]).

The problem of classical genetic programming is the representation of the program in the form of a tree because it requires that special versions of a genetic operators (i.e. crossover or mutation) have to be used. This problem is solved by Gene Expression Programming (GEP) (see e.g. [5, 29–31]). In this algorithm programs are represented in the form of linear chromosomes, which makes it possible to use standard genetic operators but require adequate conversion procedure (from the form of a tree to linear one). In original algorithm proposed by Ferreira [29–31] the population contains μ individuals. Each of them encodes one function. A chromosome $\boldsymbol{C}_{ch}, ch = 1, \ldots, \mu$, of each individual is composed of three parts:

$$\boldsymbol{C}_{ch} = \left\{ \boldsymbol{C}_{ch}^{head}, \boldsymbol{C}_{ch}^{tail}, \boldsymbol{C}_{ch}^{constants} \right\}, \tag{4}$$

where: $\boldsymbol{C}_{ch}^{head}$ can contain information about non-terminals and terminals symbols and its length $\left(\left| \boldsymbol{C}_{ch}^{head} \right| \right)$ is arbitrary, $\boldsymbol{C}_{ch}^{tail}$ can contain information about non-terminals only and its length $\left(\left| \boldsymbol{C}_{ch}^{tail} \right| \right)$ can be computed using the following formula:

$$\left| \boldsymbol{C}_{ch}^{tail} \right| = \left| \boldsymbol{C}_{ch}^{head} \right| \cdot (f_{max} - 1) + 1, \tag{5}$$

where f_{max} is a maximum arity of a non-terminal symbol. Part $\boldsymbol{C}_{ch}^{cons\tan ts}$ contains numerical constants and its length is arbitrary too.

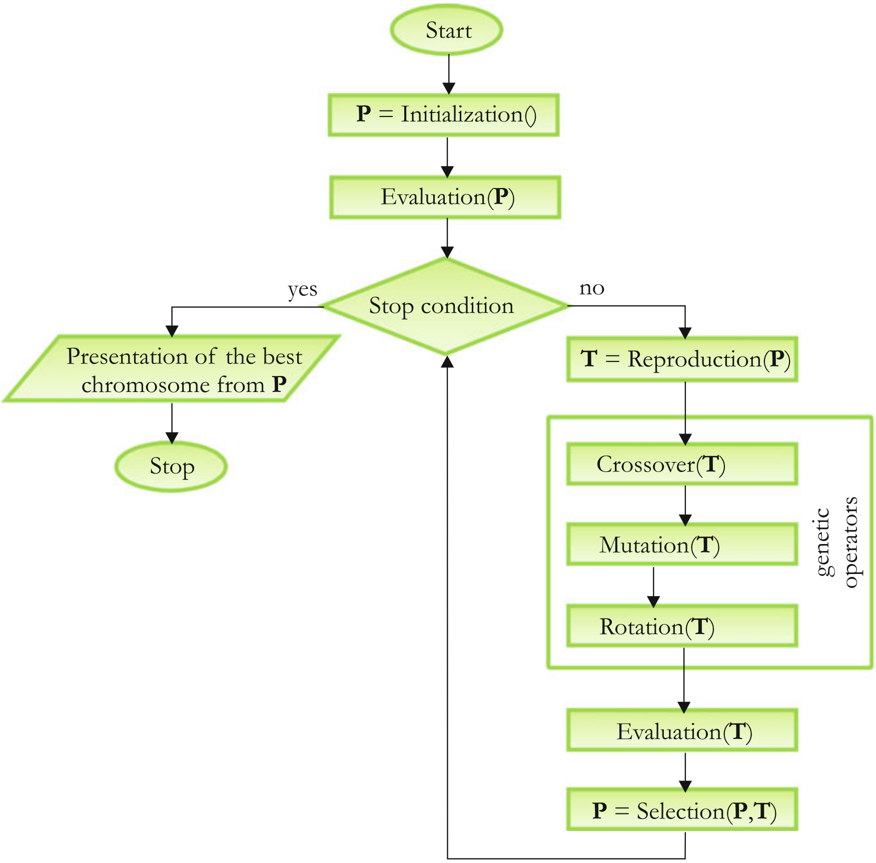

Fig. 1 Flowchart of basic GEP algorithm

The flowchart of GEP algorithm is presented in Fig. 1. Thanks to such representation of chromosome, it is possible to apply standard genetic operators used in evolutionary algorithms. In addition, some special operators like e.g. rotation can be used too.

In our implementation of GEP algorithm we assume that:

- Chromosome is composed from m 3-tuples (4), each of them encodes one of m equations required by model:

$$\mathbf{C}_{ch} = \bigcup_{j=1}^{m} \left\{ \mathbf{C}_{ch,j}^{head}, \mathbf{C}_{ch,j}^{tail} \right\} \cup \bigcup_{j=1}^{m} \left\{ \mathbf{C}_{ch,j}^{constants} \right\}. \tag{6}$$

- In order to compute the chromosome's fitness value the weighted sum method used in the multiobjective optimization algorithms (see e.g. [31]) was adopted. Thus the fitness function is defined as the average value of normalized root mean square error computed separately for each considered signal and can be written as follows:

$$ff(\boldsymbol{C}_{ch}) = \frac{1}{n}\sum_{j=1}^{n}\left(\frac{1000.0}{1.0 + \sqrt{\frac{1}{K}\sum_{k=1}^{K}\left(x_j(k+1) - \hat{x}_j(k+1)\right)^2}}\right), \tag{7}$$

where: n is a number of considered signals, K number of examples in training set, x_j is a value of jth signal generated by the created model at step $k+1$ and $\hat{x}_j$ is a reference value of jth signal at step $k+1$.

- The simple one point crossover with replacement genes is used as a crossover operation. This operation is carried out separately for part of a chromosome describing the structure of the corrections functions and for parts that contain numeric constants.
- The multigene mutation is used as a mutation operation. Similarly to the crossover, this operation is performed separately for part of a chromosome describing the structure of the corrections functions and for parts that contain numeric constants.
- We use elitist selection mechanism, so the best individual from parental population is carrying over to the next population unaltered.

3 Simulations Results

During the simulations we focused on modelling of a continuous stirred tank reactor (CSTR) (see e.g. [27, 32]) This kind of devices is very often used in many industries: chemical process, pharmaceutical, wastewater treatment etc. for mixing, reactions and crystallizations. In the CSTR reactants are continuously added into reactor, mixed and products are poured out. The simply diagram of continuously stirred tank reactor is presented on Fig. 2.

During simulations we assumed that the CSTR is used to form sulphuric acid H_2SO_4 from H_2O and SO_3 according with following chemical reaction:

$$H_2O + SO_3 \rightarrow H_2SO_4. \tag{8}$$

Based on an equation of a material balance of chemical reaction:

$$\begin{bmatrix}\text{Mass flow of}\\ \text{the reactants}\\ \text{into}\\ \text{the system}\end{bmatrix} = \begin{bmatrix}\text{Mass flow of}\\ \text{the products}\\ \text{out of}\\ \text{the system}\end{bmatrix} + \begin{bmatrix}\text{Rate of}\\ \text{accumulation}\\ \text{of mass in}\\ \text{the system}\end{bmatrix} - \begin{bmatrix}\text{Rate of}\\ \text{production of}\\ \text{the component}\\ \text{in the system}\end{bmatrix}, \tag{9}$$

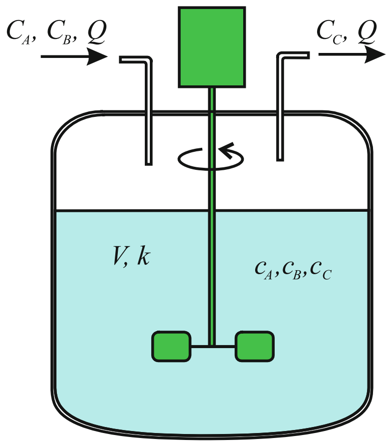

Fig. 2 A diagram illustrating the continuous stirred tank reactor

and assuming that the temperature inside the reactor is constant, the rate balance equation of reaction (9) can be presented as the system of equations:

$$\begin{cases} V\frac{dc_A}{dt} = QC_A - Qc_A - Vr \\ V\frac{dc_B}{dt} = QC_B - Qc_B - Vr, \\ V\frac{dc_C}{dt} = -QC_C + Vr \end{cases} \tag{10}$$

where c_A, c_B, c_C denote molar concentration of H_2O, SO_3 and H_2SO_4 in the reactor, C_A and C_B are concentrations of H_2O and SO_3 in the inflow, Q is a constant flow rate, V is a volume of the reactor, and r is an intrinsic rate of the reaction which can be defined by the following equation (for a some constant k):

$$r = kc_a c_b. \tag{11}$$

It should be noted that assumption about the temperature may be considered as an oversimplification of the problem, however our method can be used to solve a wide variety of problems of non-linear modelling.

Taking the $x_1(t) = c_A, x_2(t) = c_B, x_{3(t)} = c_C$ as a state variables, we can obtain the following matrix representation of the Eq. (10):

$$\begin{bmatrix} \frac{dx_1(t)}{dt} \\ \frac{dx_2(t)}{dt} \\ \frac{dx_3(t)}{dt} \end{bmatrix} = \begin{bmatrix} -\frac{Q}{V} & -kx_1(t) & 0 \\ -kx_2(t) & -\frac{Q}{V} & 0 \\ 0 & kx_1(t) & -\frac{Q}{V} \end{bmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix} + \begin{bmatrix} \frac{Q}{V} & 0 \\ 0 & \frac{Q}{V} \end{bmatrix} \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix} \tag{12}$$

In simulations of this problem we used data set generated using the set of the reactor parameters presented in Table 1.

The simulation were conducted for $T = 100$ s time interval with step $dt = 0.05$, so the training set contains 2000 samples.

Table 1 The reactor parameters used in simulations of CSTR

Reactor volume V	$2\ \mathrm{m}^3$
Flow rate Q	$0.1\ \mathrm{m}^3/\mathrm{s}$
Constant k	$0.3\ \mathrm{m}^3/\mathrm{mol}\cdot\mathrm{s}$
Initial state $\mathbf{x}(0)$	$[0.5, 0, 0]^T$
Inputs vector $\mathbf{u}\ (t)$	$[0.5, 0.5]^T$

The values of system matrix were computed as describe in [2] and can be written as follows:

$$\mathbf{A} = \begin{bmatrix} -0.1151 & -0.0651 & 0 \\ -0.0651 & -0.1151 & 0 \\ 0.0651 & 0.06513 & -0.05 \end{bmatrix}. \tag{13}$$

We assume that we know the fact that the expression for $dx_1(t)/dt$ and $dx_2(t)/dt$ do not depend on x_3, so the correction matrix can be describe as follows:

$$\mathbf{P_A} = \begin{bmatrix} p_1(\mathbf{x}) & p_2(\mathbf{x}) & 0 \\ p_3(\mathbf{x}) & p_4(\mathbf{x}) & 0 \\ p_5(\mathbf{x}) & p_6(\mathbf{x}) & p_7(\mathbf{x}) \end{bmatrix}. \tag{14}$$

Each element of matrix (14) is a function whose form is being sought by gene expression programming algorithm.

Parameters of evolutionary processes that we used in simulations are shown in Table 2.

Simulations results are presented in Fig. 3 and they can be summarized as follows:

– The best models discovered by GEP algorithm, after some arithmetical simplification can be written as:

Table 2 Parameters of the evolutionary processes used in simulations

Functions set F	$\{+, -, \cdot, /, \mathrm{pow}, \mathrm{neg}, \mathrm{abs}\}$
Head size $\left\|C_{ch,j}^{head}\right\|$	6
Number of constants $\left\|C_{ch,j}^{constants}\right\|$	10
Constants range	[−10, 10]
Number of epoch	2000
Population size μ	10
Crossover probability p_c	0.7
Mutation probability p_m	0.1

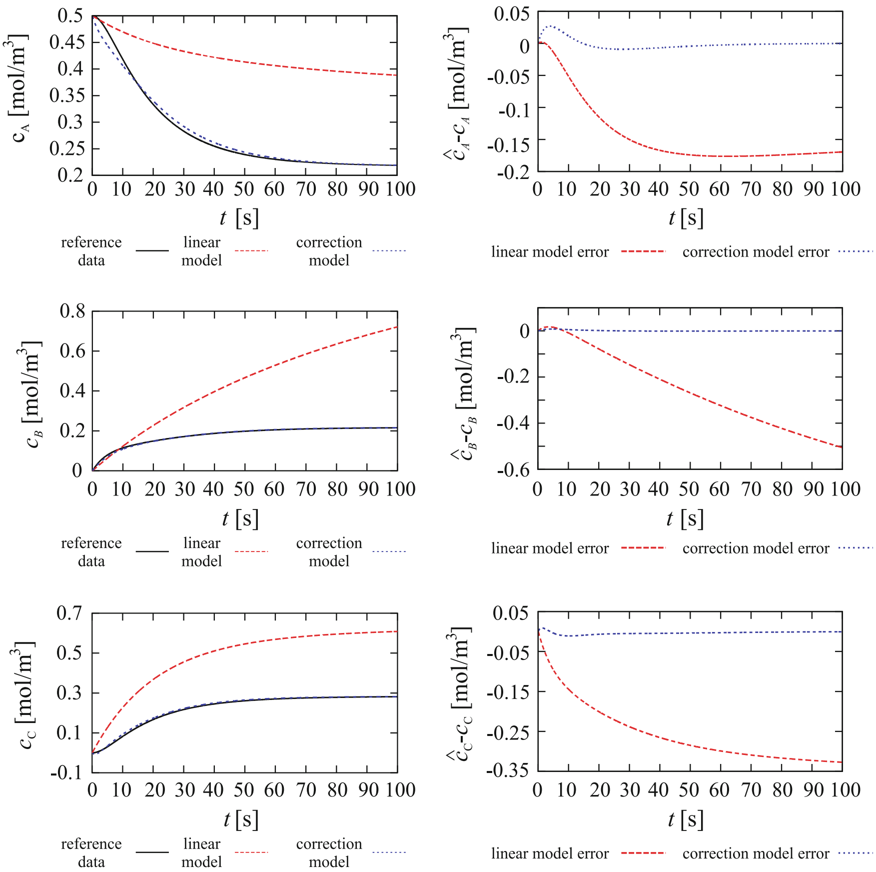

Fig. 3 The results of modelling of the continuous stirred tank reactor by the linear model and the correction model and the errors values computed as difference between reference signals and model response

$$
\begin{cases}
p_1(\mathbf{x}) = 0 \\
p_2(\mathbf{x}) = -9.4863 \cdot x_2(t) \cdot x_1^3(t) \\
p_3(\mathbf{x}) = 0 \\
p_4(\mathbf{x}) = {}^{x_1(t)}\!/_{-8.1224} \\
p_5(\mathbf{x}) = x_2(t) - x_1(t) \\
p_6(\mathbf{x}) = {}^{1.1790}\!/_{x_1(t)+7.4353} \\
p_7(\mathbf{x}) = (x_2(t) - 1) \cdot x_1(t)
\end{cases} \tag{15}
$$

– The values of root mean square error (RMSE) computed for each signal (c_A, c_B, c_C) is presented in Table 3.

As can be easily seen the correction model has achieved lower error rate than the classical linear model.

Table 3 Values of RMSE computed for each signal (c_A, c_B, c_C)

Signal	Linear model	Correction model
c_A	0.1526	0.0080
c_B	0.3017	0.0024
c_C	0.2671	0.0049

4 Conclusions

In this paper the method of developing correction model of non-linear dynamic object was presented. This method assume that we know the linear model of the same or similar object or phenomena and we looking for the way to refine it in order to increase modelling accuracy. This is achieved by adding the correction values to the system matrix. The correction values are estimated for each new operating point according with formulas discovered by gene expression programming algorithm. The presented experimental results proved the validity of the proposed method.

Acknowledgement The project was financed by the National Science Center on the basis of the decision number DEC-2012/05/B/ST7/02138.

References

1. Heinz, S.: Mathematical Modeling. Springer, Berlin (2011)
2. Łapa, K., Zalasiński, M., Cpałka, K.: A new method for designing and complexity reduction of neuro-fuzzy systems for nonlinear modelling. Lecture Notes in Artificial Intelligence, vol. 7894, pp. 329–344. Springer (2013)
3. Ismail, S., Pashilkar, A.A., Ayyagari, R., Sundararajan, N.: Neural-sliding mode augmented robust controller for autolanding of fixed wing aircraft. J. Artif. Intell. Soft Comput. Res. **2**(4), 317–330 (2012)
4 Koza JR., 1992, Genetic Programming: on the programming of computers by means of natural selection, Vol. 1. MIT Press
5. Ferreira, C.: Gene Expression Programming: Mathematical Modeling by an Artificial Intelligence, 2nd edn. Springer, Germany (2006)
6. Bartczuk, Ł., Przybył A.: New Method for Nonlinear Fuzzy Correction Modelling of Dynamic Objects. Springer LNAI 8467, pp. 169–180 (2014)
7. Bartczuk, Ł., Przybył, A., Dziwiński, P.: Hybrid State Variables—Fuzzy Logic Modelling of Nonlinear Objects. Springer LNAI 7894, pp. 227–234 (2013)
8. Cpałka, K., Łapa, K., Przybył, A., Zalasiński, M.: A new method for designing neuro-fuzzy systems for nonlinear modelling with interpretability aspects. Neurocomputing **135**, 203–217 (2014)
9. Cpałka, K., Łapa, K., Przybył, A., Zalasiński, M., Rutkowski, L.: A new method for designing neuro-fuzzy systems for nonlinear modelling with interpretability aspects. Neurocomputing **135**, 203–217 (2014)
10. Cpałka, K.: On evolutionary designing and learning of flexible neuro-fuzzy structures for nonlinear classification. Nonlinear Anal. Ser. A Theor. Methods Appl. **71**, 1659–1672 (2009). Elsevier
11. Kaczorek, T., Dzieliński, A., Dąbrowski, L., Łopatka, R.: The Basis of Control Theory. WNT, Warsaw (2006) (in Polish)

12. Lobato, F.S., Steffen Jr., V., Silva Neto, A.J.: Solution of singular optimal control problems using the improved differential evolution algorithm. J. Artif. Intell. Soft Comput. Res. **1**(3), 195–206 (2011)
13. Łapa, K., Przybył, A., Cpałka, K.: A new approach to designing interpretable models of dynamic systems. Lecture Notes in Artificial Intelligence, vol. 7895, pp. 523–534. Springer (2013)
14. Prampero, P.S., Attux, R.: Magnetic particle swarm optimization. J. Artif. Intell. Soft Comput. Res. **2**(1), 59–72 (2012)
15. Rutkowski L.: Computational Intelligence: Method and Techniques. Springer, Berlin (2008)
16. Dziwiński, P., Bartczuk, Ł., Przybył, A., Avedyan, E.D.: A new algorithm for identification of significant operating points using swarm intelligence. Lect. Notes Comput. Sci. **8468**, 349–362 (2014). Springer
17. Folly, K.: Parallel Pbil applied to power system controller design. J. Artif. Intell. Soft Comput. Res. **3**(3), 215–223 (2013)
18. Peteiro-Barral, D., Bardinas, B.G., Perez-Sanchez, B.: Learning from heterogeneously distributed data sets using artificial neural networks and genetic algorithms. J. Artif. Intell. Soft Comput. Res. **2**(1), 5–20 (2012)
19. Theodoridis, D.C., Boutalis, Y.S., Christodoulou, M.A.: Robustifying analysis of the direct adaptive control of unknown multivariable nonlinear systems based on a new neuro-fuzzy method. J. Artif. Intell. Soft Comput. Res. **1**(1), 59–79 (2011)
20. Yang, X.S.: A New metaheuristic bat-inspired algorithm. In: Nature Inspired Cooperative Strategies for Optimization (NISCO 2010), Studies in Computational Intelligence, pp. 65–74. Springer, Berlin (2010)
21. Caughey, T.K.: Equivalent linearization techniques. J. Acoust. Soc. Am. **35**(11), 1706–1711 (1963)
22. Ishibuchi H., Doi T., and Nojima Y (2006) Incorporation of scalarizing fitness functions into evolutionary multiobjective optimization algorithms, LNCS, vol. 4193, pp. 493–502. Springer
23. Vojtesek, J., Dostal, P: From steady-state and dynamic analysis to adaptive control of the CSTR reactor. In: Proceedings of 19th European Conference on Modelling and Simulation ESM, pp. 591–598
24. Przybył, A., Cpałka, K.: A new method to construct of interpretable models of dynamic systems. Lecture Notes in Artificial Intelligence, vol. 7268, pp. 697–705 Springer, Berlin (2012)
25. Eiben, A.E., Smith, J.E.: Introduction to Evolutionary Computing. Springer, Berlin (2003)
26. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer, Berlin (1996)
27. Kamyar, M.: Takagi-Sugeno fuzzy modeling for process control industrial automation, robotics and artificial intelligence (EEE8005). Sch. Electr. Electron. Comput. Eng. 8 (2008)
28. McClamroch N.H.: State Models of Dynamic Systems. A Case Study Approach. Springer, Berlin (1980)
29. Ferreira, C.: Gene expression programming: a new algorithm for solving problems. Complex Syst. **13**(2), 87–129 (2001)
30. Ferreira, C.: Gene expression programming in problem solving. Soft Computing and Industry, pp. 635–653. Springer, London (2002)
31. Chaibakhsh, A., Chaibakhsh, N., Abbasi, M., Norouzi, A.: Orthonormal basis function fuzzy systems for biological wastewater treatment processes modeling. Journal of Artificial Intelligence and Soft Computing Research **2**(4), 343–356 (2012)
32. Tran, V.N., Brdys, M.A.: Optimizing control by robustly feasible model predictive control and application to drinking water distribution systems. J. Artif. Intell. Soft Comput. Res **1**(1), 43–57 (2011)

An Application of Neural Networks to Control Stability of an Articulated Vehicle in Real Time

Kornel Warwas and Krzysztof Augustynek

Abstract The paper presents an application of neural networks to control braking torques on wheels of an articulated vehicle in an untripped rollover manoeuvre. The numerical model of the articulated vehicle and dynamic optimisation were used to calculate appropriate braking torques for each wheel in order to restore stability. The optimisation problem requires the equations of motion to be integrated at each optimisation step and the models cannot be applied for controlling the motion in real time. Therefore, in the next step multilayer perceptron network has been proposed. Multiple dynamic optimisation tasks have been solved in order to calculate appropriate braking torques, which form a training set of the neural network. In the paper, results obtained from dynamic optimisation and neural network have been presented and compared.

Keywords Neural network · Real time · Dynamic optimisation · Articulated vehicle

1 Introduction

Rollover accidents of articulated vehicles are especially violent and cause greater damage and injury than other accidents. The relatively low roll stability of trucks promotes rollovers and contributes to the number of truck accidents. Vehicle rollover accidents may be grouped into two categories, known as tripped and untripped rollovers. Tripped rollovers occur when a vehicle comes into contact with an external obstacle, such as a curb or a pothole. Anti-lock Braking Systems (ABS), Electronic Braking Systems (EBS) and Electronic Stability Programs (ESP) all help

K. Warwas (✉) · K. Augustynek
Department of Computer Science and Engineering, University of Bielsko-Biała, Bielsko-Biała, Poland
e-mail: kwarwas@ath.bielsko.pl

K. Augustynek
e-mail: kaugustynek@ath.bielsko.pl

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_12

in preventing vehicle rollovers, as they can automatically adjust the braking pattern for each wheel, possibly giving the driver greater control [1–4]. They prevent oversteering and understeering, thus preventing collisions with external obstacles. Untripped rollovers are induced by extreme driving manoeuvres, in which the forces at the contact point between the tire and the road are sufficient to cause the vehicle to rollover. Vehicle rollover accidents are typically very dangerous. Research by the National Highway Traffic Safety Administration in the United States shows that rollover accidents are the second most dangerous form of accidents in the Unites States, after head-on collisions [5]. The design of virtual computational models enables the number of experimental road tests of real vehicles to be decreased [1, 4, 6]. Computational simulations can be used in the initial phase of a vehicle design process. In the paper, the method of maintaining stability during an untripped rollover manoeuvre of an articulated vehicle is formulated. The method is based on the control of braking torques in the case of losing the stability. Braking torques patterns, which have to be applied to each wheel of the vehicle, are obtained by solving an optimisation task. The optimisation task is time-consuming, therefore neural networks are proposed to control brakes in real time.

2 Mathematical Model

The model of the articulated vehicle was formulated as a system of rigid bodies: a tractor, a fifth wheel, a semi-trailer, forming an open kinematic chain (Fig. 1).

It is assumed that the tractor is a rigid body, whose motion is described by means of six generalized coordinates, the fifth wheel has one degree of freedom (a pitch angle) in relation to the tractor, the semi-trailer has one degree of freedom (an inclination angle) with respect to the fifth wheel. Wheels are connected with the tractor and the semi-trailer and each has one degree of freedom. The tractor is set on four wheels and the semi-trailer has six wheels. Suspension stiffness has been reduced to the contact point of the tire with the road. Additionally the model contains generalized coordinates which are the front wheels' steering angles of the tractor. Generalized coordinates vectors for bodies of the articulated vehicle are shown in Table 1.

The joint coordinates vector of the articulated vehicle can be written in the following form:

$$\mathbf{q} = \begin{bmatrix} \tilde{\mathbf{q}}_T^{(1)} & \tilde{\mathbf{q}}_F^{(2)} & \tilde{\mathbf{q}}_S^{(3)} & \tilde{\mathbf{q}}_{TS}^{(1)} & \tilde{\mathbf{q}}_{TW}^{(1)} & \tilde{\mathbf{q}}_{SW}^{(3)} \end{bmatrix}^T \tag{1}$$

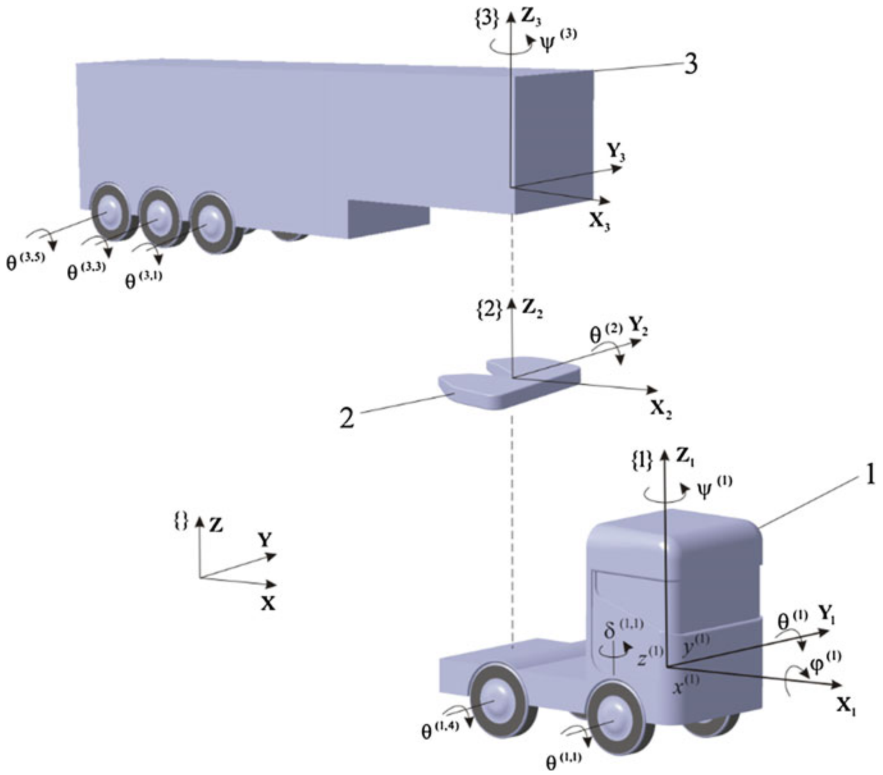

Fig. 1 The model of the articulated vehicle: (*1*) a tractor, (*2*) a fifth wheel, (*3*) a semi-trailer

Table 1 The vector of the articulated vehicle's generalized coordinates

Body name	Vector name	Vector of generalized coordinates
Tractor	$\tilde{\mathbf{q}}_{T}^{(1)}$	$\left[x^{(1)} \quad y^{(1)} \quad z^{(1)} \quad \psi^{(1)} \quad \theta^{(1)} \quad \varphi^{(1)}\right]$
Tractor suspensions	$\tilde{\mathbf{q}}_{TS}^{(1)}$	$\left[\delta^{(1,1)} \quad \delta^{(1,2)}\right]$
Tractor wheels	$\tilde{\mathbf{q}}_{TW}^{(1)}$	$\left[\theta^{(1,1)} \quad \theta^{(1,2)} \quad \theta^{(1,3)} \quad \theta^{(1,4)}\right]$
Fifth wheel	$\tilde{\mathbf{q}}_{F}^{(2)}$	$\left[\theta^{(2)}\right]$
Semi-trailer	$\tilde{\mathbf{q}}_{S}^{(3)}$	$\left[\psi^{(3)}\right]$
Semi-trailer wheels	$\tilde{\mathbf{q}}_{SW}^{(3)}$	$\left[\theta^{(3,1)} \quad \theta^{(3,2)} \quad \theta^{(3,3)} \quad \theta^{(3,4)} \quad \theta^{(3,5)} \quad \theta^{(3,6)}\right]$

where
$x^{(i)}, y^{(i)}, z^{(i)}$—mass center coordinates of the *i*th body
$\psi^{(i)}, \theta^{(i)}, \varphi^{(i)}$—rotation angles of the ith body
$\delta^{(i)}$—front wheels' steering angle of the vehicle

Equations of vehicle motion have been formulated using Lagrange formalism and homogenous transformations [7]. It can be written in the general form [6]:

$$\mathbf{A}\ddot{\mathbf{q}} = \mathbf{f} \tag{2}$$

where

$\mathbf{A} = \mathbf{A}(t, \mathbf{q})$—mass matrix,
$\mathbf{f} = \mathbf{f}(t, \mathbf{q}, \dot{\mathbf{q}}, \mathbf{M}^{(1)}, \ldots, \mathbf{M}^{(i)} \ldots, \mathbf{M}^{(n_w)})$—vector of external, Coriolis and centrifugal forces,
$\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}$—displacement, velocity and acceleration vectors,
$\mathbf{M}^{(i)}$—vector of discrete values of braking torques acting on the ith wheel,
n_w—number of wheels.

The details of the procedure which leads to formation of Eq. (2) with a description of elements in the matrix $\mathbf{A}$ and the vector $\mathbf{F}$ is presented in [6].

3 Formulation of the Problem of Control Braking Moments

As mentioned before, one of the most dangerous road manoeuvres for articulated vehicles is an untripped rollover. This situation happens mostly during a lane-change manoeuvre [5]. Stability of the articulated vehicle can be restored by an appropriate control of braking torques applied to each wheel of the vehicle. Let us consider a vector of braking torque discrete values $\mathbf{M}^{(i)}$ of the ith wheel. A continuous function $M^{(i)}(t)$ will be obtained using spline functions of the 3rd order. The vector of the decisive variables contains discrete values of the braking torques of wheels and can be written in the form:

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}^{(1)} & \ldots & \mathbf{M}^{(i)} & \ldots & \mathbf{M}^{(n_w)} \end{bmatrix}^T = (M_j)_{j=1,\ldots,m} \tag{3}$$

where

m—number of decisive variables,
$\mathbf{M}^{(i)} = \left(M_k^{(i)}\right)_{k=1,\ldots,n}$,
N—number of discrete values of the braking moment.

The stability conditions can be assured by the solution of a dynamic optimisation problem in the general form [8]:

$$\Omega(\mathbf{M}, \mathbf{q}, \dot{\mathbf{q}}) \rightarrow \min \tag{4}$$

Calculations of the objective function (4) require integration of model equations (2). The proper solution of the braking torques control task can be achieved if additional physical constrains are considered. These conditions can be included through additional inequality and equality constrains [8], as follows:

$$\mathbf{g}(\mathbf{M}, \mathbf{q}, \dot{\mathbf{q}}) \leq 0 \tag{5}$$

$$\mathbf{h}(\mathbf{M}, \mathbf{q}, \dot{\mathbf{q}}) = 0 \tag{6}$$

In the presented problem, braking torques calculated for fixed initial vehicle velocity and the front wheels' steering angle have to fulfil the following conditions:

- the articulated vehicle cannot lose stability during the manoeuvre,
- longitudinal velocity loss has to be as small as possible,
- lateral displacement of the vehicle is limited by the standard road width,
- after the manoeuvre, the vehicle has to move parallel to the axis of the road.

Above assumptions are taken into account in the objective function and also in optimisation constraints (5) and (6). The stability conditions can be assured by minimizing the functional:

$$\Omega(\mathbf{M}, \mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{t_e}\left(C_1 \int_0^{t_s} \left(\varphi^{(1)}\right)^2 dt + C_2 \int_0^{t_s} \left(v_0 - \dot{x}^{(1)}\right)^2 dt \right) \to \min \tag{7}$$

where

C_1, C_2—empirical coefficients,
t_s—time of simulation,
v_0—initial velocity.

In the considered optimisation problem, constraints can be written as follows:

$$\mathbf{g} = \begin{bmatrix} M_1 - M_{max} & \ldots & M_j - M_{max} & \ldots & M_m - M_{max} \end{bmatrix}^T \tag{8}$$

$$\mathbf{h} = \begin{bmatrix} \psi^{(1)}(t_d) & \dot{\psi}^{(1)}(t_d) \end{bmatrix}^T \tag{9}$$

where

M_{max}—acceptable maximal braking torque,
t_d—duration of the manoeuvre.

In order to solve the minimization task (4), the Nelder-Mead method with an exterior penalty method was used [8, 9].

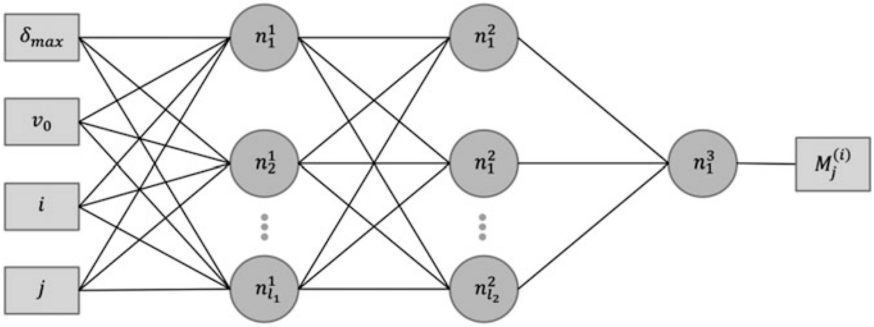

Fig. 2 The topology of the neural network

4 Neural Networks

The solution of the dynamic optimisation problem is time-consuming and cannot be applied to a brake controller in order to control braking torques in the articulated vehicle. The real-time response can be achieved by using an artificial neural network. The network has to be trained on the basis of optimal solutions of the optimisation problem for different values of input parameters. It is possible to build artificial neural network which after training will be able to predict braking torques on wheels of the vehicle with a satisfying accuracy. In order to determine optimal braking torque courses, the multilayer perceptron [10, 11] network has been used. The network consists of four input signals: the velocity (v_0), the maximal steering angle (δ_{max}), the index of the wheel (i) and the index of the braking moment (j) in discrete time (t_j). Only one output is necessary in the considered case. It is braking torque $M_j^{(i)}$ of the ith wheel in the time step t_j. After the input has been entered into the neural network, the response is calculated using a transfer function (Fig. 2).

As the transfer functions the hyperbolic tangent and linear functions were used, for the hidden and the output layer respectively.

5 Numerical Simulations

Algorithms related to formulation and solution of the multibody system and its simulation and optimisation have been implemented in the own program [6] written in the C++ language. In order to prepare and simulate neural networks own programs also have been written using the Encog library [12] and its .NET implementation.

Physical parameters of the articulated vehicle model were taken from [6]. Interpolation of the braking moment has been performed for $n = 4$ discrete values. The Bulrish-Stoer-Deuflhard [8] method with an adaptive step size has been used for integrating the equations of motion. Duration of a single evaluation of the objective function was 5 s. Results obtained from multiple optimisation tasks have been used as learning sets for the neural network. Each optimisation task has been solved by means of the gradientless Nelder-Mead method [8, 9]. During simulations the maximum value δ_{max} of the steering angles $\delta^{(1,1)}$, $\delta^{(1,2)}$ and the initial velocity v_0 of the tractor were changed. It contains data for v_0 from 45 to 85 km/h with a step of 5 km/h and steering angles matched to the rollover vehicle. It has been assumed that the neural network contains two hidden layers with $l_1 = 30$ and $l_2 = 20$ neurons respectively.

During the training process of the neural network, the efficiency and the convergence of gradient and non-gradient training methods have been studied. In the analysis, the following methods have been taken into consideration:

– Resilient backpropagation (RB),
– Levenberg-Marquardt (LM),
– Quickprop (QP),
– Scaled conjugate gradient (SC),
– Nelder-Mead (NM),
– Genetic algorithm (GA),
– Simulated annealing (SA).

In order to compare the effectiveness of the training methods, let us introduce a coefficient which describes the relation between the duration of network training for the particular method and the duration of the fastest method:

$$\gamma = \frac{e_i}{e_b} \tag{10}$$

where

e_i—time of training with the i method,
$i \in \{RB, LM, QP, SC, NM, GA, SA\}$,
e_b—time of training with the fastest method.

A set of simulations has been carried out with different training methods. Results show that the fastest training method is the resilient backpropagation. Figure 3 presents a comparison of γ coefficient values and the number of epochs obtained for different training methods.

Figure 3 shows that the gradient methods are the most effective. In the case of training using genetic algorithms (GA), the Nelder-Mead (NM) and simulated annealing (SA) methods, assumed threshold of network error rate has not been achieved or the training time was many times longer than in gradient methods. Due to the short time of network training and small influence of the starting point on convergence of the method in further analysis, the resilient backpropagation method was used.

In order to numerically evaluate quality of courses obtained from the trained network, the integral error has been calculated according to the formula:

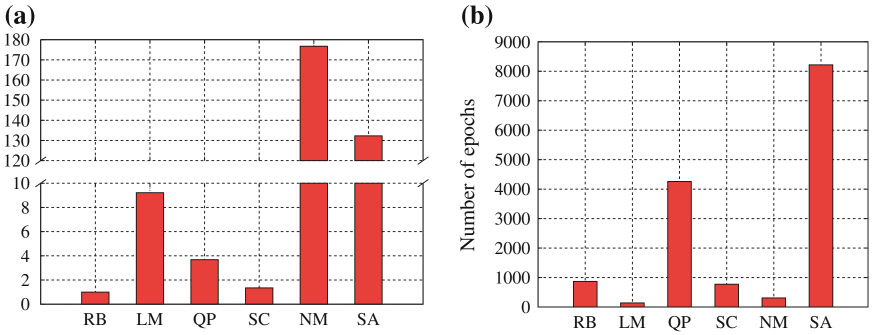

Fig. 3 A comparison of **a** γ coefficient values and **b** the number of epochs obtained for different training methods

$$\varepsilon = \left| \frac{I_O - I_{NN}}{I_O} \right| \cdot 100\ \% \tag{11}$$

where

$I_O = \frac{1}{T}\sum_{i=1}^{n_w} \int_0^T M^{(i)}(t)dt$—integral mean value for optimisation results,
$I_{NN} = \frac{1}{T}\sum_{i=1}^{n_w} \int_0^T M_{NN}^{(i)}(t)dt$—integral mean value for trained network results.

The integral error between courses obtained from the network and its training data set has been presented in Table 2.

Figure 4 shows numerical simulations for the maximal steering angle $\delta_{max} = 8.5°$ and the vehicle initial velocity $v_0 = 55$ km/h. Figure 4a presents course of the steering angles which have been applied to the front wheels of the articulated vehicle. Results obtained from: simulation before optimisation (BO), Nelder-Mead optimisation method (OP), trained neural network (NN) have been presented in Fig. 4b–d.

Table 2 The integral error for some results obtained from the trained neural network

Initial velocity v_0 (km/h)	Maximal steering angle δ_{max} (°)	Integral error ε (%)
65	6.2	0.1896
60	7.0	0.1319
55	8.5	0.1490
50	10.0	0.4029
45	12.5	0.0658

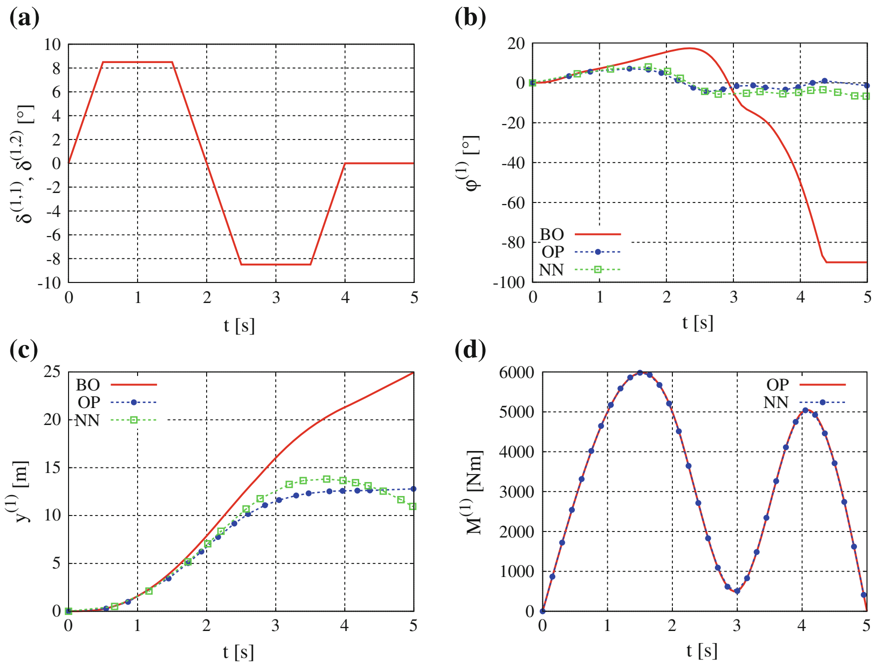

Fig. 4 Courses of: **a** steering angles, **b** the tractor roll angle, **c** the tractor lateral displacement, **d** the braking moment of the 1st wheel obtained for $\delta_{max} = 8.5°$ and $v_0 = 55$ km/h

It shows courses of the tractor roll angle, the tractor vertical displacement and the braking moment of the 1st wheel. For both analysed cases, the results obtained from optimisation and the trained neural network are acceptable. The roll angle and the lateral displacement are significantly smaller than before optimisation. Braking torques obtained from optimisation and the neural network allow us to restore stability of the articulated vehicle. Additionally, trajectory of the vehicle fulfils all the requirements. The main difference between the simulation using the Nelder-Mead dynamic optimisation and the one using the neural network is the time in which optimal braking torques are calculated. In the case of the neural network, the calculation can be performed in real time. After training, neural networks can be used for a simulation with input parameters different than in the training set. Steering angle course and results of simulations are presented in Fig. 5.

Figure 5 shows that the vehicle is stable and the trajectory is achieved when braking torques obtained from the neural network are applied. Quality of results obtained before optimisation, from the Nelder-Mead method and the neural network, can be assessed by using root mean square (RMS) of the tractor roll angle $\varphi^{(1)}$ (Fig. 6b). RMS has been calculated according to the formula:

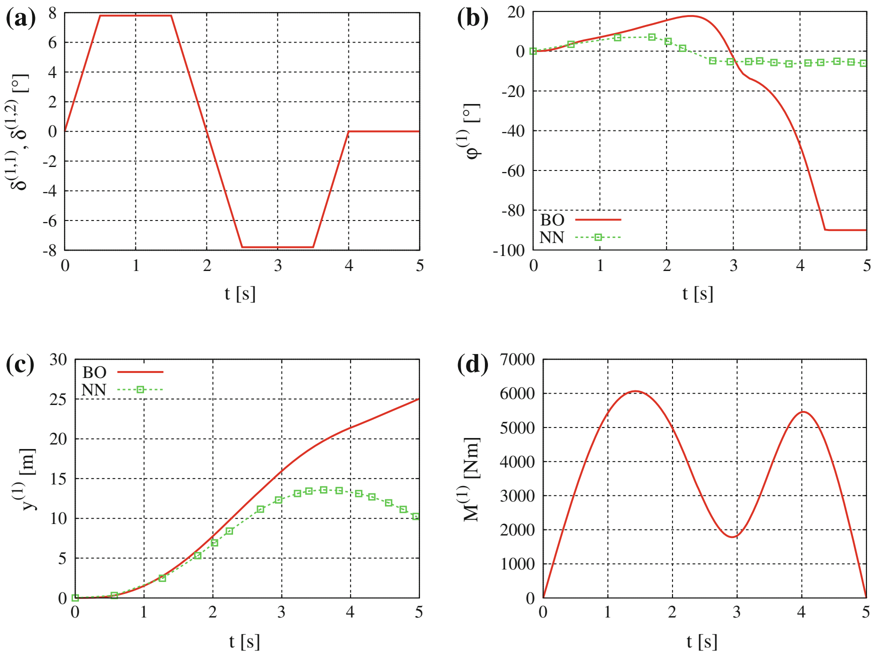

Fig. 5 Courses of: **a** steering angle, **b** tractor roll angle, **c** tractor lateral displacement, **d** braking moment of 1st wheel obtained for $\delta_{max} = 7.8°$ and $v_0 = 57.5$ km/h

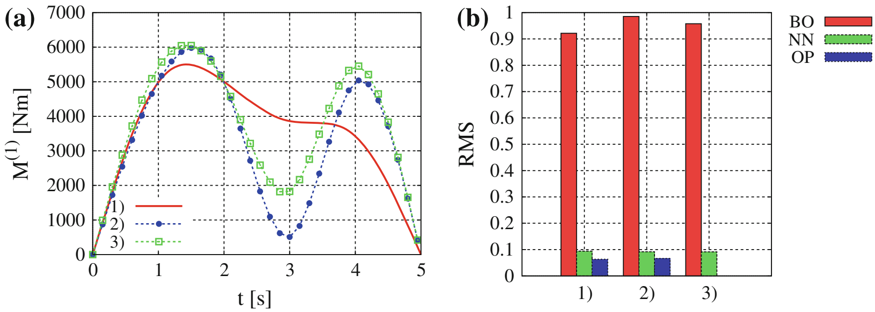

Fig. 6 A comparison of **a** braking torque courses of the 1st wheel and **b** RMS values of the tractor roll angle, (1) $\delta_{max} = 7°$, $v_0 = 60$ km/h, (2) $\delta_{max} = 8.5°$, $v_0 = 55$ km/h, (3) $\delta_{max} = 7.8°$, $v_0 = 57.5$ km/h

$$\Delta = \sqrt{\frac{\int_0^{t_e} \left(\varphi^{(1)}\right)^2 dt}{t_e}} \tag{12}$$

Figure 6a presents braking torque courses of the 1st wheel obtained from the neural network for all considered earlier cases of simulations.

As can be seen, the tractor roll angle's RMS values obtained from optimisation and the neural network are significantly smaller than before optimisation. Braking torques calculated for input parameters different than in the training set (series 3) have features of other courses (series 1, 2).

6 Conclusions

The articulated vehicle rollover is strongly associated with severe injury and fatalities in highway accidents. Stability can be achieved by an appropriate control of braking torques. In the paper, the problem of controlling brakes is formulated and solved. The long duration of the optimisation process has resulted from the necessity of integrating equations of motion in each step. Despite reducing the time of calculations, it is still difficult to provide a real-time solution. In the paper, the neural network has been applied to solve this problem. Trained and validated neural network can be used in real-time prediction of braking torques. After applying the braking torques obtained from the trained neural network, vehicle stability is restored, the roll angle and the lateral displacement are smaller than before optimisation. The presented issue has already been analysed in [6]. In this paper, topology of the neural network is simpler. Two additional input signals have been added and the number of outputs has been reduced to only one. As a result, the number of neurons and weights is smaller and the network can be trained in an easier way, which consumes less time than before.

References

1. Huang, H.H.: Controller design for stability and rollover prevention of multi-body ground vehicles with uncertain dynamics and faults. Ph.D. Thesis, Graduate School of The Ohio State University, Ohio (2009)
2. Yedavalli, R.K.: Robust stability and control of multi-body ground vehicles with uncertain dynamics and failures. Technical Report, Ohio State University Research Foundation, Ohio (2010)
3. Carlson, C.R., Gerdes, J.C.: Optimal rollover prevention with steer by wire and differential braking. In: Proceedings of the ASME International Mechanical Engineering Congress and Exposition, Washington (2003)
4. Yao, Z., et al.: Dynamic simulation for the rollover stability performances of articulated vehicles. J. Automobile Eng. 771–783 (2014)
5. Forkenbrock, G.J., Garrot, W., Heitz, M., O'Harra, B.C.: A comprehensive experimental examination of test maneuvers that may induce on-road, untripped, light vehicle rollover—phase iv of NHTSA's light vehicle rollover research program. Technical Report, National Highway Traffic Safety Administration (2002)
6. Warwas, K.: Analysis and control of motion of articulated vehicles with flexible elements. Ph. D. Thesis, University of Bielsko-Biała, Bielsko-Biała (2008)
7. Bauchau, O.A.: Flexible Multibody Dynamics, Solid Mechanics and its Applications. Springer, Dordrecht (2011)

8. Press, W., Teukolsky, S., Vetterling, W., Flannery, B.: Numerical Recipes 3rd Edition: The Art of Scientific Computing. Cambridge University Press, Cambridge (2007)
9. Luersen, M., Le Riche, R.: Globalized Nelder–Mead method for engineering optimisation. Comput. Struct. 2251–2260 (2004)
10. Haykin, S.O.: Neural Networks and Learning Machines. Prentice Hall, Englewood Cliffs (2009)
11. Gupta, M., Jin, L., Homma, N.: Static and Dynamic Neural Networks. Wiley, New York (2003)
12. Heaton, J.: Programming Neural Networks with Encog3 in C#, Heaton Research (2011)

Extended Recognition Effectiveness of a Sonar System

Teodora Dimitrova-Grekow and Marcin Jarczewski

Abstract Although sonars are among the most popular elements used in intelligent navigation vehicles, they still have unexplored specifics offering useful information. In this paper, we present further results of experiments focused on the exploration of sonars' drawbacks. We combined the regular information obtained from a simple sonar system with information derived from measurement abnormalities and thus achieved an extended recognition effectiveness of the sonar system, which better distinguished the experimental shapes. The main objective of this paper is to calculate the effectiveness of identification using the ratio of parallelepipeds to cylinders and the average values of certain sequences of measurements. Further work targeted several navigation tasks on a real-time robot on an HCR base. Besides the implementation of map building with a relative degree of confidence, we performed experiments in rapid localization algorithm, which was developed simultaneously in our work group.

Keywords Shape recognition · Sonar signal · Features extraction · Signal analysis · Mobile robotics · Shape recognition · Sonar signal

1 Introduction

For all navigation tasks, sonar sensors are often used for their robustness and unambiguous acting in narrow and strictly-defined work conditions. To achieve good accuracy in range measurement, all at-tempts to identify objects using sonar have to deal with the following problems:

T. Dimitrova-Grekow (✉) · M. Jarczewski
Faculty of Computer Science, Bialystok University of Technology, Bialystok, Poland
e-mail: t.grekow@pb.edu.pl

M. Jarczewski
e-mail: m.jarczewski@o2.pl

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_13

- shape, size and material of which the object is made;
- position of the object and the distance to the sensor system;
- stability of objects at a time;
- temperature and humidity;
- structure of the environment.

One of the most popular ways to use ultrasonic sensors today are sonar arrays, containing several sensors. This compensates for some of the drawbacks to some extent. We present a complimentary attempt: in addition to the classical signal recovery and ridding of bad echoes, we sorted them and brought them into use. Because this paper is a continuation of our recent work, we have to shortly mention the previous work to explain the main idea of our approach. In our experiments, we tested shape distinguishing using simple objects, such as cylinders and parallelepipeds. A set of shape recognition rules was set and implemented in a real-time system: NXT robot. The tests have proved the offline assumptions and calculations. In our former experiments with a simple ultrasonic range measurement system of a mobile robot, we took an intuitive approach considering the parallelepiped to cylinder ratio [1].

In this paper, we present further development of the considerations, experiments, and results. Especially effective information delivered the average values from certain measure sequences, framed depending on the local circumstances, such as distance, level of certainty, etc. This is the base on which we ex-tended the effectiveness we obtained in our previous research.

Using the current study, we have the possibility to identify objects, as well as their exact orientation and collocations. Hence, building maps to a relative degree of confidence [2] is also a future aim.

2 Related Work

Sonar sequences contain much more information about the environment [3] compared to single measurements [4]. In all major branches of navigation, distance establishment and obstacle avoidance are the most popular uses of ultrasonic signals [5–8].

Sonar possesses some well-known advantages, such as its wide accessibility, relative accuracy and low costs. Its main disadvantage is its physical properties—the conical emission area and its interaction with the environment common for acoustic waves. That is the source of faked measurements occurring in the signal sequences. The more complex the structure of the environment, the more additional complicating and disturbing echoes arise. Thus, direct sonar data use becomes too unreliable. To compensate for all these disadvantages of real-time work of sonar-navigated systems, a powerful calculation engine is required for information processing. However, there are many examples of sonar use for mapping [9], localization [10], and SLAM [11], using different mathematical models and calculation power for real-time implementation.

The first information about the use of sonar shortcomings are described in [1, 12]. The article [1] discusses several features extracted from the sonar drawbacks. More information about its effectiveness is shown in [13]. The main objective of this paper is to calculate the effectiveness of identification using the ratio of parallelepipeds to cylinders and the average values of certain sequences of measurements. We present considerations about new features extraction aimed at further improvement of the recognition effectiveness of the sonar system.

3 System Structure

With this approach, we present a complimentary attempt: beside the classical signal recovery and ridding of bad echos, we sorted them and brought them into use. Because of the limited hardware, we tested shape distinguishing using simple objects, such as cylinders and parallelepipeds. Similarly as in our previous papers, an initial set of shape recognition rules was determined and implemented in a real-time system: NXT robot. The mobile robot was used as an independent data-collecting vehicle. So far, the tests have proved the offline assumptions and calculations.

The first stage of research involved data processing. The information gathered from the sonar system was treated in different ways while searching for the most relevant interpretation. The route the robot takes is straight, passing alongside the obstacles. The robot maintains a path parallel to the wall. Analyzing the collected data gathered from the sonar sensors, we noticed a number of interesting dependencies between the received reflections and the history of scanning. After the Data processing, a set of object features enabling effective object recognition emerged. Features extraction can be considered a separate second stage of the system structure. The third stage of the system was creating the Object recognition rules, which were the base for an effective, real-time object recognition (Fig. 1).

In all our experiments, we used a simple set of objects: two types of solids—parallelepipeds and cylinders. This choice was based on the principle difference of these figures and their popularity. We used several different sizes of each type throughout the experiments. The final recognition rules were applied to the real-time test vehicle.

In our initial tests, we achieved over 80 % accuracy in shape recognition. In this experiment, the results are about 5 % better on average than the previous ones. The experimental results are presented in the last section of this paper. The aim of further work is to implement mobile robot localization. A hybrid location method is being developed to test the pattern recognition approach in real-time conditions. Its main objective is to build a relative degree of confidence map to determine the vehicle location.

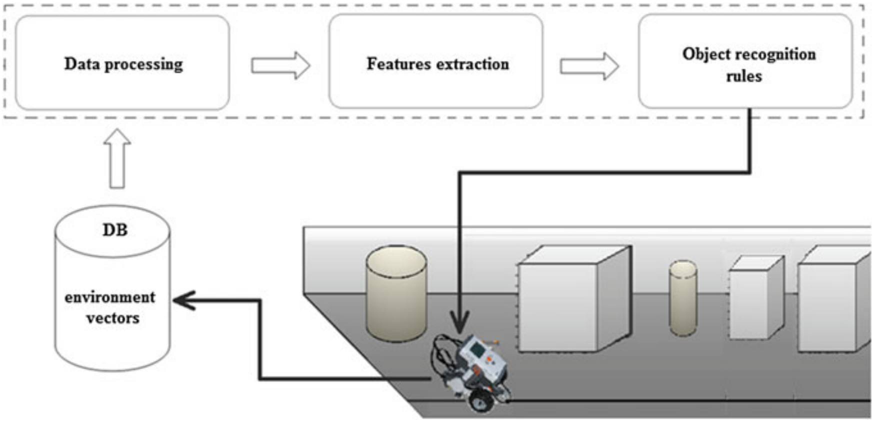

Fig. 1 Stages of the system

4 Shape Recognition Method

Although the most common approach to shape recognition is shape context consideration, based on complex data related directly to the analyzed objects, we developed a shape recognition method that also uses indirect data. We take into account 'bad' reflections, which we called 'glitters'. According to the system structure, described in the previous point, we took three steps to implement our approach: (1) data collection, (2) features extraction, (3) creation of shape recognition rules.

4.1 Data Collection

The data used in this research comes from an experimental mobile robot, traversing a certain distance and scanning the environment along its path (Fig. 1). Every file with data saved from the sonar was in-terpreted as a vector. The gathered data was double-processed: the raw sonar data was smoothed so that the registered shapes became sharper and more reliable; all data exceptions and irregularities were counted exactly, related, and compared, simultaneously.

When the data irregularities were removed, clear object shapes became visible (Fig. 2). The term 'glitter' stands for these measurements, which differ essentially from the contiguous ones. The 'glitters' are reading exceptions in a way. They are observed by comparison with neighboring ones. And although their quick smoothing facilitated further processing of the received signal, just they were the main source of information in our approach—all these irregularities were statistically assessed.

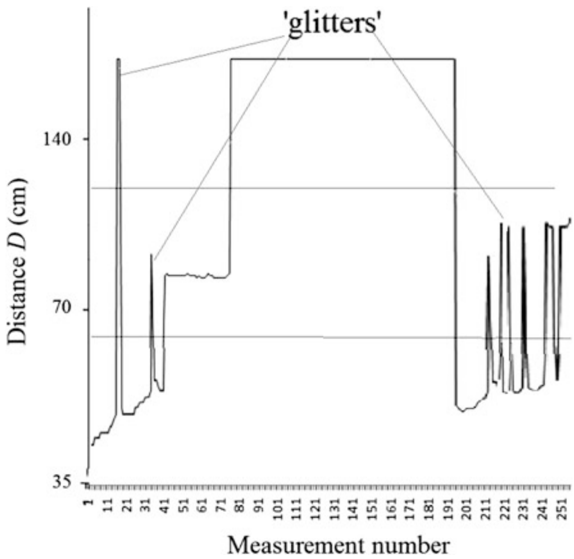

Fig. 2 Environment scanning—real objects

4.2 Features Extraction

The second step of our shape recognition method may be divided into two stages: initial and secondary features mining. The first one was based on regularly shaped data, obtained from double smoothed sonar sequences. The final one was focused on larger 'exceptions' and smoothing of the discovered shapes (Fig. 2). The 'glitters' were a result of overlapping echoes from various objects and measurements made in sequence, which can be especially useful in future development of the navigation system.

We based all considerations on the acoustic rules and simplification derived from the chosen work conditions. We primarily used a single object scene: the robot scanned and saved data from a single object and after many observations several important dependencies were found. The adjustment scheme was empirically determined during the experiments. The initial observation allowed to determine one of the most important features, used also in our previous works: the minimal measured distance.

Minimal measured distance: D_{MIN} This parameter is useful for detecting if it is an edge, side or cylinder. The minimal value itself is very important, but it becomes meaningful only after colligation with 10–30 contiguous distances.

Some of the most important selected characteristics in the second stage were described in our previous papers. Below is a short description:

Wall-greater distances: D_{OV}—percentage of measurements greater than the distance to the wall N_W.

Alien echo: *DNR*—percentage of measured distances less than NW although no object is displaced at the spot.

The new features introduced in this paper are relative distribution and curve probability. Both have a tendency to compensate for unsolved problems with distinguishing the narrow cylinders from the parallelepipeds standing orthogonal to the sonars beam.

Relative distribution: R_{DS}—percentage of measurements greater than the shortest currently measured distance plus a tolerance of 10–25 %. The value of R_{DS} is calculated as follows:

$$R_{DS} = \frac{D_{MIN}}{N} k \tag{1}$$

where:

DMIN	minimal measured distance,
K	correction of applied tolerance,
N	total number of measurements

Initially, we disregarded this feature, however, additional data analyses, including an artificial intelligence tool support, the program Weka, pointed toward a classical attempt. And this paid off. We achieved a simple and effective improvement of the recognition effectiveness we had obtained before. An important element in this approach is the frame in which this parameter is calculated. This point is still being explored.

Curve probability: P_C—a probability, calculated based on relative distribution R_{DS} and the wall-greater distance D_{OV}:

$$P_C = \frac{D_{OV}}{R_{DS}} \rho \tag{2}$$

where:

ρ correction of applied tolerance

This feature is used as an additional parameter in cases when the qualification function generates an ambiguous value.

4.3 Shape Recognition Rules

We explored several dependencies from the features described above. The function DNR = f1(DOV) was built and described in detail before. The DNR = f2(RDS) characteristic for ca. several hundred vectors is presented here to be compared with

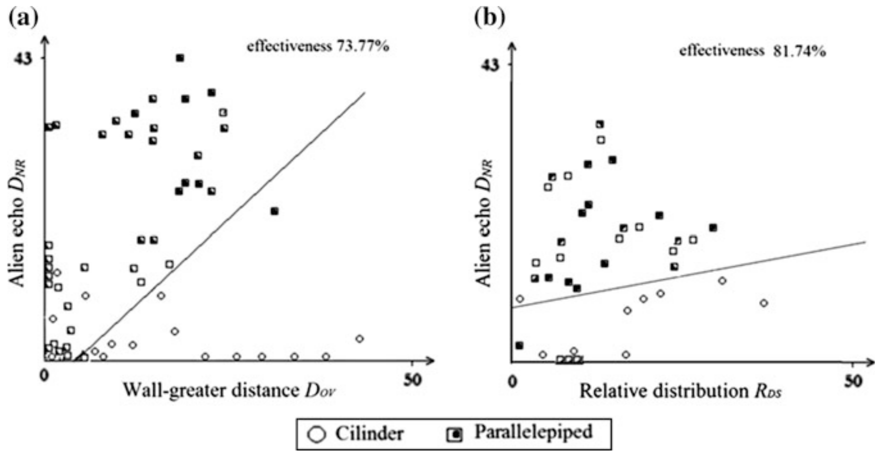

Fig. 3 Distinguishing parallelepipeds and cylinders

this between alien echo and wall-greater distance. Linear dependences, which were implemented and loaded on the robot for online tests, resulted from the characteristic families (Fig. 3). After choosing the function, we created and applied a classifier. We used the concept of similarity—the simplest and most common approach—known as 'template matching'. This allowed us to discriminate parallelepipeds and cylinders with minimum calculations. An optimization of these observations is proof of the simple reliance between DOV, RDS and DNR.

A comparative illustration of both dependencies DNR = f1(DOV) and DNR = f2 (RDS) are shown in Fig. 3. The optimal clustering generated a linear function that was different for every group of objects. In fact, a rule generalizing these differences can be found. Objects clustering was determined by linear functions F1CL (x) = a1x + b1 and F2CL(x) = a2x + b2. Points located on the same side of the line belong to the same cluster. The experiments showed that there is no universal linear characteristic fulfilling the accuracy requirements for the total range of various distances (in meaning, minimal distance robot-object). We retrieve a bunch of linear functions (rules), allowing to select the optimal line. This optimization is a balance between maximal accuracy for a specific, narrow distance region on the one hand, and complex precision for a wide one on the other. After normalizing, all parameters got integer values from 0 to 100. Thus, we assumed that if the distance between two points is greater than 1, it is a sufficient condition to set an optimal line.

For all dependencies, finding the best function includes three steps:

1. Defining points at a certain distance from each other on the x and y axes.
2. For each point a dozen simple lines, differing by a factor 2 are determined.
3. By evaluating the test collection, the best of the lines is selected.

5 Estimation of the Achieved Identification Effectiveness

In this study, we used various orientations of the parallelepipeds; the difference in the number of scanned cylinders and parallelepipeds was significant. In order to ensure objective evaluation, factor S was introduced, depending on the total number of cylinders C, and the total number of parallelepipeds P. Thus we obtain:

$$S = \frac{P}{C} \tag{3}$$

We assumed R to stand for effectiveness of identification and it was calculated as a ratio of the quantity of parallelepipeds to cylinders on the same side of the line. Indexes O stand for objects over the line and indexes U for those under. Then two effectiveness values concerning all objects under and all objects above the line, respectively, could be calculated:

$$R_U = \frac{P_U}{(P_U + C_U S)} \tag{4}$$

$$R_O = \frac{C_O S}{(C_O S + P_O)} \tag{5}$$

Generalizing, the complex effectiveness we get R1:

$$R_1 = \frac{R_U(C_U S + P_U) + R_O(C_O S + P_O)}{C + P} 100 \tag{6}$$

Later, we can make the opposite assumptions that under the line there are cylinders and over parallelepipeds:

$$R_2 = 1 - R_1 \tag{7}$$

And finally the optimal value can be chosen:

$$R = \max(R_1; R_2) \tag{8}$$

Repeating this procedure for all lines, we find the best clustering existing for a particular experiment.

6 Experiments and End Result Assessment

As before, we found that there is no ideal solution, satisfying all environment configurations. Our approach had almost 100 % effectiveness if the objects were situated on the same distance from the robot path with a tolerance of several

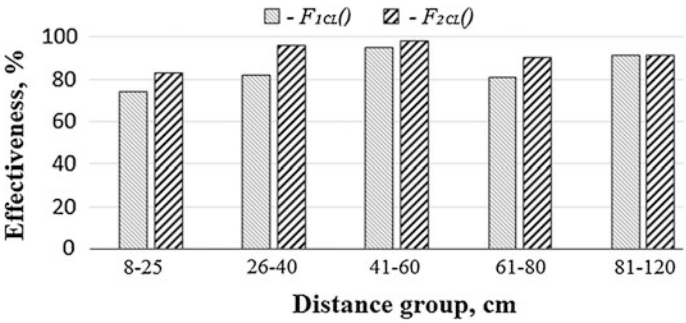

Fig. 4 Effectiveness for the distance groups

centimeters. Accuracy fell dramatically if the analyzed data came from objects standing at all possible distances from the robot's path. Thus, the most appropriate way to obtain satisfactory shape recognition requires splitting the distances into groups as well as collecting and retrieving data separately for each group. Such range clustering was determined during the hundreds of tests, considering the specific effectiveness for separate distance fractions and the complex effectiveness for the entire experiment. An optimal division of the distance ranges is 20 cm areas, except the first and the last distance region: the first is a bit shorter—starts from 8 cm instead of from 0 and the last is doubled. So our total range was from 8 to 120 cm.

We conducted a teaching set of tests to obtain proper clustering for the objects, and a testing one to evaluate the results.

This experiment, consisting of many hundreds tests, showed an improvement in accuracy. Considering that for the largest distances (120 cm), most of the test objects were hardly 'visible' and D_{OV}, R_{DS} and D_{NR} were almost the same, the relative distribution R_{DS} showed an significant approve of the recognition, especially for distances under 100 cm. The lowest effectiveness of the clustering function $F_{1CL}()$ 74.05 % for the first distance group of objects increased to 83.74 % thanks to the new parameter R_{DS} and the clustering function $F_{2CL}()$ (Fig. 4). The best effectiveness for $F_{1CL}()$ 96.15 % was obtained in the third group 41–60 cm and improved for $F_{2CL}()$ to 97.21 %. We obtained maximum improvement for the second group of objects. The new recognition effectiveness reached 95.3 %, which is 15 % better than before (81.32 %).

7 Conclusions

The main objective of this paper is to calculate the effectiveness of identification using the ratio of parallelepipeds to cylinders and the average values of certain sequences of measurements.

We combined the regular information obtained from a simple sonar system with information derived from measurement abnormalities and thus achieved an extended recognition effectiveness of the sonar system, which better distinguished the experimental shapes. This original approach minimizes the calculations for real-time implementation. We presented and compared shape recognition rules: linear functions $F_{1CL}()$ and $F_{2CL}()$ expressing the dependencies between the sonar signal features in the described conditions. The better average accuracy we achieved was 91.6 % for $F_{2CL}()$. Tests on the robot Lego Mindstoms NXT proved the approved effectiveness of both functions $F_{1CL}()$ and $F_{2CL}()$, which considered relative distribution of the values from the objects area.

We aim to find general rules for shape and position recognition. Further work targeted several navigation tasks on a real-time robot on an HCR base. Besides the implementation of map building with a relative degree of confidence, we performed experiments in rapid localization algorithm, which was developed simultaneously in our work group.

Acknowledgments This paper is supported by the S/WI/1/2013.

References

1. Dimitrova-Grekow, T., Jarczeweski, M.: Sonar method of distinguishing objects based on reflected signal specifics. In: LNCS, vol. 8502, pp 506–511. Springer, Berlin (2014)
2. Dimitrova-Grekow T.: A hybrid algorithm for self-location. In: PAK, vol. 11, pp. 1163–1166 (2013)
3. Ohtani, K., Baba, M.: Shape recognition for transparent objects using ultrasonic sensor array. In: SICE, Annual Conference, pp. 1813–1818 (2007)
4. Barshan, B., Kuc, R.: A bat-like sonar system for obstacle localization. Pattern Anal. Mach. Intell. **12**, 686–690 (1992)
5. Walter, S.: The sonar ring: obstacle detection for a mobile robot. Robot. Autom. IEEE J. **4**, 1574–1579 (1987)
6. Lenser, S., Veloso, M.: Visual sonar: fast obstacle avoidance using monocular vision. Intell Robot. Syst **1**, 886–891 (2003)
7. Del Castilloa, G., Skaara, S., Cardenasb, A., Fehr, L.: A sonar approach to obstacle detection for a vision-based autonomous wheelchair. Robot. Autom. IEEE J. **54**, 967–981 (2006)
8. Kiyoshi, O., Masamichi, M., Hiroyuki, T., Keihachiro, T.: Obstacle arrangement detection using multichannel ultrasonic sonar for indoor mobile robots. J. Artif. Life Robot. **15**, 229–233 (2010)
9. Dimitrova-Grekow, T., Zach, M.: Topological-metric indoor path planning. In: PAK Pomiary, Automatyka, Kontrola, vol. 58, pp. 611–618 (2012)
10. Fromberger, L.: Representing and selecting landmarks in autonomous learning of robot navigation. In: Xiong et al. C. (ed.) ICIRA Part I, LNAI vol. 5314, pp. 488–497. Springer, Berlin (2008)
11. Yap, TN., Shelton, CR.: SLAM in large indoor environments with low-cost, noisy, and sparse sonars. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 1395–401 (2009)

12. Joong-Tae, P., Jae-Bok, S., Se-Jin Lee, M.K.: Sonar sensor-based efficient exploration method using sonar salient features and several gains. J. Intell. Robot. Syst. **63**, 465–480 (2011)
13. Dimitrova-Grekow, T., Jarczeweski, M.: Identification effectiveness of the shape recognition method based on sonar. In: LNCS, vol. 9339, pp. 244–254, Springer, Berlin (2015)

Experimental Examination of Facilities Layout Problems in Logistics Systems Including Objects with Diverse Sizes and Shapes

Jerzy Grobelny and Rafał Michalski

Abstract This study demonstrates a simulation experiment results regarding a flexible approach to solving facilities layout problem on a regular grid. In this paper approach we model objects with different dimensions and various shapes. We have implemented our version of the simulated annealing algorithm [5] and analyzed how the number of cycles (100, 200, 300), objects types (uniform, diverse) affect average goal function values, mean concentration degrees, and the number of disintegrated objects. The formal statistical analysis was conducted separately for two sizes of regular grids, i.e. 6 × 6 and 10 × 10. The outcomes generally showed a significant influence of the studied effects on the analyzed dependent variables.

Keywords Logistics · Facilities layout problem · Simulated annealing · Flexible approach · Simulation experiment

1 Introduction

Since the facilities layout (FL) problem is significant in various areas of operations research, it has been subject to numerous studies throughout the years. The problem may be formulated as finding such an optimal arrangement of objects (e.g. machines or plants) that transport costs are minimized. Sahni and Gonzalez [10] demonstrated that FL belongs to at least NP-complete class of problems, thus it is not possible to find the optimal solution in polynomial time. Extensive descriptions

J. Grobelny · R. Michalski (✉)
Faculty of Computer Science and Management,
Wrocław University of Technology, Wrocław, Poland
e-mail: rafal.michalski@pwr.edu.pl
URL: http://RafalMichalski.com

J. Grobelny
e-mail: jerzy.grobelny@pwr.edu.pl
URL: http://JerzyGrobelny.com

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*,
Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_14

of available methods for solving FL problems along with their typology are available, for instance, in Kusiak and Hergau [7] or Singh and Sharma [11].

Many of the classical algorithms require some restrictions in modeling real situations. For instance it is very popular to use regular grids as models of space available for designing layouts. In such cases, possible objects locations are predefined and it is usually assumed that objects have square shapes with the same dimensions. These requirements were naturally considered as drawbacks and as early as in the 1970s Armour and Buffa [1] proposed an algorithm which allowed for including objects with various sizes. In their method, objects could consist of many unitary building blocks corresponding to individual grid cells. Thanks to that it was also possible to define objects with diverse shapes. The problem of finding optimal shapes was subsequently analyzed by Kim and Kim [4]. More flexible approaches to modeling objects with different sizes and shapes were also developed in a number of genetic algorithms. For example the concept of flexible bays [13, 14] or slicing tree methods [8, 12].

In the present study we continue research in examining a flexible approach to objects dimensions and sizes located on a regular grid. Unlike Armour and Buffa [1] or Bazaraa [2] we propose an application of simulated annealing concept to specifically defined relationships where only zero-one values are used for describing associations between objects. Additionally, in contrast to Bazaraa [2] or CRAFT [3] algorithms in this proposal we do not put any restrictions on objects' shapes. Each object consists of the appropriate number of unitary building blocks. One of these components represents always the input—output place while the rest items are linked with it. This relationship value amounts to one.

To test our approach we designed and conducted a simulation experiment which was focused on relatively simple FL problems on a regular grid. The methods used and all the obtained results are presented and thoroughly analyzed in the following sections.

2 Methods

2.1 Algorithm Implementation

We implemented a Simulated Annealing algorithm following the general recommendations provided by Kirkpatrick et al. [5] to produce reasonable solutions on a regular grid. Some other modifications of the Simulated Annealing algorithm may be found in Meller and Bozer [9] or recently in Kulturel-Konak and Konak [6]. This study approach resembles the physical process of metal particles' movements during the annealing process. In our facility layout context, these movements correspond to exchanging objects' components places. The whole process evolves along with decreasing the temperature (T). There also exists a possibility of accepting worse solutions with a predefined probability. The number of objects' components pair changes in a specific temperature is proportional to the number of objects' item problem $k \times N$, where k amounted to 10 in this study. The temperature changes are calculated as follows: $T_j = r \times T_j - 1$, with r = 0.9. The stop criterion determined by this relation $T_f = 0.9(i - 1) \times T_i$, where i = 100 is the number of steps.

2.2 Dependent Measures

The classical goal function formula was used as a dependent measure in our simulation experiments:

$$\text{Euclidean based goal function (OF)} = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(D_{p(i)p(j)} \cdot L_{ij}\right), \tag{1}$$

where $D_{p(i)p(j)}$ is the standard Euclidean distance obtained according to the following formula:

$$D_{p(i)(j)} = \left(\sum_{k=1}^{N} \left| x_{p(i)k} - x_{p(j)k} \right|^2 \right)^{1/2}, \tag{2}$$

with k denoting the dimension—here equal two, and L_{ij} specifying the relationship strength between two objects.

2.3 Experimental Design and Procedure

In our simulation experiments we confined to three variates, that is (i) the number of objects' components arranged in a square regular grid (6 × 6 = 36 and 10 × 10 = 100), (ii) the type of objects, that is their complexity: either all included the same number of items (six 6-item objects for the 6 × 6 grid and twenty objects consisting of 5 items each for the 10 × 10 grid) or included a diverse number of components (for the 6 × 6 grid: 4 objects with 6 items and 4 objects with 3 components; for the 10 × 10 grid: 10 objects with three components and ten objects with 7 elements), and finally (iii) the number of cycles applied in the SA algorithm (100, 200, 200).

We employed a full experimental design, which produced 12 different simulation conditions: *Grid size* (2) × *Objects type* (2) × *No of SA Cycles* (3), and replicated the simulation 100 times for each of the 12 conditions. We were interested not only in minimizing goal function values but also in how the objects items behave in terms of their shape and possible objects' disintegrations. Therefore, we analyzed both the Euclidean based goal function as well as the average concentration degree of the examined objects and a number of objects that were divided into two or more parts. The concentration degree was computed by dividing the bigger by the smaller object dimension. In this way, the biggest possible concentration degree amounted to 1 when the objects' items formed a square whereas smaller values indicated how far a specific shape was from being a square.

The distance between all neighboring objects' items was set at one and we assumed that each link between objects' components amounted to one.

3 Results and Discussion

3.1 *Basic Descriptive Statistics*

For all three dependent variables and all 12 experimental conditions described in a previous section, we computed the minimum, mean, and mean standard error values. These basic descriptive statistics are put together in Table 1.

It may be observed that the obtained minimal values for the Euclidean based goal function for the 6 × 6 grid actually do not depend on the number of cycles applied in a SA algorithm which is not the case for the 10 × 10 grid. On the other hand the average values seem to decrease significantly along with the increase of the number of cycles used. For the concentration degree there seems to be a clear pattern visible only for the larger grid: the mean concentration increased for the bigger number of cycles.

Table 1 Basic descriptive statistics for all experimental conditions and all three dependent variables analyzed in the current study

Condition	Euclidean based goal function			Average concentration degree			Number of disintegrated objects		
	Min	Mean	[a]MSE	Min	Mean	[a]MSE	Min	Mean	[a]MSE
06 × 06-U-100	40.7	44.7	0.18	0.62	0.77	0.0068	0	0	0
06 × 06-U-200	40.7	43.7	0.19	0.60	0.78	0.0076	0	0	0
06 × 06-U-300	40.7	43.4	0.16	0.62	0.79	0.0074	0	0.02	0.020
06 × 06-D-100	42.0	46.2	0.17	0.50	0.79	0.0097	0	0.01	0.010
06 × 06-D-200	41.3	45.4	0.18	0.50	0.78	0.010	0	0.01	0.010
06 × 06-D-300	41.3	45.2	0.16	0.50	0.78	0.011	0	0.02	0.014
10 × 10-U-100	146	164	0.73	0.59	0.72	0.0043	0	2.96	0.160
10 × 10-U-200	139	153	0.70	0.62	0.73	0.0043	0	0.74	0.086
10 × 10-U-300	133	147	0.71	0.65	0.74	0.0036	0	0.28	0.049
10 × 10-D-100	151	165	0.69	0.59	0.76	0.0050	0	3.58	0.157
10 × 10-D-200	135	153	0.66	0.65	0.78	0.0053	0	1.57	0.078
10 × 10-D-300	135	148	0.68	0.64	0.80	0.0056	0	1.19	0.058

[a]*MSE* mean standard error, *U* uniform and *D* diverse no of objects' components

The presented data also suggest that for the larger grid, the mean number of disintegrated objects was decidedly smaller when the bigger number of SA cycles was employed. Moreover, it is worth noting that it was possible to obtain at least one solution where none of the objects was parted for all examined experimental conditions.

3.2 Analysis of Variance

The described observations in a previous section are formally verified by means of analyses of variance. Since there was a significant difference between goal function values for 6 × 6 and 10 × 10 grids, a series of two-way analyses of variance were applied separately to check if the examined factors are statistically.

Analysis of variance for the 6 × 6 arrays of objects' components. The obtained ANOVA results for the Euclidean based goal function are summarized in Table 2. They revealed that both examined factors, that is objects type and the number of cycles significantly influenced the mean goal function. The interaction between these effects was irrelevant. The mean goal function values for both factors are illustrated in Figs. 1 and 2.

Figure 1 shows that mean goal functions were significantly smaller than for diverse objects. The results presented in Fig. 2 formally support the outcomes of the descriptive statistics analysis.

The average concentration degree of objects for 6 × 6 arrays occurred to be significantly influenced ($\alpha = 0.01$) neither by the examined factors nor by their interaction. A similar situation was observed for the number of disintegrated objects where the tested effects and their interaction were not statistically meaningful ($\alpha = 0.01$).

Analysis of variance for the 10 × 10 arrays of objects' components. The results of 10 × 10 array ANOVA are given in Table 3 and show that the mean goal function was significantly affected only by the number of SA cycles. The meaningful relation is presented in Fig. 3.

Table 2 Two-way ANOVA regarding the influence of objects' type (uniform versus diverse number of object's components) and the number of SA cycles (100, 200, 300) on the goal function for 6 × 6 arrays of objects

Effect	SS	df	MS	F	p	η^2
Objects type (O)	408	1	408	137	<0.0001*	0.188
Cycles (C)	153	2	77	25.8	<0.0001*	0.0799
O × C	3.12	2	1.56	0.526	0.591	0.00177
Error	1764	594	2.97			

*$p < 0.05$; *df* degrees of freedom; *SS* sum of squares; *MS* mean sum of squares; η^2 partial eta-squared

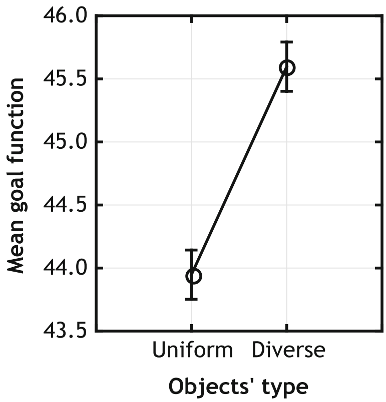

Fig. 1 The influence of objects' type (uniform versus diverse number of object's components) on the Euclidean based goal function for 6 × 6 arrays. $F(1, 594) = 137$, $p < 0.0001$. *Vertical bars* denote 0.95 confidence intervals

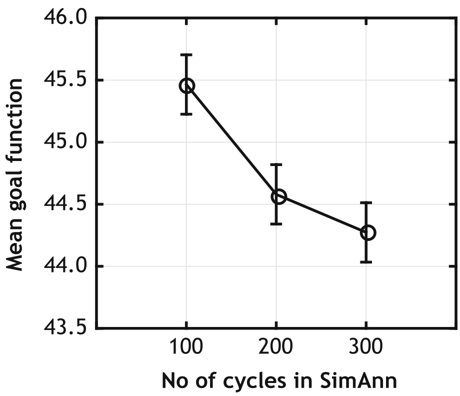

Fig. 2 The influence of the number of SimAnn cycles (100, 200, 300) on the Euclidean based goal function for 6 × 6 arrays. $F(2, 594) = 22.8$, $p < 0.0001$. *Vertical bars* denote 0.95 confidence intervals

Table 3 Two-way ANOVA regarding the influence of objects' type and the number of SA cycles on the goal function for 10 × 10 arrays

Effect	SS	df	MS	F	*p*	η^2
Objects type (O)	64.3	1	64.3	1.33	0.249	0.00224
Cycles (C)	30,523	2	15,262	316	<0.0001*	0.516
O × C	4.12	2	2.06	0.0426	0.958	0.000144
Error	28,671	594	48.3			

$^*p < 0.05$; *df* degrees of freedom; *SS* sum of squares; *MS* mean sum of squares; η^2 partial eta-squared

Another two-way ANOVA was applied to check whether examined factors affect the average concentration degree. The outcomes are demonstrated in Table 4; Figs. 4, 5 and 6 present a significant influence of both factors and their interaction.

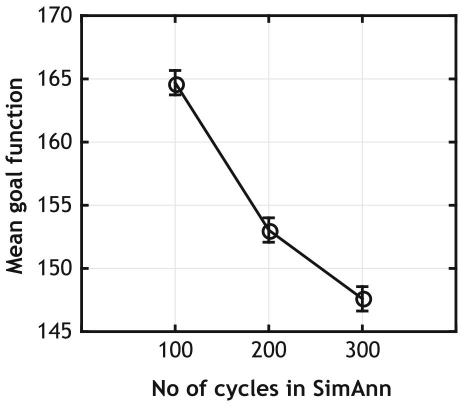

Fig. 3 The influence of the number of SA cycles on the goal function for 10 × 10 arrays. $F(2, 594) = 316$, $p < 0.0001$. *Vertical bars* denote 0.95 confidence intervals

Table 4 Two-way ANOVA regarding the influence of objects' type and the number of SA cycles on the average concentration degree for 10 × 10 arrays

Effect	SS	df	MS	F	p	η^2
Objects type (O)	0.370	1	0.370	166	<0.0001*	0.218
Cycles (C)	0.0807	2	0.0404	18.1	<0.0001*	0.0574
O × C	0.0137	2	0.0069	3.07	0.0469*	0.0102
Error	1.33	594	0.00223			

*$p < 0.05$; *df* degrees of freedom; *SS* sum of squares; *MS* mean sum of squares; η^2 partial eta-squared

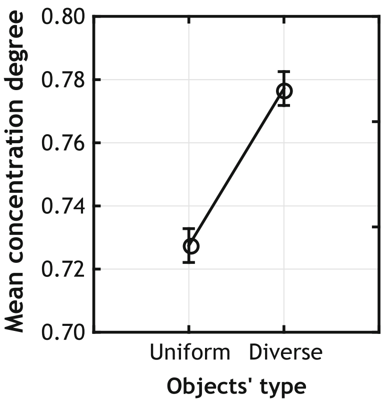

Fig. 4 The influence of objects' type on the mean concentration degree for 10 × 10 arrays. $F(1, 594) = 166$, $p < 0.0001$. *Vertical bars* denote 0.95 confidence intervals

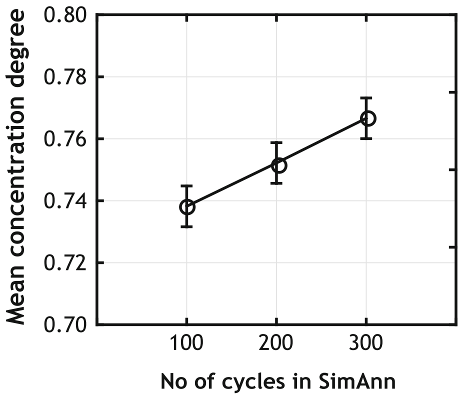

Fig. 5 The influence of the number of SA cycles on the mean concentration degree for 10 × 10 arrays. $F(2, 594) = 18.1$, $p < 0.0001$. *Vertical bars* denote 0.95 confidence intervals

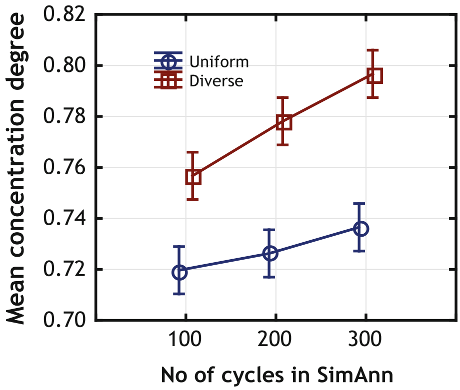

Fig. 6 The influence of the analyzed factors interaction on the mean concentration degree for 10 × 10 arrays. $F(2, 594) = 3.07$, $p < 0.0469$. *Vertical bars* denote 0.95 confidence intervals

Finally, another ANOVA was employed to verify if the investigated effects significantly influenced the number of disintegrated objects. The results are provided in Table 5 and illustrated in Figs. 7 and 8. The analysis revealed that both the type of objects as well as the number of SA cycles significantly differentiated the number of disintegrated objects.

Table 5 Two-way ANOVA regarding the influence of objects' type and the number of SA cycles on the number of disintegrated objects arranged in a 10 × 10 array

Effect	SS	df	MS	F	*p*	η^2
Objects type (O)	92.8	1	92.8	80.0	<0.0001	0.119
Cycles (C)	738	2	369	318	<0.0001	0.517
O × C	2.24	2	1.12	0.966	0.381	0.00324
Error	690	594	1.16			

$^*p < 0.05$; *df* degrees of freedom; *SS* sum of squares; *MS* mean sum of squares; η^2 partial eta-squared

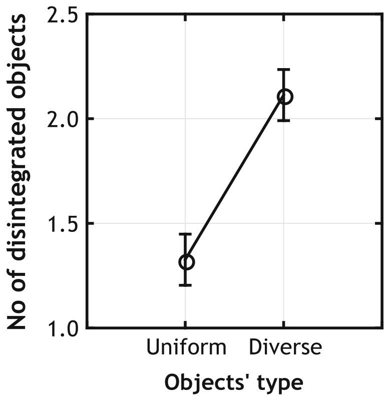

Fig. 7 The influence of objects' type on the number of disintegrated objects for 10 × 10 arrays. $F(1, 594) = 80$, $p < 0.0001$. *Vertical bars* denote 0.95 confidence intervals

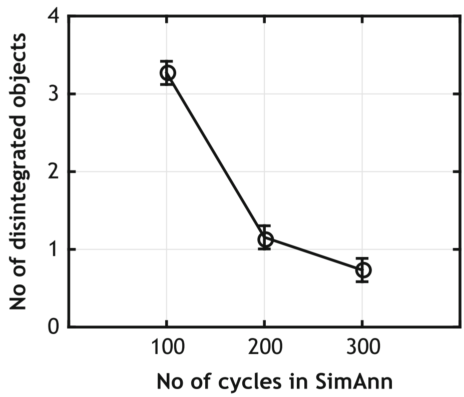

Fig. 8 The influence of the number of SA cycles on the number of disintegrated objects for 10 × 10 arrays. $F(2, 594) = 318$, $p < 0.0001$. *Vertical bars* denote 0.95 confidence intervals

4 Discussion and Conclusions

The liberalization of restrictions regarding objects' shapes is the crucial element of the presented approach. Every object is constructed from uniform, modular building blocks of the size 1 × 1. The relationships used in our experiment refer to the necessity of the given pair of objects to be located next to each other. The zero value means that such a requirement is not needed while one denotes that their proximity is essential.

Since links between whole objects and between objects' components are treated exactly the same, the total value of the goal function may be interpreted as an estimated joint cost of transportation between departments and within them, that is, between the input-output place and other objects' items representing single workstations inside departments.

The lack of other constraints may, however, result in disintegration of departments (objects) into smaller, not necessarily adjacent fragments. Still, the conducted simulation experiments showed that in each of the experimental conditions it was possible to obtain at least one solution where none of the objects was separated. It also seems that solutions with a small degree of objects' disintegration may be interesting for logistics systems designers. First, one may ponder whether the disintegration proposed by the algorithm could be a better solution and lead to lowering the transport costs despite dividing or even doubling some of the departments. Furthermore, some of the modern transportation systems (e.g. gantry robots) do not require a physical neighborhood of the cooperating workstations.

The results regarding the departments' concentration degree coefficients (measured by a ratio of the shorter side to the longer one) demonstrated that for the FL problems with binary relationships the increase of the number of cycles in the simulation annealing algorithm generates more condensed solutions. Additionally, these condensed designs provide more optimal results which suggests that objects' shapes obtained in the presented method may significantly influence the goal function value.

The promising findings presented in this paper point out potential directions of research and developments of the current study approach. It seems that the most interesting would be the examination of the algorithms' behavior in various types of transportation systems inside departments and investigations of diverse inter- and intradeparmental relationships and levels of transportation costs. In particular, the examination these parameters influence on the proposed departments/objects concentration and disintegration coefficients.

Acknowledgments The work was partially financially supported by the Polish National Science Center grant no. 2011/03/B/HS4/04532.

References

1. Armour, G.C., Buffa, E.S.: A heuristic algorithm and simulation approach to relative location of facilities. Manage. Sci. **9**(2), 294–309 (1963). doi:10.1287/mnsc.9.2.294
2. Bazaraa, M.S.: Computerized layout design: a branch and bound approach. AIIE Trans. **7**(4), 432–438 (1975). doi:10.1080/05695557508975028
3. Buffa, E.S., Armour, G.C., Vollmann, T.E.: Allocating facilities with CRAFT. Harvard Bus. Rev. **42**(2), 136–158 (1964)
4. Kim, J.-Y., Kim, Y.-D.: Graph theoretic heuristics for unequal-sized facility layout problems. Omega **23**(4), 391–401 (1995). doi:10.1016/0305-0483(95)00016-H
5. Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science **220** (4598), 671–680 (1983). doi:10.1126/science.220.4598.671
6. Kulturel-Konak, S., Konak, A.: A large-scale hybrid simulated annealing algorithm for cyclic facility layout problems. Eng. Optim. **47**(7), 963–978 (2015). doi:10.1080/0305215X.2014.933825
7. Kusiak, A., Heragu, S.S.: The facility layout problem. Eur. J. Oper. Res. **29**(3), 229–251 (1987). doi:10.1016/0377-2217(87)90238-4
8. Liu, X., Sun, X.: A multi-improved genetic algorithm for facility layout optimisation based on slicing tree. Int. J. Prod. Res. **50**(18), 5173–5180 (2012). doi:10.1080/00207543.2011.654011
9. Meller, R.D., Bozer, Y.A.: A new simulated annealing algorithm for the facility layout problem. Int. J. Prod. Res. **34**(6), 1675–1692 (1996). doi:10.1080/00207549608904990

10. Sahni, S., Gonzalez, T.: P-complete approximation problems. J. ACM **23**(3), 555–565 (1976). doi:10.1145/321958.321975
11. Singh, S.P., Sharma, R.R.K.: A review of different approaches to the facility layout problems. Int. J. Adv. Manuf. Technol. **30**(5–6), 425–433 (2006). doi:10.1007/s00170-005-0087-9
12. Tam, K.Y.: A simulated annealing algorithm for allocating space to manufacturing cells. Int. J. Prod. Res. **30**(1), 63–87 (1992). doi:10.1080/00207549208942878
13. Tate, D.M., Smith, A.E.: Unequal-area facility layout by genetic search. IIE Trans. **27**(4), 465–472 (1995). doi:10.1080/07408179508936763
14. Tong, X.: SECOT: a sequential construction technique for facility design. Unpublished Doctoral Dissertation, Department of Industrial Engineering, University of Pittsburgh (1991)

A Concept of a Flexible Approach to the Facilities Layout Problems in Logistics Systems

Jerzy Grobelny and Rafał Michalski

Abstract This paper describes a concept of computer software devoted to flexible modeling and solving facilities layout (FL) problems in logistics systems. In the beginning, a brief review of available approaches in this regard is provided. Next, diverse aspects of flexibility in designing FL are identified in the literature and discussed. These considerations constitute a background for the presentation of the system modules.

Keywords Facilities layout · Logistics · Flexible approach · Ergonomics

1 Introduction

The main goal concerned with finding optimal arrangements of related objects is to minimize costs of their mutual functioning. In the area of industrial engineering one usually searches for such a layout of machines and tools in the production space that ensures minimal costs of material flows between analyzed objects.

From the mathematical modeling point of view this optimization is a combinatorial task belonging, at least, to the NP-Complete class of problems [32]. This means that there is no efficient method of finding an optimal solution for big problems including more than twenty objects. Hence, a major trend in this area involves developing heuristic procedures for finding decent suboptimal solutions relatively quickly. In the literature one can find dozens of such proposals. Extensive reviews and classifications of optimization algorithms and systems are provided, for

J. Grobelny · R. Michalski (✉)
Faculty of Computer Science and Management,
Wrocław University of Technology, Wrocław, Poland
e-mail: rafal.michalski@pwr.edu.pl
URL: http://RafalMichalski.com

J. Grobelny
e-mail: jerzy.grobelny@pwr.edu.pl
URL: http://JerzyGrobelny.com

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_15

instance, in the works of Kusiak and Heragu [24], Meller and Gau [28], Singh and Sharma [36], Drira et al. [7].

All methods of searching FLP solutions may be generally divided into two groups i.e. classical and artificial intelligence based approaches. Among the first group methods one can distinguish optimal and suboptimal algorithms called also heuristics. Artificial intelligence based tools usually include: metaheuristics, expert systems, artificial neural networks, genetic and fuzzy logic algorithms.

The fundamental advantage of optimal algorithms is their ability to produce best possible solution every time they are applied. The most popular in this group are methods based on a branch and bound approach which was originally proposed by Gilmor [9] and then Lawler [25]. Despite being able to find the best solution they also have a generic drawback resulting from the NP-Complete nature of the problem under consideration. Since they would not provide optimal solutions in a reasonable time for problems with larger than twenty objects, therefore approximate algorithms of the heuristic character are more practical. They include construction, improvement (neighborhood search) and hybrid algorithms [35].

Creating a solution from scratch by specifying step by step the consecutive locations of objects according to the given rule is a characteristic feature of construction methods. The procedure is finished when all items are assigned to appropriate places. As construction algorithms may lead to weak solutions [41], they are especially useful for generating initial solutions for more efficient algorithms. Among the most known construction algorithms there are ALDEP [34], CORELAP [26], HC66 [19], PLANET [5], MAT [8], FATE [2], FLAT [18]. CORELAP has become very popular and it has been implemented many times in other computer programs such as those proposed by Parsaei and Galbiati [30], Ziai and Sule [42], or Chen and Kengskool [4].

Improvement algorithms called also neighborhood search algorithms constitute the next group of heuristics. The specificity of these methods is a generation of an initial solution and then making improvements of such a layout by making beneficial locations changes of adjoining objects. The elaborated by Buffa et al. [3] CRAFT method was one of the first improvement algorithms for solving FL problems. To this day, it is among the most respected optimization algorithms. It was repeatedly modified and the following methods were based on its original idea: COFAD [39], CRAHT-3D [20], or SPACECRAFT [21].

Classical algorithms are usually based on assumptions that make it possible to model the reality in a digital computer world. Unfortunately, they introduce significant constrains that simplify the real problems. The vast majority of algorithmic approaches take advantage of regular grids that model the space for the layout design. For instance, methods based on pairwise changes of objects' locations require strict specifications of available spaces in such grids (e.g. CRAFT or COFAD).

2 Various Aspects of a Flexible Approaches to Facilities Layouts

Early, classical approaches to the facilities layout problem proposed regular algorithms with specified constraints imposed on input data and searching for deterministic, repetitive solutions for a specific data types. Very soon, however, works began on weakening the restrictions and making the FL models more flexible. Particular importance was gained by allowing the mathematical models to analyze different objects' dimensions and shapes as well as applying various goal functions and input data. Developments in these aspects are described in the subsections that follow.

2.1 Diverse Objects' Dimensions and Shapes

Despite restrictions caused by using regular grids for modeling FA problems, already Armour and Buffa [1] proposed to include objects having various sizes into their algorithm. The authors assumed that objects may be constructed by a different number of uniform bricks that were defined as regular grid cells of 1x1 dimensions. In this way it was possible to analyze more flexibly the objects sizes and even search for optimal objects shapes. The latter idea was implemented by Kim and Kim [23] in the construction heuristic named SHAPE [17].

The need for treating the FL problems in a more flexible way and allowing for representing objects of diverse dimensions embodied also in multiple approaches based on genetic algorithms. For example, the concept of flexible bays [38, 40] provides solutions in a form of rectangular objects arranged in bays of various widths. In turn, the slicing tree approach [27, 37] allows for optimizing objects as rectangles having individually different dimensions.

2.2 Flexible Approach to Defining Goal Functions and Input Data

Classical FL approaches mainly use objective criteria based on physical units like costs in monetary units, objects transport distances, man-meters etc. An interesting trend of research was initiated in the 70s of the last century, inter alia, by Scriabin and Vergin [33]. They drew attention to the human expert capabilities of solving facilities layout problems. Generally, investigations conducted in this direction demonstrated that intuition and experience help experts in achieving good layout designs in numerous situations.

These outcomes probably acted as an impulse to seek methods based on soft approaches to modeling and solving FL problems. An artificial intelligence manifesting itself, among other things, by applying fuzzy logic allowed for taking advantage of experts' knowledge in a form close to the natural language. As a starting point of modeling in such a perspective one may consider linguistic patterns proposals, put forward in the works of Grobelny [10, 11, 13, 14, 16]. Along with the Łukasiewicz formula used for obtaining a gradual truth of an implication they enable to express goal functions and output data in a versatile and very flexible way. Such an approach gives a possibility of taking advantage of experts' knowledge, experience, and intuition.

Beside treating the FL criteria softly, a significant reduction of FL modeling constrains was achieved in a breaking through method developed by Drezner [6] concerned with arranging a large number of objects on a plane without the necessity of defining a grid of available locations. This heuristic way of computing objects coordinates is based on eigen values and eigen vectors and produces very good solutions for objects represented by points on a plane. Another method that follows this idea was proposed by Grobelny [15]. In contrast to the Drezner's algorithm, Grobelny's model can dynamically generate various solutions for the same FL problem as some of its parameters are specified at random.

2.3 Hybrid Approaches

Another demonstration of a flexible approach to FL modeling is an attempt of applying several methods or phases within the confines of a single computer system implemented to solve FL problems. One of the first such ideas was presented by Scriabin and Vergin [33]. They first used a multidimensional scaling method for seeking initial objects' arrangements in a form a graphical sketch and then applied the classical CRAFT procedure for improving the initial solution projected on a regular grid.

Combining different algorithms into a single approach was also the subject to analyses in proposals based on genetic algorithms (e.g. [22, 31]. Significant improvements in effectiveness of applying the CRAFT algorithm with initial solutions generated by scatter plot methods were reported also in the works of Grobelny et al. [12] and Michalski and Grobelny [29].

3 Proposed Concept of the Flexible System Supporting Facilities Layout Optimization

The various aspects of a flexible approaches to FL optimizations depicted in a previous section constituted a basis for determining assumptions and implementing the initial version of our Alinks software. The general idea of our approach is demonstrated in Fig. 1 and described in next subsections.

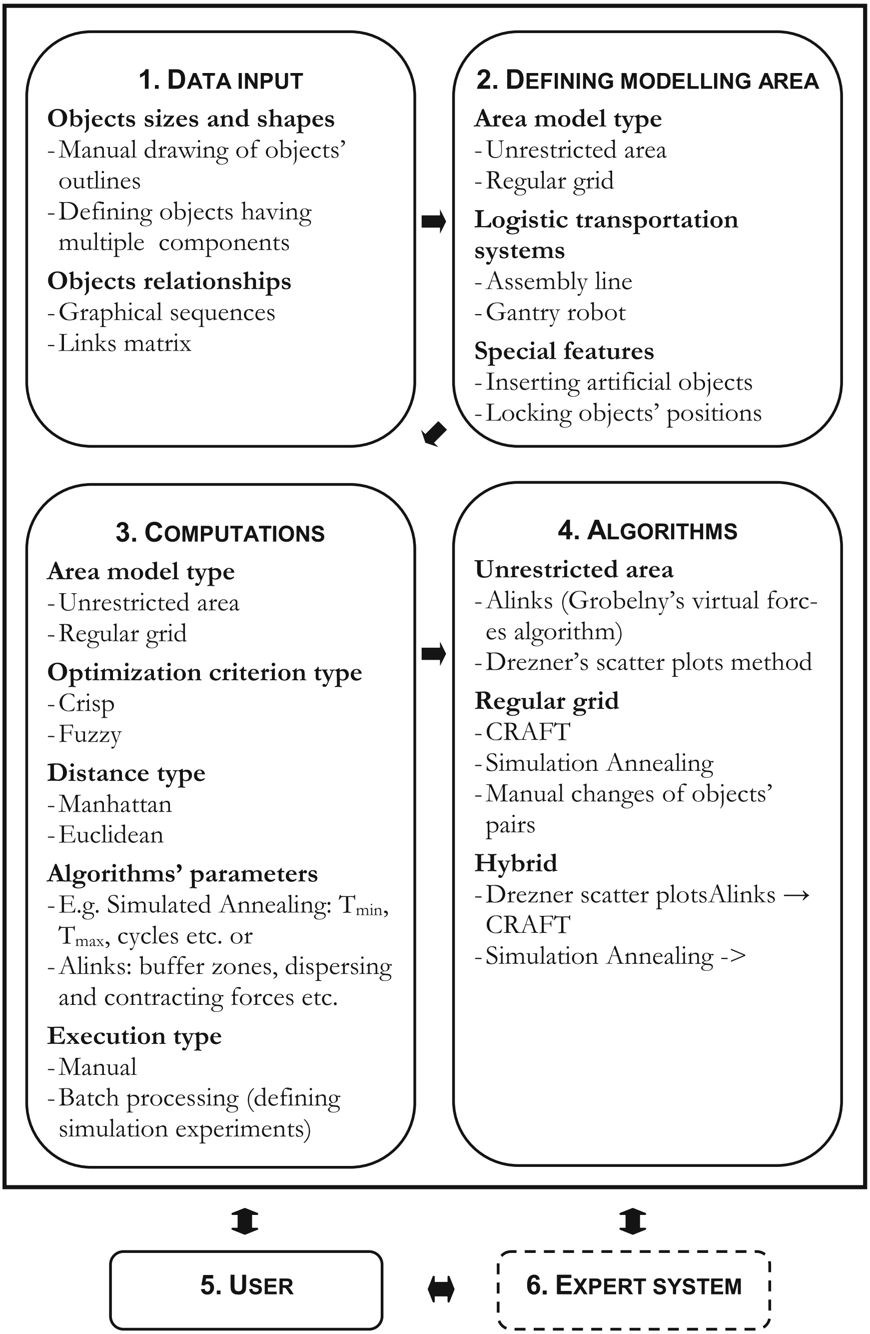

Fig. 1 The general idea of the software allowing for a flexible approach to model and solve various types of facilities layout problems

3.1 Data Input

The flexibility concerned with defining objects' sizes and shapes is realized in the first module of the system named data input. A user may specify objects' dimensions by drawing objects' rectangular outlines manually or by assigning an appropriate number of unitary regular grid cells to a given object. Relationships between objects and objects' components may be introduced either on a graphical scene by providing appropriate sequences that correspond to the material transport paths between objects (like in the spaghetti diagram) or by putting link values directly into the relationships matrix.

Associations between objects may be defined by: (i) binary values (1—objects should be located next to each other, 0—direct neighborhood of this pair of objects is not required), (ii) precise quantitative values, but also as (iii) linguistic variables. The latter case refers to the flexibility described in Sect. 2.2. The concept of linguistic patterns developed by Grobelny [10, 11, 13, 14, 16] is used here for determining the fuzzy goal function values. The linguistic patterns, define the understanding of terms strong relationship and small distance in trapezoidal or/and triangular fuzzy numbers.

3.2 Defining Modeling Area

Decisions realized in the second module called Defining modeling area allow for taking advantage of the scatter plot concepts developed by Drezner [6] and Grobelny [10, 11, 13, 14, 16]. In this mode, there are no predefined locations for objects. The user defines only the available space for objects by specifying rectangle dimensions in natural measurement units.

In the regular grid mode, one defines sizes and shapes of objects and their locations by means of unitary grid cells. It is possible for specify a simplified grid shape for typical logistic transport systems such as an assembly line or gantry robot. Both approaches are illustrated in Fig. 2.

Apart from that it is possible to lock the location of a specific object and insert artificial objects. These features are especially useful for designing layouts of already existing spaces where it is necessary to include various constrains, e.g. aisles or transport routes that need to be excluded from the possible target places for objects being arranged.

3.3 Computations

The third module regards computations directed at searching for optimal layouts either on an unrestricted plane or on a regular grid. The performance and behavior

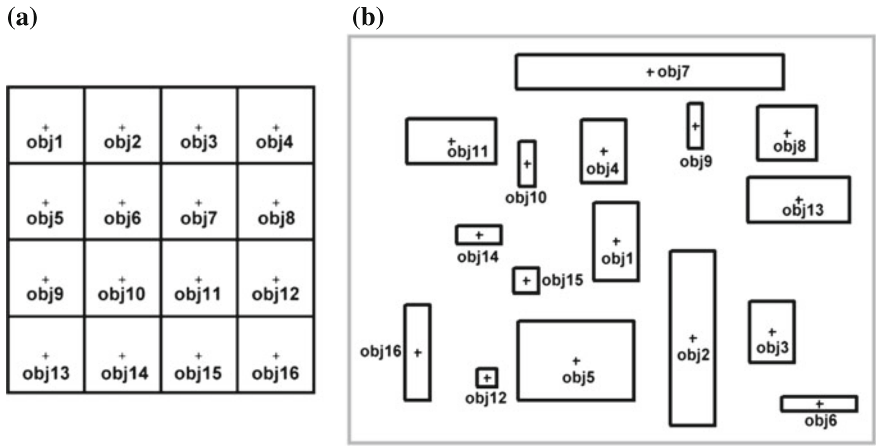

Fig. 2 The illustrations of regular grid (**a**) and unrestricted (**b**) modeling areas for FL

of selected algorithms from the fourth module depend to a significant extent on the attributes selected in this part of the system. The flexibility of our approach allows for selecting the goal function type (crisp or fuzzy) and the way the distance between objects should be calculated (Manhattan or Euclidean). Parameters necessary for the implemented algorithms to operate properly are also set in this section of the software. Additionally, the system has been prepared in such a way that it is possible to perform not only a single simulation but also a series of predefined simulations.

3.4 Algorithms

Algorithms that are available in the proposed system can be generally divided into those that operate on an unrestricted area and those where a regular grid is necessary for optimization procedure. The greatest degree of flexibility is characteristic for methods using unrestricted plane, that is: the Drezner's scatter plots method and Grobelny's virtual force algorithm. Although both approaches are similar in their general idea, there is a fundamental philosophical difference between them. While the former is deterministic in nature, the latter involves stochastic components. Therefore, Drezner's procedure provides only one solution for the given FL problem whereas Grobelny's virtual forces heuristic may generate numerous different solutions thanks to incorporated random mechanisms. Both approaches in their original implementations analyzed only dimensionless objects. In the presented system we developed a procedure that allows for taking objects' rectangular outlines into account by preventing them from overlapping.

3.5 User

The implemented capabilities available in the presented modules of the supporting system provide users with a great variety of possibilities for solving facilities layout problems. It is worth noting that apart from using the software in an interactive mode, one can also prepare and conduct automatically a series of simulations. This feature may be particularly useful for carrying out scientific experiments. The ideas of a flexible approach to modeling and solving various types of FL problems may be realized by applying qualitatively different algorithms, their combinations, supported by additional features developed within the framework of the current system. For instance, in the works of Grobelny et al. [12] and Michalski and Grobelny [29] it was shown that it was possible to decidedly improve the FL designs provided by CRAFT by involving the scatter plots concepts into generating initial solutions. Such a hybrid approach that combines in a sequential way the algorithms for regular grids like CRAFT or Simulated Annealing and unrestricted plane methods (Grobelny's virtual forces or Drezner's approach) is supported by the software. It is feasible to transform modular grid solutions to the natural measurement units' space and vice versa.

3.6 Expert System

The expert system module which is currently being developed is meant as a kind of a user supporting guide that prompts how to use the software in a specific design situation. Based on the FL problem parameters entered by a user, the module would produce appropriate suggestions. The system's proposals would be particularly concerned with modeling the solution space type and selecting appropriate optimization criteria definitions from among available options. The expert system section will also contain rules about combining implemented algorithms in sequences to achieve better designs. The latter feature would be mainly based on experimental studies outcomes.

4 Summary

In the presented concept of a computer system we managed to realize a number of postulates concerned with a flexible approach to modeling and solving FL problems. A practical verification of the described ideas requires, naturally, a series of empirical investigations. It seems to be especially interesting to conduct comparative studies demonstrating superiority of the proposed approach over the existing systems that usually take advantage of a single method without a possibility to mix different procedures. Such a research direction was undertaken in experiments reported by Grobelny et al. [12] and Michalski and Grobelny [29]. Particularly promising finding presented in this paper is the behavior of the hybrid method combining classical CRAFT with the Grobelny's virtual forces algorithm. The first one seeks initial solutions on a plane of specific dimensions. The second one, based on objects' pair changes improves the initial solution provided by the previous procedure. Application of algorithms in such a sequence along with the batch processing produced significantly better solutions to test FL problems taken from the literature than those obtained by the CRAFT method alone. However, it was also demonstrated in the same study that Simulated Annealing was not sensitive to initial solutions generated by the virtual forces procedure. A wide area of possible experimental research is additionally related with the virtual forces approach involving fuzzy parameters. Results obtained in such studies would be used for elaborating rules included in the expert system module. The presented concept may be used in practice, for instance by managers or engineers that have in-depth knowledge about the given production process on one hand side, and basic information on how heuristic optimization algorithms work on the other.

Acknowledgments The work was partially financially supported by the Polish National Science Center grant no. 2011/03/B/HS4/04532.

References

1. Armour, G.C., Buffa, E.S.: A heuristic algorithm and simulation approach to relative location of facilities. Manage. Sci. **9**(2), 294–309 (1963). doi:10.1287/mnsc.9.2.294
2. Block, T.E.: Note—a note on "comparison of computer algorithms and visual based methods for plant layout" by M. Scriabin and R. C. Vergin. Manage. Sci. **24**(2), 235–237 (1977). doi:10.1287/mnsc.24.2.235
3. Buffa, E.S., Armour, G.C., Vollmann, T.E.: Allocating facilities with CRAFT. Harvard Bus. Rev. **42**(2), 136–158 (1964)
4. Chen, C.-S., Kengskool, K.: An AUTOCAD-based expert system for plant layout. Comput. Ind. Eng. **19**(1–4), 299–303 (1990). doi:10.1016/0360-8352(90)90126-7
5. Deisenroth, M.P., Apple, J.M.: A computerized plant layout analysis and evaluation technique. In: Annual AIIE Conference, Norcross, GA, pp. 88–110 (1972)
6. Drezner, Z.: A heuristic procedure for the layout of a large number of facilities. Manage. Sci. **33**(7), 907–915 (1987). doi:10.1287/mnsc.33.7.907
7. Drira, A., Pierreval, H., Hajri-Gabouj, S.: Facility layout problems: a survey. Ann. Rev. Control **31**(2), 255–267 (2007). doi:10.1016/j.arcontrol.2007.04.001
8. Edwards, H.K., Gillett, B.E., Hale, M.E.: Modular allocation technique (MAT). Manage. Sci. **17**(3), 161–169 (1970). doi:10.1287/mnsc.17.3.161
9. Gilmor, P.C.: Optimal and suboptimal algorithms for the quadratic assignment problem. J. Soc. Ind. Appl. Math. **10**(2), 305–313 (1962). doi:10.1137/0110022
10. Grobelny, J.: On one possible fuzzy approach to facilities layout problems. Int. J. Prod. Res. **25**(8), 1123–1141 (1987)
11. Grobelny, J.: Metoda "wzorca lingwistycznego" w projektowaniu ergonomicznym struktury przestrzennej. Ergonomia **11**(1), 49–63 (1988)

12. Grobelny, J., Michalski, R., Koszela, J., Wiercioch, M.: The use of scatter plots for finding initial solutions for the CRAFT facility layout problem algorithm. In: Annual International Conference on Industrial, Systems and Design Engineering, 24–27 June 2013, Athens, Greece, ATINER'S Conference Paper Series, No: IND2013-0625 (2013)
13. Grobelny, J.: The fuzzy approach to facilities layout problems. Fuzzy Sets Syst. **23**(2), 175–190 (1987). doi:10.1016/0165-0114(87)90057-1
14. Grobelny, J.: The 'linguistic pattern' method for a workstation layout analysis. Int. J. Prod. Res. **26**(11), 1779–1798 (1988). doi:10.1080/00207548808947991
15. Grobelny, J.: Some remarks on scatter plots generation procedures for facility layout. Int. J. Prod. Res. **37**(5), 1119–1135 (1999). doi:10.1080/002075499191436
16. Grobelny, J., Karwowski, W., Zurada, J.: Applications of fuzzy-based linguistic patterns for the assessment of computer screen design quality. Int. J. Hum. Comput. Interact. **7**(3), 193–212 (1995). doi:10.1080/10447319509526121
17. Hassan, M.M.D., Hogg, G.L., Smith, D.R.: SHAPE: a construction algorithm for area placement evaluation. Int. J. Prod. Res. **24**(5), 1283–1295 (1986). doi:10.1080/00207548608919803
18. Heragu, S., Kusiak, A.: A construction algorithm for facility layout problem. Working paper #14/86, Department of Mechanical and Industrial Engineering, University of Manitoba, Winnipeg, Manitoba, Canada (1986)
19. Hillier, F.S., Connors, M.M.: Quadratic assignment problem algorithms and the location of indivisible facilities. Manage. Sci. **13**(1), 42–57 (1966). doi:10.1287/mnsc.13.1.42
20. Jacobs, F.R.: Note—a note on spacecraft for multi-floor layout planning. Manage. Sci. **30**(5), 648–649 (1984). doi:10.1287/mnsc.30.5.648
21. Johnson, R.V.: Spacecraft for multi-floor layout planning. Manage. Sci. **28**(4), 407–417 (1982). doi:10.1287/mnsc.28.4.407
22. Kazerooni, M., Luonge, L., Abhary, K.: Cell formation using genetic algorithms. Int. J. Flex. Autom. Integr. Manuf. **3**(3–4), 283–299 (1995)
23. Kim, J.-Y., Kim, Y.-D.: Graph theoretic heuristics for unequal-sized facility layout problems. Omega **23**(4), 391–401 (1995). doi:10.1016/0305-0483(95)00016-H
24. Kusiak, A., Heragu, S.S.: The facility layout problem. Eur. J. Oper. Res. **29**(3), 229–251 (1987). doi:10.1016/0377-2217(87)90238-4
25. Lawler, E.L.: The quadratic assignment problem. Manage. Sci. **9**(4), 586–599 (1963). doi:10.1287/mnsc.9.4.586
26. Lee, R., Moore, J.M.: CORELAP—computerized relationship layout planning. J. Ind. Eng. **18**, 195–200 (1967)
27. Liu, X., Sun, X.: A multi-improved genetic algorithm for facility layout optimisation based on slicing tree. Int. J. Prod. Res. **50**(18), 5173–5180 (2012). doi:10.1080/00207543.2011.654011
28. Meller, R.D., Gau, K.-Y.: The facility layout problem: recent and emerging trends and perspectives. J. Manuf. Syst. **15**(5), 351–366 (1996). doi:10.1016/0278-6125(96)84198-7
29. Michalski, R., Grobelny, J.: The role of initial solutions in craft and simulated annealing applied to real life logistics problems. In: International Symposium on Marketing and Logistics (ISML 2014), 8–10 Sept 2014, Nagoya, Japan (2014)
30. Parsaei, H.R., Galbiati III, L.J.: Facilities planning and design with microcomputers. Comput. Ind. Eng. **13**(1–4), 332–335 (1987). doi:10.1016/0360-8352(87)90109-4
31. Sadrzadeh, A.: A genetic algorithm with the heuristic procedure to solve the multi-line layout problem. Comput. Ind. Eng. **62**(4), 1055–1064 (2012). doi:10.1016/j.cie.2011.12.033
32. Sahni, S., Gonzalez, T.: P-complete approximation problems. J. ACM **23**(3), 555–565 (1976). doi:10.1145/321958.321975
33. Scriabin, M., Vergin, R.C.: Comparison of computer algorithms and visual based methods for plant layout. Manage. Sci. **22**(2), 172–181 (1975). doi:10.1287/mnsc.22.2.172
34. Seehof, J.M., Evans, W.O.: Automated layout design program. J. Ind. Eng. **18**(2), 690–695 (1967)
35. Silver, E.: An overview of heuristic solutions methods. J. Oper. Res. Soc. **55**, 936–956 (2004). doi:10.1057/palgrave.jors.2601758

36. Singh, S.P., Sharma, R.R.K.: A review of different approaches to the facility layout problems. Int. J. Adv. Manuf. Technol. **30**(5–6), 425–433 (2006). doi:10.1007/s00170-005-0087-9
37. Tam, K.Y.: A simulated annealing algorithm for allocating space to manufacturing cells. Int. J. Prod. Res. **30**(1), 63–87 (1992). doi:10.1080/00207549208942878
38. Tate, D.M., Smith, A.E.: Unequal-area facility layout by genetic search. IIE Trans. **27**(4), 465–472 (1995). doi:10.1080/07408179508936763
39. Tompkins, J.A., Reed Jr., R.: An applied model for the facilities design problem. Int. J. Prod. Res. **14**(5), 583–595 (1976). doi:10.1080/00207547608956377
40. Tong, X.: SECOT: a sequential construction technique for facility design. Unpublished Doctoral Dissertation, Department of Industrial Engineering, University of Pittsburgh (1991)
41. Zanakis, S.H., Evans, J.R., Vazacopoulos, A.A.: Heuristic methods and applications: a categorized survey. Eur. J. Oper. Res. **43**(1), 88–110 (1989). doi:10.1016/0377-2217(89)90412-8
42. Ziai, R.M., Sule, D.R.: Plant layout—local area networks: microcomputer facility layout design. Comput. Ind. Eng. **15**(1–4), 259–263 (1988). doi:10.1016/0360-8352(88)90096-4

Smart Innovation Management in Product Life Cycle

Mohammad Maqbool Waris, Cesar Sanin and Edward Szczerbicki

Abstract The present paper proposes a framework for smart innovation management of the product using a Smart Knowledge Management System comprising Set of Experience Knowledge Structure (SOEKS) and Decisional DNA. This proposed system will allow the entrepreneurs and organizations to perform the innovation process technically and quickly as this framework will store knowledge as well as experiences of the past innovations done in various products. It will also consider the possibility of introducing the new technology into the product. In short it will act as the expert innovator capable of doing the innovation technically utilizing the past experiences and several possible options based on the priorities of the user.

Keywords Smart innovation · Innovation management · Product life cycle · Set of experience · Decisional DNA · Smart knowledge management system

1 Introduction

Since the beginning of life on Earth, humans are continuously gathering knowledge through experiences they face during their day to day activities. Fortunately, this knowledge is stored in human mind that can be easily recalled if similar situations arise. Moreover, based on previous experiences, humans are able to analyze smartly whenever a new task comes. Further, Innovation can be considered as the outcome of human minds who think and analyze the knowledge smartly like entrepreneurs. Both knowledge and experience are crucial attributes of an innovator to find an

M.M. Waris (✉) · C. Sanin (✉)
The University of Newcastle, Callaghan, NSW, Australia
e-mail: MohammadMaqbool.Waris@uon.edu.au

C. Sanin
e-mail: cesar.sanin@newcastle.edu.au

E. Szczerbicki (✉)
Gdansk University of Technology, Gdansk, Poland
e-mail: edward.szczerbicki@newcastle.edu.au

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_16

optimum solution for the necessary changes to the established product to increase the product life. But due to rapid changes in the dynamic environment and enormous amount of ever increasing knowledge, the task of innovation management is becoming difficult to practice. Innovators need to take quick and proper decisions so that the changes in the product can be implemented in time.

Similarly, if a Smart Knowledge Management System (SKMS) can be formulated for representing and analyzing data that is able to present results to queries for possible innovation options, and most importantly, it can store the experience based on the decision taken for a particular query, such a system will make the innovation process technically sound as it is dynamic in nature and continue to grow or capturing knowledge in the same way as the human gene does. A systematic approach in Knowledge Management (KM) process will help to survive in the dynamic environment. Decision makers look back on the lessons learnt from the previous similar situations as a base for making current decisions [1]. However, due to inefficient knowledge administration and failure to capitalize the experiences within the organization, the decision making process takes high response time, repetition of decisions already made, and inability to keep pace with the dynamic environment. Using a SKMS for taking decisions in product innovation will make the innovation process faster and smarter.

Product innovation can be broadly divided into two aspects; (a) Technical Innovation, which is based on technological grounds, and (b) Commercial Innovation which is based on business and marketing grounds. This present study deals with the technical innovation of the product. These two aspects are not entirely independent of each other. In fact, some of the characteristics based on business and marketing ground can be implemented only by properly selected technical changes. For example one of the main feature of commercial is the cost price of the product to compete with other competitors manufacturing the same product. The cost price of the product can be reduced by controlling the manufacturing cost that can be accomplished by replacing the material(s) of the product with the cheaper one or by replacing the manufacturing process with the one requiring less manufacturing cost. Similarly some other features of the commercial innovation can be accomplished by making proper changes in features representing technical innovation. To perform the task of technical innovation, a knowledge representation technique is required that is capable of storing knowledge as well as experience from the formal decision events. Such a system was presented by Sanin and Szczerbicki [2, 3] known as Set of Experience Knowledge Structure (SOEKS) comprising Decisional DNA (DDNA).

In short, introduction of the SKMS in product innovation process will help the organizations and entrepreneurs to take proper decisions on time as this SKMS not only stores information and knowledge but store experiences of the formal decision events. These experiences are recalled when similar query occurs in the same way as humans utilizes their past experiences.

2 Background

2.1 Product Life Cycle

Every time a new product is launched into the market, it looks that it is going to be used for a long time due to its initial impact on the customers. But it cannot continue to be used forever, every product has some life after that it is no longer be used by the customers, the reasons can be introduction of the new product having enhanced features like cheaper, faster, lighter, and so on. The life of a product is represented by Product Life Cycle (PLC). It generally consists of four phases, as shown in Fig. 1, which are briefly described below:

Introduction: The product is designed and manufactured in the organization and then launched into the market. The cost of the new product is high due to less competition and sales rises slowly. The profits are negative due to high cost of research and development and marketing expenditure.

Growth: As sales increase due to improved product and market penetration, the profit increases resulting from lower production cost even if the cost of the product decreases due to more competition. At this stage more organizations jump into the business and start the manufacturing.

Maturity: The sales continues to increase and reaches the peak, the cost of the product is reduced to lowest due to shear competition. The product is differentiated with other brands to attract the consumers.

Decline: The sales starts to decline that may be caused by introduction of better products, product substitution, trend and mood change.

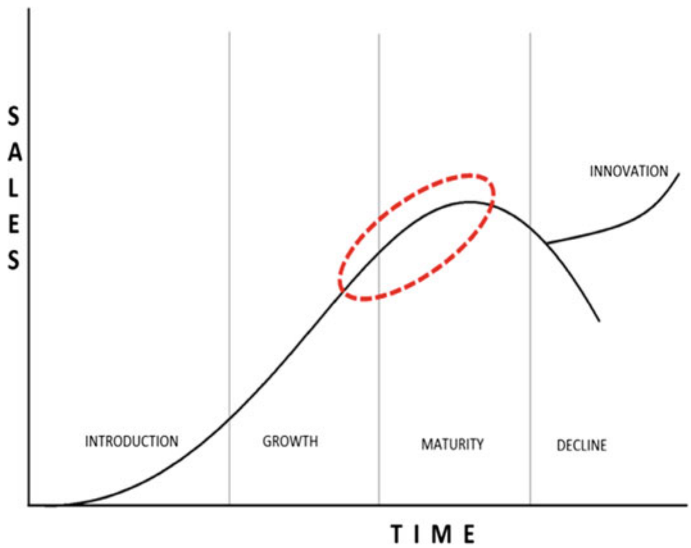

Fig. 1 The product life cycle with the introduction of innovation at later phases

Due to continuous rapid changes in the technology and preferences of the consumers, new products can no longer be developed by organizations solely based on their ideas [4]. The organizations need to consider the consumer needs and strategies adopted by their competitors. This will lead to a better innovative changes in the product at proper time.

2.2 Innovation

Innovation has been an important topic of study in various fields including economics, management science, business, engineering, science, hospitality, sociology, among others. In spite of the fact that it is widely studied, it is often confused with terms such as invention, creativity and improvement [5]. Innovation is defined as the process of making changes to something established by introducing something new that add value to users and contributes to the knowledge store of the organization [5]. It is important to note that innovation necessarily adds value to the customers. Invention on the other hand is the creation of something new and does not need to fulfil any customer need. Invention however can be exploited and transformed into a change that adds value to the customers; thus, becoming an innovation. Some inventions are not considered to be incorporated into the products as it will not result in addition of some value to the user or even it may result in higher cost. Drucker [6] defines innovation as the specific tool of entrepreneurs, the means by which they exploit changes as an opportunity for a different business or service. According to him the sources of innovation are changes in industry structure or market structure, demographics, changes in perception, mood or meaning and application of new knowledge.

Innovation is often associated with physical products. However, innovation is broadly divided into three categories viz. product innovation, process innovation and service innovation. The present study is concerned with the technical innovation in the domain of product/process innovation that is tangible products in general.

2.3 Set of Experience Knowledge Structure and Decisional DNA

The Set of Experience Knowledge Structure (SOEKS or simply SOE) is a smart knowledge structure capable of storing explicitly formal decision events [7–9]. This smart knowledge based decision support tool stores and maintains experiential knowledge and uses such experiences in decision-making when a query is presented. The SOE has four basic components: variables (V), functions (F), constraints (C) and rules (R) (Fig. 2) [1] and it comprises a series of mathematical

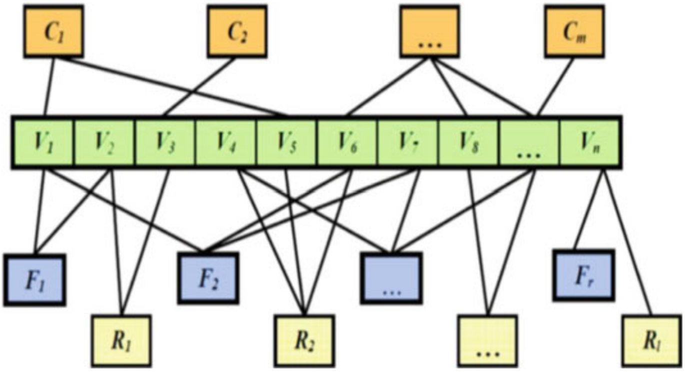

Fig. 2 Structure of set of experience

concepts (logical element), together with a set of rules (ruled based element), and it is built upon a specific event of decision-making (frame element). Its structure is motivated by Artificial Bio-inspired intelligent techniques supporting knowledge-based solutions of real world problems and has enormous potential to enhance automation of decision making and problem solving for a number of diverse areas, creating unprecedented research opportunities in an extent of fields. Our research converges on the application of SOE in the development of smart innovation management platform that could cover processes of the whole product life cycle.

A formal decision event is uniquely represented by the set of SOE components as shown in Fig. 3. Variables are considered as the root of the structure as they are required to define the other components. Functions are the relationship between a

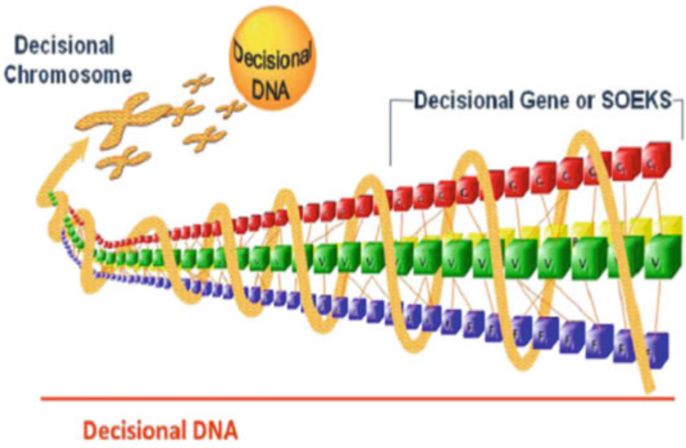

Fig. 3 Set of experience and decisional DNA [9]

dependent variable and a set of input variables. Functions are used by the SOE for establishing links between variables and constructing multi-objective goals. Constraints are also functions that are used to set the limit to the feasible solutions and control system performance with respect to its goals. Rules, on the other hand, are the conditional relationships among the variables and are defined in terms of If-Then-Else statements.

A formal decision event is represented by a unique set of variables, functions, constraints and rules within the SOEKS. Groups of SOEs are called chromosomes that represent a specific area within the organization and store decisional strategies for a category. Properly organized and grouped sets of chromosomes of the organization is collectively known as the Decisional DNA of the organization. Applications of SOE and DDNA have been successfully applied in various fields like industrial maintenance, semantic enhancement of virtual engineering applications, state-of-the-art digital control system of the geothermal and renewable energy, storing information and making periodic decisions in banking activities and supervision, e-decisional community, virtual organisation, interactive TV, and decision support medical systems for Alzheimer's diagnosis to name a few. For details see Shafiq et al. [10].

In the next section it is shown that how the Set of Experience Knowledge Structure and Decisional DNA can be applied in taking proper decisions in product innovation. The framework for smart innovation management is discussed that explains how solutions to the particular query are provided utilizing the knowledge of the past decisional events at the same time.

3 Innovation in Product Life Cycle

Organizations involved in product manufacturing cannot grow only through cost reduction and improved engineering processes. To survive and prosper, they need to innovate continuously to find out new ideas. The reasons are rising costs of material and energy, competition at international level, new technologies, frequent changes in lifestyles and automation. Schumpeter [11] describes innovation as the use of an invention to create a new product or service resulting in the creation of new demand. The introduction of a new product into the market destroys the demand of existing products and creates demand for new ones, and so on. He termed it as creative destruction.

Despite the fact that we one cannot predict future, a systematic and proper approach can increase the life of the product. The most important point is to determine the proper time for applying innovation at later phases of the PLC. Many organizations fail to recognize an alarming time for starting innovation process (the critical zone is shown enclosed by a red circle in Fig. 1) so that the resulted innovative changes can be applied to the product at the proper time (preferably at the end of maturity or at the beginning of decline phase); they start the innovating process when the sales start to decline or at least becomes stagnant. By the time they introduce changes into the product, sales are almost declined to the minimum and do not find time for further necessary innovations to survive in the market. The sales continue to increase exponentially in the maturity phase. The alarming time should be the point of inflection i.e. the time at which the rate of increase of sales starts decreasing or the curve changes from concave to convex. At this point, the organization needs to analyze the factors for innovation, on both technical and business ground.

Once innovation is implemented on suitable time, at the end of maturity phase or during the start of the decline phase, product sales increase again. And this can be repeated few times until the effect of innovation does not results in appreciable sales of the product or the product becomes obsolete [12].

Once the data from the above sources is available, organizations can find out which features or functions of the product need to be upgraded, which ones may be excluded and which new features or functions may be added to the product. These features and functions are attributed to some component(s) of the product. Innovative changes in the product can be performed by modifying one or more components of the product. Accordingly, the required changes can be incorporated into the product to complete the innovation process. Product innovation is a continuous process; no organization, however big or successful, can continue to grow on the past achievements. In fact, like PLC, innovation also has some life and after some time it is replaced by another innovation. If organizations do not innovate regularly, their product will become non-competitive. Okpara [12] says that enterprises that rely exclusively on innovation will prosper until their products and services become obsolete and non-competitive.

3.1 Framework for Smart Innovation Management

The framework for the technical innovation incorporating SOEKS and DDNA is shown in Fig. 4. It consists of user interface, integrator, prognoser, solution and decisional DNA. The detail about these components is explained as follows:

User Interface: Users can interact with the system by inputting data into the user interface. The data can be a simple query for innovation or addition of any new technology/process to the knowledge base of the platform; for example a query for suitable replacement of fuel injector of a particular car, or addition of the new technology/device used for fuel injection.

Integrator: The integrator receives data from the user interface and acquires information through various applications. It produces sub-solutions according to the objectives of the users; for example size and weight of the fuel injector, its input/output considerations, cost, service life, and so on. Next it transforms the information into a unified language and measurement system. It gathers and organizes the data and transforms it into sets of experience described by the unique set of variables, functions, constraints, and rules. The integrator interacts with the

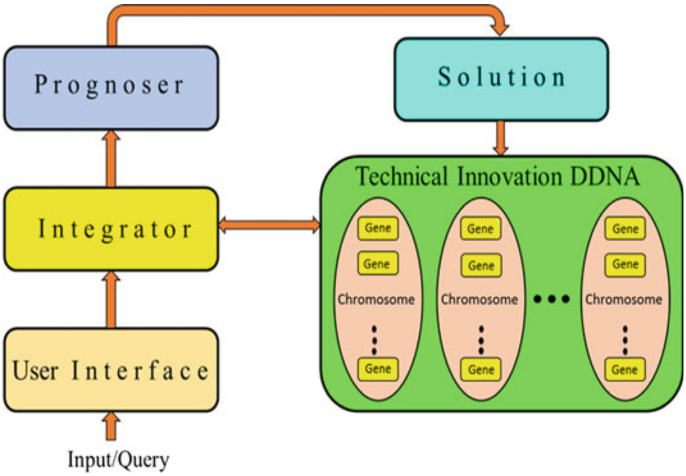

Fig. 4 Framework for smart innovation management

decisional DNA for similar sets of experiences and sends the results to the prognoser for further processing.

Prognoser: The prognoser first produces sub-solutions provided by multiple applications according to its established objectives. Then depending on the various scenarios new sets of model can be built taking into account measurements of uncertainty, incompleteness and imprecision. The prognoser finally produces set of proposed solutions that are sent to the solution layer.

Solution: The solution layer allows the user to select the best solution out of the proposed solutions. Based on the priorities defined by the user it chooses the best solution among the possible solutions provided. The decisional event is then sent to the Decisional DNA of technical innovation.

Technical Innovation DDNA: It is where the sets of experience, formal decisional events, are stored and managed. One set of experience represents a decisional DNA gene [7]. Same category of genes are grouped together which is collectively called as decisional chromosome. Group of such chromosomes; product chromosome, process chromosome and technology chromosome, constitute a decisional DNA of technical innovation. It also interacts with other components of the framework during the solution process and presents similar experiences that helps in finding a reliable solution in less time.

3.2 Capturing Knowledge for Technical Innovation

Innovation is done on the bases of new ideas based on collected information. This results in the creation of new knowledge and decision making events. If a knowledge structure can be made that store the data from various sources mentioned above and captures formal decision events, whether successful or past failures, the innovative decision can be made by enquiring the SOEKS-DDNA structure. Any query sent into this system will return with some options corresponding to the priorities and preciseness of the proposed changes. Once the decision has been made, the decisional event will be stored in the system as an experience in terms of a defined set of variables, functions, constraints and rules. Such experiences are stored every time the query is made and decision has been taken. This way the system grows dynamically in collecting knowledge and experience. These experiences will useful when similar query comes to the system and this way the repetition process is avoided.

4 Conclusion

This paper presented the smart innovation management process utilizing the SOEKS and DDNA. It was explained how the innovation can be done on technical grounds based on the previous innovative experiences of the similar products. It was also demonstrated how the present decision event can be stored in an organized manner so that it can be utilized in future. The concept of SOEKS/DDNA is a novel knowledge representation structure, it enabled the innovation process to be dynamic and ever increasing in experience.

References

1. Sanin, C., Szczerbicki, E.: Set of experience: a knowledge structure for formal decision events. Found. Control Manage. Sci. **3**, 95–113 (2005)
2. Sanin, C., Szczerbicki, E.: A complete example of set of experience knowledge structure in XML. In: Szuwarzynski, A. (ed.) Knowledge Management: Selected Issues 2005, pp. 99–112. Gdansk University Press, Gdansk, Poland (2005)
3. Sanin, C., Szczerbicki, E.: Decisional DNA and the smart knowledge management system: a process of transforming information into knowledge. In: Gunasekaran, A. (ed.) Techniques and Tool for the Design and Implementation of Enterprise Information Systems 2008, pp. 149–175, New York, USA (2008)
4. Day, G.S.: The capabilities of market-driven organizations. J. Mark. **58**(1), 37–52 (1994)
5. O'Sullivan, D., Dooley, L.: Applying Innovation. Sage Publications, London (2008)
6. Drucker, P.: Innovation and Entrepreneurship: Practice and Principles. Butterworth Heinemann, Boston (1985)
7. Sanin, C., Szczerbicki, E.: Towards the construction of decisional DNA: a set of experience knowledge structure java class within an ontology system. Cybern. Syst. **38**, 859–878 (2007)
8. Sanin, C., Szczerbicki, E.: Towards decisional DNA: developing a holistic set of experience knowledge structure. Found. Control Manage. Sci. **9**, 109–122 (2008)
9. Sanin, C., Toro, C., Haoxi, Z., Sanchez, E., Szczerbicki, E., Carrasco, E., Peng, W., Mancilla-Amaya, L.: Decisional DNA: a multi-technology shareable knowledge structure for decisional experience. Neurocomputing **88**, 42–53 (2012)

10. Shafiq, S.I., Sanin, C., Szczerbicki, E.: Set of experience knowledge structure (SOEKS) and decisional DNA (DDNA): past, present and future. Cybern. Syst. **45**(2), 200–215 (2014)
11. Schumpeter, J.A.: The Theory of Economic Development. Harvard University Press, Cambridge (1934)
12. Okpara, F.O.: The value of creativity and innovation in entrepreneurship. J. Asia Entrepreneurship Sustain. **3**(2), 81–93 (2007)

Transformation in Model Driven Architecture

Matilda Drozdova, Martin Kardos, Zuzana Kurillova and Boris Bucko

Abstract This article focuses on using of top level Model Driven Architecture (MDA) modelling and how automatic transformation to lower levels can be accomplished in information systems development. The primary aim of this work is to design a systems approach to Computer Independent Model (CIM)-Platform Independent Model (PIM) transformation in MDA. These approaches to CIM-PIM transformation are designed for selected models representing CIM and PIM levels of abstraction. The final automated CIM to PIM transformation we present is based on business process modelling by means of Business Process Model and Notation (BPMN). Models of business processes in BPMN are stored in Extensible Markup Language Process Definition Language (XMLPDL/XPDL) format, which is used for automated transformation into selected Unified Modelling Language (UML) design models, using transformation algorithms generated in eXtensible Stylesheet Language Transformations (XSLT) language.

Keywords Model driven architecture · Computer independent model · Platform independent model · Automated transformation · Business process model and notation · Unified modelling language

M. Drozdova (✉) · M. Kardos · Z. Kurillova · B. Bucko
Faculty of Management Science and Informatics, University of Zilina,
Univerzitna 8215/1, 010 26 Zilina, Slovak Republic
e-mail: matilda.drozdova@fri.uniza.sk

M. Kardos
e-mail: martin.kardos@matoo.sk

Z. Kurillova
e-mail: zuzana.kurillova@fri.uniza.sk

B. Bucko
University Science Park, University of Zilina, Univerzitna 8215/1,
010 26 Zilina, Slovak Republic
e-mail: boris.bucko@uniza.sk

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*,
Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_17

1 Introduction

Continuous innovation and development of new information systems (IS) is designed to elevate IS development to higher levels of abstraction. Thus, from the standpoint of implementation, IS development is of concern not only to software designers, but to non-IT professionals as well. From the standpoint of IS implementation, relative lower level of IT knowledge should not be a constraint on involvement of non-IT professionals in the development process. Non-IT professionals are required to possess a thorough understanding of design issues, as their contribution to the functionality of the system is crucial. Many existing approaches do not distinguish between IT/business alignment situations. Enterprise architecture (EA), mentioned in [1, 2], can be used for this connection, and EA models are the outputs of these researches.

This principle is important mainly by development of personalized IS that influence success of an organization. To attain this goal it is necessary to guarantee that IS supports all major business and management processes to be automated. Hence, when designing IS, we need to include and understand many viewpoints concerning how the IS interacts with organizational systems external to it. System architecture enables capture, specification and description of entities and relationships to support communication between viewpoints, and has therefore become a principal method of IS design.

Description of system architecture is well defined by standard ISO/IEC/IEEE 42010:2011. Application of this standard in a concrete IS leads to architecture based on partial viewpoints of which relationships help to create architecture frameworks for a certain IS. Communication between participants requires rules of correspondence to translate that transform relationships among various viewpoints of IS. It is the one of problems mentioned in [3] context with system and software architecture (SSA), between requirements specification and architectural design description presented in [4] and requirements engineering and software architecture in [5].

We have come to understand that in the past we have overlooked this type of communication among various participants in IS design in support processes at University of Zilina. Therefore, these rules of correspondence as they apply to the two architecture viewpoints have become the focus of our research. To define rules of communication we have chosen the principle of Model Driven Architecture (MDA) as standardized by Object Management Group (MDA Guide Version 1.0.1).

Model Driven Architecture—MDA is one of the approaches to IS design that can help to attain the above-mentioned requirements for communication among various architecture viewpoints. MDA framework is based on systematic use of models that represent various levels of abstraction and transformations [6]. It is based on the understanding that using models we can build, modify and manage information systems in an improved manner. MDA introduces four interconnected levels. The top level of abstraction is CIM (Computer Independent Model), then PIM (Platform Independent Model) next is PSM (Platform Specific Model) and finally, the lowest level is Implementation Code. OMG describes levels as models; however these models represent different levels of abstraction. The MDA framework does not precisely define exact types of models required at all levels. It describes individual levels and relationships between them. There is no standardized method to create and transform the levels defined by MDA, only general rules describing the level of abstraction.

Considering the MDA standard mentioned above, CIM level represents the business viewpoint understandable even by non-IT professionals while PIM level represents the software design viewpoint.

Nowadays we can find plenty of solutions for MDA, starting at the PIM level and using automatic PIM-PSM transformations, and further to the creation of basic implementation code. There are CASE tools and some MDA tools for automation of these procedures. However there is a lack of standardized approaches for CIM to PIM transformations and the tools for this purpose. Therefore we propose that resolving this issue would be a major advance towards achieving the final vision of MDA.

In our research we focus on transformation rules between the top level of abstraction CIM and lower level PIM. We find this to be the most problematic part of MDA framework. Only a few researches we have found of this area and some of them we analyse in the related works. Transformation of CIM to PIM is not defined by OMG and carries out mostly analytically. Therefore we assume that this issue should be a great asset to the MDA approach.

In the following sections we introduce our research methods and the results obtained.

2 Related Works

In examining the current state of the art of the problem under assessment, we evaluated different approaches of CIM-PIM transformation in MDA [7–13].

From them we derive following:

1. All notations of PIM levels are defined in UML language.
2. PIM level is, apart from approach No. 2, a model of components for basic software architecture. This does not uniformly equate to models of business entities.
3. Notations of CIM level are:

   (a) UML 2.0,
   (b) BPMN,

(c) AND/OR graph,
(d) Description.

CIM level is defined as:

1. feature model,
2. requirement model,
3. business processes model,
4. value model,
5. model goals and information requirements,
6. secure business process.

By examining CIM to PIM transformations illustrated in similar studies it becomes evident that no existing approaches as yet is fully supported by standards specified for MDA. So far only non-standardized approaches have been defined by the authors and these results in loss of overall interoperability. The majority of tools used to support modelling and transformations are limited by cost (for example by QVT) and by limited cooperation among systems. A standard definition of a stable framework for CIM to PIM transformation and methodology that supports generally accepted standards is needed. This approach would enhance compatibility among many standardized development tools.

3 Problem Solving

We have considered four basic conditions when designing automatic CIM-PIM transformation:

1. Notation for business process model must be sufficiently expressive and easily understandable for business and IT users.
2. Notation for business process model has to be standardized in graphical as well as in semantic form.
3. Notation for design models has to be standardized in graphical as well as in semantic form.
4. Built-in automatized approach to CIM-PIM transformation should be universal (platform independent) and extensible.

Conformance to the first condition is ensured by usage of BPMN notation (OMG Inc. BPMN version 2.0). BPMN is standardized graphical notation, assigned for process modelling. It can be readily understood by those in the business as well as IT domains. This notation provides a sufficiently large expressiveness because it is based on graphical visualization of business processes. This has the great advantage of making business processes readily understandable to the other domains.

Because BPMN is only a graphical standard, for the purpose of transformation it is necessary to transform the business process models from BPMN to standardized semantic (textual) form. The XPDL format (WFMC–TC-1025 Version 2.1a; WFMC-TC-1025 Version 2.2) that introduces standardized semantic form of script of the business process model in BPMN notation, is one solution for this requirement. Conversion of BPMN–XPDL is automatically accomplished in most of the available, business process oriented modelling tools. Via XPDL it is possible to expand even BPMN element semantics. Use of XPDL ensures compliance with the second condition.

Compliance with the third condition is an issue pertaining to the modelling language UML, as well as its standardized interchange MOF XMI format, which allows semantic form of UML model script. XPDL and MOF XMI are formats based on descriptive XML (eXtensible Markup Language) language. Therefore we have chosen XSLT (eXtensible Stylesheet Language Transformations) for creation of transformation algorithms for automatic CIM-PIM approach. XSLT language enables creation of transformation rules among various formats based on XML. XSLT provides versatility to these transformations because XSLT processors that perform transformation algorithms are freely available and can be implemented on any platform (Windows, Linux). This approach also ensures expandability in cases where there is a requirement for enrichment of BPMN semantics, as well as transformation to other models based on XML. By this means, even the fourth condition of transformation is fulfilled.

3.1 Methodology Transformation Principles

Automatic transformation of CIM-PIM is based on creation of transformation rules (algorithms) in XSLT language, where elements of business process model in BPMN represented by XPDL format are transformed into elements of selected UML models represented by MOF XMI format. This format is normally importable to UML/CASE modelling tools. Two types of output models were chose to facilitate this transformation:

- UML use case diagram.
- Model of business entities of IS represented by UML conceptual class diagram.

This is because UML use case diagram represents functionality of IS, and model of business entities represents business entities having defined operations or attributes that must be processed in IS. For MOF XMI format it was necessary to perform an analysis by placement of individual elements in selected UML models. Based on these considerations, general XMI templates were created for selected

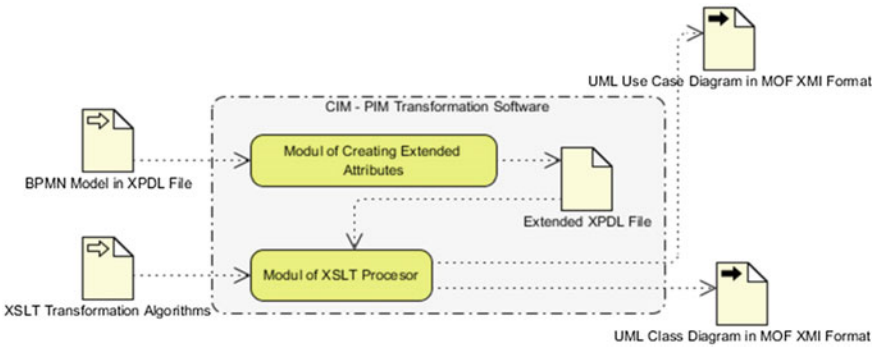

Fig. 1 Automated approach to CIM-PIM transformation

UML model types that serve as a foundation for generating output models. This transformation process is illustrated in Fig. 1.

Principle of transformation is as follows. After creation of business process model in BPMN notation, the model is automatically transformed by modelling tool to XPDL format. XPDL format, along with resulting XSLD transformation rules, become the input parameters for software application created by Kardos [14]. The software application has following functionalities:

1. It automates creation of expanded attributes of XPDL format to expand semantics of BPMN elements.
2. XSLT processor applies generated XSLT transformation algorithms to the input XPDL format.
3. Applying XSLT rules creates outputs in the form MOF XMI files that represent UML use case class diagrams. UML conceptual class diagram represents business entity model of IS.

Creation of software application is necessary because it automates two routine activities. First, it extends the semantics BPMN element about data attributes that would otherwise have to be added manually by editing the file representing XPDL format. This is a time consuming and non-transparent procedure. Second is usage of XSLT processor that is implemented in the resulting application. Without the application, it would be necessary to enter the inputs to XSLT processor via command line. The processor of library MSXML, which is part of .Net C#, was implemented within the created XSLT application. In addition, the resulting software application automates routine activities which are encapsulated into one unit.

The authors present as an example of transformation algorithms we refer to as an Actors creation of UML Use Case diagrams. To simplify XSLT transformation algorithm, the principles are described by a pseudo-algorithm that captures mapping foundation of BPMN elements represented by XPDL format to the Actors type elements of Use Case UML diagram.

Actors can be created based on two basic elements. First is creation of actor from BPMN elements—swimlanes that are blocks of pools or lanes. These BPMN elements help to organize and sort single activities. Pool in case of interactive B2B (Business to Business) processes represents business roles—participant/member who performs the activities in the process. Pool as a participant can act in two roles, as a business entity (for example University of Zilina) or business role (for example teacher, researcher). Lane is a sub partition of a pool and therefore it can represent certain business roles (for example manager). It can also represent any other necessary features of business process. Since a pool can represent business role and contains other lanes, then by creation of actors we can use the principle of generalization to create actor-ancestor from pool, and actor-descendant from lanes. In XPDL format there are pools and lanes described by XPDL elements with equal name—pool and lane. Creation of actors based on pools or lanes is suitable in case of simple, well mapped business processes.

Creation of performers in framework of business processes has developed into a universal method of creating actors suitable for cases of simple and complex business process maps.

Performers represent participants that carry out certain process or activity and can also appear in two roles: either as a business entity or a business role. Creation of actors via performers is completely universal and can be used to represent the most complex business processes. Performers in XPDL format are described as elements of type Participant.

A pseudo algorithm for actor creation of a UML use case diagram is shown in Fig. 2. Algorithm starts by initialization of variables. The input function argument is XPDL file and parameter *FromParticipant* express, if actors should be created on the basis XPDL elements, type Participant. If the value of this parameter is set to true, XPDL elements, type Participant are checked. If the number of elements in the array Participant is greater than zero, this array is checked one by one, and from the current element Participant an actor *umlActor* is created and inserted into array of actors*umlActors* (pseudo function *CreateUmlActor*). If the value of parameter *FromParticipant* is set to false, actors are being created from pools. Array pools are checked one by one element and from each element of pool, there is an actor *umlActor* created that is inserted into actors-array *umlActors*. If the current element pool contains lanes and at the same moment it is of "*Business Role*" type, the array Pool. Lanes is being checked, where each element lane is turned into actor *umlActor* and a generalization of actors is created too (relationship ancestor–predecessor) among currently checked element pool and lane. In this case parameter *Generalization* is set to true.

```
Function Create_UMLUseCase_Actors(var xpdl: file;
                                  var FromPArticipant) {
  var Pools:= xpdl.Pools;
  var Participants := xpal.Participants;
  var umlActors := Array();
  var i := 0;

  If(FromParticipant == true) {
   If(Count(Participants) != 0) {
    Foreach (var Participant in Participants) {
     umlActors[i] := CreateUmlActor(Participant,null,false);
     i++;
    }
   }
  }
  Else {
   Foreach (var Pool in Pools) {
    umlActors[i] := CreateUmlActor(Pool, null, false);
    i++;
    If(Count(Pool.Lanes)>0 and Pool.RoleIs "Business Role") {
     Foreach (var Lane in Pool.Lanes) {
      umlActors[i] := CreateUMLActor(Lane, Pool, true);
      i++;
     }
    }
   }
  }
  Return umlActors;
}
```

Fig. 2 Pseudo algorithm of actor creation in UML use-case diagram

Pseudo function CreateUmlActor creates new actor umlActor by accepting of unique identifier ID and its identifier name from XPDL element of type pool, lane or participant. These identifiers are further mapped to identifiers of actor umlActor with the same name. Pseudo function has two other parameters: Parent and Generalization. If the value of the parameter Parent is set and value of parameter Generalization is true, then a link to the parent is created (parameter Parent) in order to create association type generalization. If value of parameter Parent is empty and value of parameter Generalization is false then parameter Parent has no associated value, which means that created umlActor will not be included in the link of type generalization with another actor. The return value function is variable umlActor.

Algorithm of actor creation is just one part at the transformation to UML use case diagram. Further transformation algorithms necessary for complete creation of UML use case diagram are: creation of functionalities (UseCase) and creation of associations (Association) among actors and functionalities.

In case of transformation to the model of business entities (UML conceptual class diagram) there are these transformation algorithms used: creation of classes (Class), creation of methods (Methods), creation of attributes (Attributes) and creation of associations (Association) among classes.

4 Conclusion

In this paper we presented a unique solution to automated CIM-PIM transformation in the MDA framework. We devised useful models and notations that represent CIM and PIM level. We searched for automated solution for transformation from business processes modelled in BPMN to UML models, concretely UML use case model and business entity model of IS, represented as conceptual class diagram in UML.

Our recommendations of methods of transformation have been verified by development of IS which have successfully supported two processes at University of Zilina: science and research process and mobility process. The models in BPMN notation (representing the CIM model) were automatically transformed into two types of UML diagrams (representing the PIM model). The examples of the transformation application will be shown during a presentation at the conference.

Our original intent of creating automated transformation of CIM to PIM has not been completely fulfilled. This is because the output UML models still require adjustment by the UML modelling tool that imports the MOF XMI format. It is therefore more appropriate to call this a semi-automated approach to transform CIM to PIM in MDA. The automatic transformation helped to create initialization PIM level that still required manual adjusting of the represented models so that they were formatted for further transformation into PSM models.

A further goal is to determine the precise rules for enlargement of XPDL format so that it can achieve the highest possible degree of automated transformation of business process models in BPMN. This goal applies not only to selected UML models but also to other models required for the successful development of IS. Another objective is to create a graphical editor of BPMN models. This would allow us to enlarge the semantics of BPMN elements of the graphical model and transform these enlargements directly into the XPDL format. One possibility could be the automatic creation of XSLT transformation rules.

Although certain manual changes are still necessary as part of the automatically generated PIM model, this unique method offers a successful solution to the proposed problem of automated CIM-PIM transformation using a systematic approach. This solution helps us to come closer to the vision of the MDA of creating information systems at the highest level of abstraction possible and automating these transformations at all levels until the implementation model-code.

The authors are aware that BPMN is used for modelling business process flow and it is tightly connected with timeline. All events and activities in BPMN are ordered or chained within hierarchy. During the transformation between BPMN and

UML there is a possibility of losing these features. UML is used to model software from different view and that is why it is hard to retain mentioned features (BPMN semantics) in UML after conversion. It is quite difficult to capture not only BPMN semantics but also to capture data semantics in general. In a field of "semantics and knowledge capturing" there is often used an ontology. According to this point of view our next research will focus how the ontology could also bring new possibilities to MDA approach.

Acknowledgments The case study in the paper is a partial outcome of the projects "Quality education by supporting innovative forms, quality research and international cooperation—a successful graduate for practice" ITMS code 26110230090 and project University Science Park of the University of Zilina ITMS 26220220184 supported by the Research and Development Operational Program funded by the European Regional Development Fund.

The authors thank Neven Vrcek for valuable advice and ideas.

References

1. Kang, D., Lee, J., Choi, S., Kim, K.: An ontology-based enterprise architecture. Expert Syst. Appl. **37**, 1456–1464 (2010)
2. Saat, J., Franke, U., Lagersteom, R., Ekstedt, M.: Enterprise architecture meta models for IT/Business Alignment Situations. In: 14th IEEE International Enterprise Distributed Object Computing Conference, pp. 14–23 (2010)
3. Preface/J. Syst. Softw. **85**, 459–462 (2012)
4. Castro, J., Lucena, M., Silva, C., Alencar, F., Santos, E., Pimentel, J.: Changing attitudes towards the generation of architectural models. J. Syst. Softw. **85**, 463–479 (2012)
5. de Boer, R., van Vliet, H.: On similarity between requirements and architecture. J. Syst. Softw. **82**, 544–550 (2009)
6. Roebuck, K.: Model-driven architecture (MDA). ISBN-13: 978-1743044735. ISBN-10: 1743044739. Publisher tebbo (2011)
7. Kherraf, S., Lefebvre, E., Suryn, W.: Transformation from CIM to PIM using patterns and archetypes. In: 19th Australian Conference on Software Engineering, ASWEC 2008, pp. 338–346 (2008)

8. Rodriguez, A., Fernandez-Medina, E., Piattini, M.: Towards CIM to PIM transformation: from secure business processes defined in BPMN to use-cases. In: Lecture Notes in Computer Science, vol. 4714/2007, pp. 408–415. Springer, Berlin (2007). doi:10.1007/978-3-540-75183-0_30
9. Zhang, W., Mei, H., Zhao, H., Yang, J.: Transformation from CIM to PIM: a feature-oriented component-based approach. In: Proceedings of ACM/IEEE Eighth International Conference on Model Driven Engineering Languages and Systems (MoDELS'05), pp. 248–263 (2005)
10. Cao, X., Miao, H., Chen, Y.: Transformation from computation independent model to platform independent model with pattern. J. Shanghai Univ. (Eng. Ed.) **12**(6), 515–523 (2008). doi:10.1007/s11741-008-0610-2
11. Rodriguez, A., García-Rodriguez de Guzman, I., Fernandez-Medina, E., Piattini M.: Semi-formal transformation of secure business processes into analysis class and use case models: an MDA approach. Inf. Softw. Technol. **52**(9) 945–997 (2010)
12. Bousetta, B., El Beggar, O., Gadi T.: A methodology for CIM modelling and its transformation to PIM. J. Inf. Eng. Appl. **3**(2) (2013). ISSN 2224-5782 (print), 2225-0506 (online)
13. Mazon, J., Pardillo, J., Trujillo J.: A model-driven goal-oriented requirement engineering approach for data warehouses. In: Proceedings of the Conference on Advances in Conceptual Modeling: Foundations and Applications, ER Workshops, pp. 255–264, Auckland, New Zealand, (2007)
14. Kardos, M.: CIM-PIM transformation in Model Driven Architecture. Dissertation Thesis, University of Zilina (2011)

Conflictless Task Scheduling Concept

Mateusz Smoliński

Abstract This paper presents new concept of task scheduling without resource conflicts. The proposed task scheduling bases on conflict array, which was created using two binary resource identifiers as representation of task. Those resource identifiers support fast conflict detection between tasks. Preparation of conflict array can be supported by modern GPU. Prepared schedule enable parallel tasks execution without resource conflicts. Elimination of resource conflicts provides deadlock free task processing. Proposed conflict free task scheduling can be used in local and distributed environments with high contention for limited resources. Therefore tasks can be executed without other synchronization mechanisms.

Keywords Rigorous history · Conflict free parallel task execution · Resource conflict free task scheduling · Resource conflict elimination with conflict array · Deadlock avoidance · GPU deadlock detection · Parallel programming

1 Introduction

Modern informational systems perform many tasks in parallel, which can have resource conflict [1, 9]. Regardless of the type of task (process, thread, fiber) any tasks without conflict can be executed in parallel. But conflicted tasks have to use one type of concurrency: cooperative or competitive. In cooperative concurrency tasks work together using communications and synchronization mechanism, but in competitive concurrency tasks do not know each other and are rivals in obtaining resources necessary for further processing [5, 9]. Therefore the development of

M. Smoliński (✉)
Institute of Information Technology, Lodz University of Technology, Wólczańska 215, 90-924 Łódź, Poland
e-mail: mateusz.smolinski@p.lodz.pl
URL: http://it.p.lodz.pl

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_18

tasks for concurrent processing is difficult, because it requires selection of appropriate communication or synchronization methods for exclusive access to global resources. It also requires choice of fine or coarse grain strategy of mutual exclusion in access to resources, which additionally complicates task implementation for concurrent processing. Using any synchronization mechanism like semaphore or lock to synchronize access to global resource introduces new waiting queue for tasks that need to operate on resource in exclusive manner. Incorrect usage of many instances of synchronization mechanism and its queues introduces the potential risk of task deadlock. This problem can be resolved by supervision mechanism that recognizes and prevents transition to unsafe state [4]. As alternative, a task schedule can be prepared to eliminate all resource conflicts between task executed in parallel. Schedule is the cooperation plan for tasks prepared without their internal participation and it facilitates task implementation. However conflictless schedule should be prepared before requested task starts its execution, so any new requested task must be suspended before being included in the execution plan. Suspension of the task generate additional delay before its execution, therefore it has to be as short as possible. In accordance with the principle of fairness in concurrent processing any suspended task has to be resumed in a finite time. Otherwise a task starvation problem occurs [9]. Elimination of resource conflicts provides deadlock free task execution, so discussed concept can be alternative to popular deadlock avoidance methods [2, 4, 8].

1.1 Assumptions About the Tasks and Resources

Parallel task processing can be performed in local or distributed environment. In further discussion only selected environments will be considered, where tasks and resources meets all of conditions:

- task includes sequence of operations, where some of them need local and other require global resources. Number of task operations must be minimal and all of them have to be processed without interaction of external objects (like users). For any task the set of all its resources is known and all required resources exists.
- tasks can be requested from various sources,
- task execution time is not known and is not limited.
- task can be repetitive, many instance of the same task can be requested simultaneously.
- each global exclusive resource has only one instance.
- global resources are accessed by tasks in high contention manner.
- tasks have no preferences in access of global resources.
- the execution order of requested task is not important (requested tasks are independent).

Selection of operations for single task defines a set of necessary global resources and affects task execution time. The definition of task allows to decomposition of set of work like processes or its threads to smaller sub-tasks. Even interactive task can be divided to sub-task, which one realizes a smaller unit of work. In the case of transactions processing, each task can be defined as single transaction.

1.2 Binary Resource Identifiers for Task

In considered environment any task can have many operations which use global resources. Some of them can be only read operation, and other can be write, which change the resource state. In environments with limited number of global resources, task's resources can be represented by two binary identifiers IRW and IR granted by central resource controller. Central resource controller manage resources representations by assigning them a individual bit representation in naming space, when requested tasks uses this resource. When task execution is finished, the central resource controller can unregister all task resources. If fixed bit is set in task IRW identifiers, this means that some of task operations need a global resources, which is represented by this bit. Additionally, if this bit is set in second task resource identifier IR, then no task's operation that changes this resource. So if a resource bit is set in two of identifiers IRW and IR, then task uses this global resource in read-only mode. Task conflict between two tasks can be detected using simple condition (proof is presented in [6]):

$$\left(\mathrm{IRW_i \text{ and } IRW_j}\right) \text{ xor } \left(\mathrm{IR_i \text{ and } IR_j}\right) \neq 0 \tag{1}$$

Length of binary identifiers limits the resource number, than can be used by requested tasks. Model with task representation using binary resource identifiers is scalable, their representation can be extended and with it linearly increase number of operations to detect resource conflict between tasks [6, 7].

2 Conflictless Task Execution

Execution of task without resource conflicts requires to manage a set R of running tasks. Resource conflicts detection have to be performed, before adding a task to set R. During the resource conflict verification between tasks, a new task is suspended. All suspended tasks belong to set S, where $S \cap R = \varnothing$. To minimize a task suspension time, a dedicated structure the conflict array was proposed. This data structure helps to avoid multiple conflict detection between fixed types of tasks.

2.1 Conflict Array for Tasks

The main structure in proposed conflictless task scheduling is conflict array, which stores information about conflicts between different task class. Novelty in conflict array structure is representation all resource conflicts, which is not a representation of resource allocation graph. This structure can be used to monitor tasks, to prevent undesirable resource conflicts occurence.

In first step of creation of conflict array, each task has to be classified using its binary identifiers according to class definition:

$$C_k = \{t_i : IRW_i = IRW_k \wedge IR_i = IR_k\} \tag{2}$$

Additionally for each task class its cardinality is known ident. $n(C_k)$. Each non-empty class has also its own queue FIFO (First In First Out), where all suspended tasks belonging to this class are placed. Each FIFO queue determine resuming order of suspended tasks for class.

Conflict array M is a data structure, that helps to verify conflicts between two task classes C_i and C_j, using condition (1). When value m_{ij} in conflict array $M = [m_{ij}]$ is other than zero, then any task from class C_i has a conflict with any task from class C_j (Fig. 1). According to conflict definition using condition (1) the conflict array is symmetric $M = M^T$, so only bottom or top half of it has to be calculated. Anytime when task from new class was requested, a dimension of conflict array must be increased depending on the number of new class of tasks. Adding new class to conflict array includes verification of resource conflict occurrence with any other existing class. In this time all requested tasks belonging to new class must be suspended, because it can be determine whether their resuming does not cause resource conflict with one of executed tasks. Increase a conflict array dimension and effective conflicts calculation between task class can be performed by modern GPU architectures.

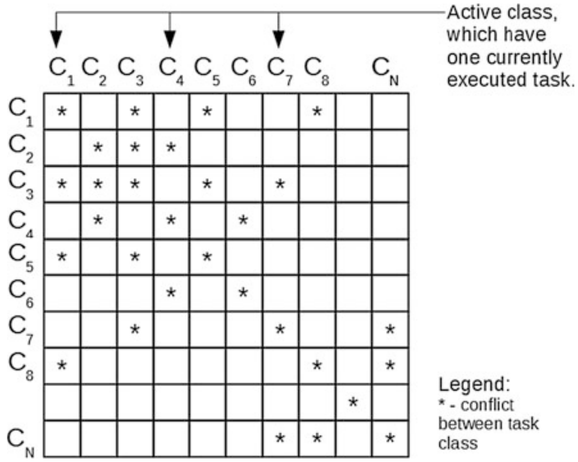

Fig. 1 Conflict array for task class

2.2 Conflictless Task Excecution

Any new requested task t_i, must be verified whether its execution will not cause resource conflict. If $t_i \in C_k$ and $\sum_{t_j \in R} m_{kj} = 0$ then new task execution is started. Otherwise new task will be suspended (ident. $t_j \in S$) until all running conflicted tasks will be finished (Fig. 2). Running sequence of suspended task are determined by the conflictless task schedule, which must be prepared in advance for each executed task from set R. The need to prepare many conflictless schedules as the cardinality of the set R is due to the fact, that execution time of any task $t_i \in R$ is not known. In preparation of schedules a modern GPU can be used.

Another important rule in conflictless task processing determine when finished task can resume another task from the same class C_k. If task $t_i \in R, t_i \in C_k$ finishes another tasks $t_j \in C_k$ from the same task class can be running only if there is no other suspended conflicted task from other class, what can be verified by condition:

$$\sum_{l \neq k} m_{lk^*} n(C_l) = 0 \tag{3}$$

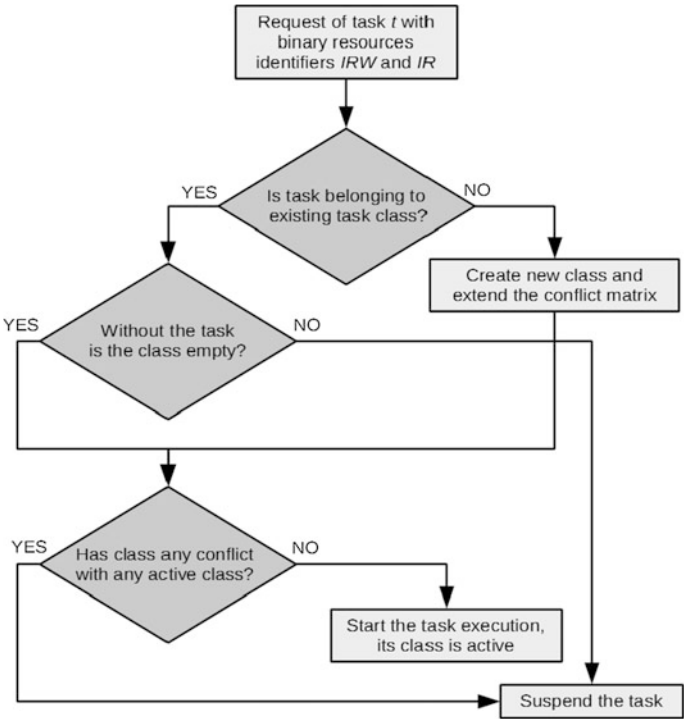

Fig. 2 Procedure for execution of requested task

This rule avoids the suspended task starvation, when continuous task execution from single class block execution of tasks form other conflicting class.

2.3 Conflictless Task Schedule Generation

Preparation of conflictless task schedule for situation when execution of $t_k \in R \cap C_k$ is finished begins with finding of subset S_k of suspended task where:

$$\begin{aligned} S_k = & \left\{t_i \in S \cap C_q, t_j \in S \cap C_p : p \neq q \wedge m_{kq} \neq 0 \wedge m_{kp} \neq 0 \wedge m_{pq} = 0\right\} \\ & \cap \left\{t_i \in S : m_{ik} \neq 0 \wedge \forall_{t_r \in R \setminus \{t_k\}} m_{ir} = 0\right\} \end{aligned} \tag{4}$$

The set S_k is a group of task, which anyone belongs to different class C and is the longest suspended task in FIFO queue class. All task belonging to S_k can be resumed after finished execution of task t_k, then $S_k \subset R$. Minimization of total tasks suspension time requires to find most numerous set S_k before task t_k finishes. This most numerous subset of S_k is marked as S'_k. The designation the S'_k can be done by elimination of all conflicted tasks from D_k:

$$D_k = \left\{t \in S \cap C_q, \forall_{t_r \in R \setminus \{t_k\}} : \; m_{kq} \neq 0 \wedge m_{kr} = 0\right\} \tag{5}$$

where the conflictless schedule $S'_k \subseteq D_k$. Detection of conflict between tasks $t_i, t_j \in D_k$ can be done using conflict array by single read operation check. If $t_i \in C_p$ and $t_j \in C_q$ and $m_{pq} \neq 0$ then there is a resource conflict between tasks t_i and t_j.

In situation, when two different subset of $S'_j, S'_i \subseteq Dk$ with identical cardinality were found, then only one of them can be selected as group of tasks to conflictless execution. With a random number generator this selection can be perform by random. Alternatively counter of selections of S'_k can be keep in memory, and choice of S_k base on lower value of its counter. This provides the fairness of selection of S'_k when many subset of D_k have the same cardinality. Additionally having counters of earlier selections for any not empty S_k can be used to provide fairness in next selection of S_k, even when its cardinality is not the highest one. To determine set S'_k a dynamic verification of conflicts between two tasks using the conflict array must be performed. Each conflict between suspended and active tasks reduces cardinality of D_k and also can introduce alternative set $S_k \subseteq D_k$.

Conflictless schedules preparation requires analysis of conflict between suspended tasks and between suspended and active tasks. Any change in sets of suspended or active task can change previously prepared conflictless schedules and even number of conflictless schedules. This results from direct and indirect resource conflicts between suspended and running tasks.

In example situation presented on Fig. 3 in active tasks set R are three executed tasks from class: C_1, C_4, C_7. Time of its execution is not known, so conflictless schedule has to be prepared for each case of completion one of this task. The

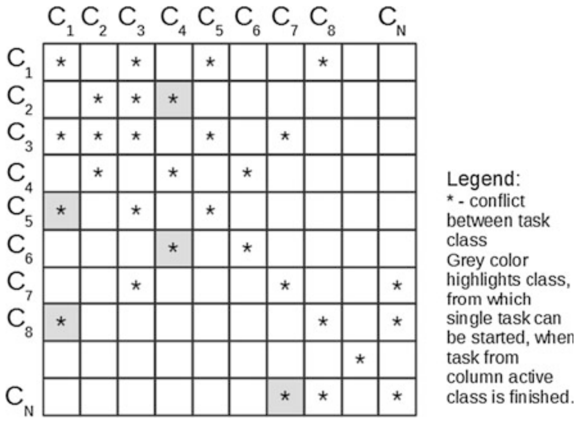

Fig. 3 Conflict array with selected class for preparation conflictless schedules

cardinality of set R determine the number of conflictless schedules to prepare. Analysis of conflict array supplies necessary information for conflictless schedule preparation. If task from class C_1 will be finished only task from class C_5 and C_8 can be started and executed in parallel, assuming that queue of those class are non-empty. Note that task from class C_3 can not be started when $t \in R \cap C_1$ finishes execution, because this class has conflict with other running task from class C_7. If cardinality of each class C_3, C_5, C_8 is zero and $n(C_1) > 0$, only then next task from class C_1 can be started. If running task from class C_4 is finished only tasks from non-empty queue of class C_2 and C_6 can be started otherwise if $n(C_2) = 0$ and $n(C_6) = 0$ then next task from C_4 is started. While task from class C_7 is finished only task from class C_N can be started, but if $n(C_N) = 0$ and $n(C_3) = 0$ only then execution of next task from C_7 can be started. For conflict array presented on Fig. 3 all prepared conflictless schedule are suitable to specific conditions. If active task from R will be finished, then prepared schedule can be used to provide conflictless execution of tasks.

In situation presented in Fig. 3 efficient creation of there conflictless schedules is crucial for task execution without resource conflict. For any task $t_k \in R \cap C_k$ a conflict schedule should be calculated and for any two tasks belonging to D_k conflict verification is required. All calculations associated with conflictless schedules preparation can be performed in parallel, because resource conflict detection between tasks required only single read operation from conflict array. In situation when some running task $t_i \in C_k$ finishes and condition 3 is fulfilled then another task belonging to the same class $t_j \in C_k$ can be executed (Fig. 4). Selection of task from class C_k, which execution is started is determined by class FIFO queue.

Other earlier prepared schedules are useless anytime when cardinality of set R of active tasks is increased. Global resources used by active tasks can create conflict with tasks included in prepared earlier schedules, so they do not assure execution without resource conflicts. Therefore when cardinality of set R is increased all conflictless schedules need to be prepared again. Conflictless schedule calculations for situation when active task execution finishes have to be prepared often and effectively. Parallel calculations improve efficiency of preparation many conflictless

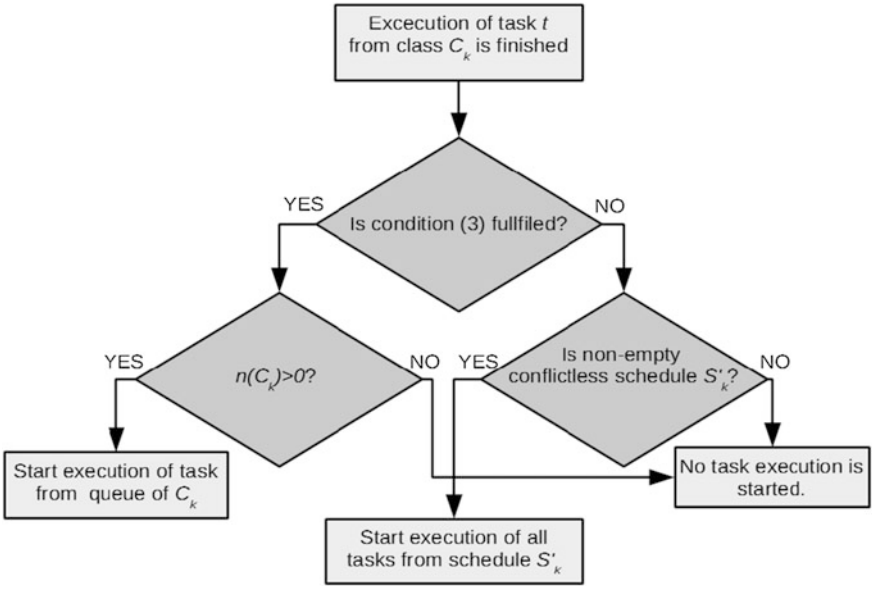

Fig. 4 Choosing tasks to execution without resource conflicts

schedules, but due to minimization of time when tasks are suspended all those calculations should not be performed in the same environment where tasks are executed.

2.4 *Effective GPU Calculations of Conflictless Schedules*

Proposed scheduling strategy gives increase the number of task processed in parallel without resource conflicts, however preparation of conflictless task schedules S'_k for any $t_k \in R$ are time-consuming calculations. In any S'_k number of task conflict verification is calculated as number of 2-elements combinations from set D_k. Therefore GPU it seems to be appropriate to find conflictless task schedule, because at the same time may verify the various subset of D_k. The modern GPU provides a isolated environment, that is not used by tasks, therefore GPU can also be used to calculate conflict array. Isolation of task processing environment from schedule calculation environment theoretically allows to prepare a conflictless schedule S'_k before the end of task $t_k \in R$. Number of conflictless schedules that need to be prepared is limited by cardinality of R. Main benefit from the designation of conflictless schedule in advance is no additional delay in parallel task processing because all tasks from S'_k can be started immediately after t_k is finished. The modern GPU architecture uses SIMT (Single Instruction Multiple Threads), which is a abstraction of SIMD according to Flynn Taxonomy [3]. Threads have access to

GPU shared memory. Therefore fast determination of task schedules without resource conflict needs to maintain in GPU shared memory a conflict array, which helps to detect conflict between tasks. Single verification by thread of conflict occurrence between task class require only read of single value from conflict array. Even in situation, when only a subset of $S_k \subset S'_k$ is determined by GPU, all tasks from this subset can be executed in parallel without conflicts. In situation where execution time of $t_k \in R$ is inadequate to determine S'_k, its subset can used to designation of tasks for conflictless execution. In the simplest case, the subset S_k may have only one task.

Each change of number of task classes requires extension of conflict array, which create the need of new resource conflict detection for task from new class. In the case of fixed number of task class, a conflicted array and all conflictless schedules can be calculated only once. This assumptions is met only in limited environment of task processing.

3 Conclusions

Modern systems and applications use parallel task processing, which provides effective resource utilization. The proposed parallel task processing without resource conflict facilitate task creation, especially in environments with high contention of resources. When conflictless scheduling is used there is no need of using synchronization mechanisms, which simplifies the implementation of non-interactive tasks. Without usage of synchronization mechanisms task execution time will be shorter, because for started task all required global resources will be available which is assured by prepared schedule without resource conflicts. The conflicless schedule can be determined only if for all tasks, which global resources are known in advance. Proposition of conflictless task scheduling bases on representation of global task resources as binary resources identifiers, which enables fast resource conflict detection between two selected tasks. Grouping tasks in class according to identical binary identifiers and using class conflicts array structures improves preparation of conflictless task schedule. If the time of each executed task is unknown the number of conflictless schedules that have to be prepared is related to number of currently executed task in parallel. Due to the fact that conflictless schedule has to be prepared as fast as possible a GPU can be applied to all calculation, especially in the case of an significant number of task class. The GPU can also be used to fast preparation and maintaining of conflict array.

Task processing without resource conflicts excludes also deadlocks, which are a significant problem in concurrent task processing. Conflictless schedule eliminates the occurrence of wait-for cycle between tasks. In the case of transaction processing a conflictless schedule eliminates the transaction rollback caused by its isolation. Reduction of rollbacks allows for more efficient use of resources in transaction processing. Conflictless scheduling can be used in local and distributed environments i.e. in systems or applications integration.

Further work will include the implementation of GPU algorithm for conflictless tasks scheduling and practical verification for various group of requested task with maximization of task executed in parallel without resource conflicts.

References

1. Bernstein, P.A., Newcomer E.: Principles of Transaction Processing, 2nd edn. Morgan Kaufman (2009)
2. Ferenc, B.: An efficient deadlock avoidance technique. IEEE Trans. Comput. **39**(7), 882–888 (1990)
3. Flynn, M.J., Rudd, R.W.: Parallel architectures. ACM Comput. Surv. **28**(1), 67–70 (1996)
4. Nazeem, A., Reveliotis, S.: Maximally permissive deadlock avoidance for resource allocation systems with R/W-locks. Discrete Event Dyn. Syst. **25**(1–2), 31–63 (2015)
5. Silberschatz, A., Galvin, P. B., Gagne, G.: Operating system concepts. Wiley John Sons, 9 edition (2012)
6. Smoliński M.: Rigorous history of distributed transaction execution with systolic array support. XXXI ISAT Conference, Information Systems Architecture and Technology New Developments in Web-Age Information Systems, Oficyna Wydawcznicza PW, pp. 235–254 (2010)
7. Smoliński M.: Scheduling system excluding distributed transaction conflicts with systolic support. Dissertation (2010)
8. Stetsyura, G.G.: Fast decentralized algorithms for resolving conflicts and deadlock resource allocation in data processing and control systems. J. Autom. Remote Control **71**(4), 708–717 (2010)
9. Tanenbaum, A., Bos H.: Modern Operating Systems, 4th edn. Prentice Hall (2014)

Joint Train Rescheduling and Track Closures Planning: Model and Solution Algorithm

Grzegorz Filcek, Dariusz Gąsior, Maciej Hojda and Jerzy Józefczyk

Abstract Selected complex railway track maintenance problem, composed of train rescheduling and track closures planning, is considered. It is assumed that a set of obligatory and facultative planned track closures is given for a specified railway network with the operating railway traffic managed by a current timetable. A new timetable is derived including all obligatory closures as well as a sub-set of facultative closures to maximize a combined performance index evaluating not only a number of accepted facultative closures but also a traffic quality expressed by a similarity of current and new timetables. A proposed heuristic algorithm is based on the decomposition of the problem into several connected sub-problems solved by solvers together with greedy algorithms. The algorithm responsible for the generation of railway sub-networks on the basis of the railway network is described in detail. Other algorithms are characterized in a general way. An illustrative numerical experiment is also given.

Keywords Optimization · Modelling · Rescheduling · Railway track maintenance

G. Filcek (✉) · D. Gąsior · M. Hojda · J. Józefczyk
Department of Informatics, Wroclaw University of Technology,
Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland
e-mail: Grzegorz.Filcek@pwr.edu.pl

D. Gąsior
e-mail: Dariusz.Gasior@pwr.edu.pl

M. Hojda
e-mail: Maciej.Hojda@pwr.edu.pl

J. Józefczyk
e-mail: Jerzy.Jozefczyk@pwr.edu.pl

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*,
Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_19

1 Introduction

Different decision making problems for railway transportation systems are investigated and corresponding results are reported in the literature, e.g. [1, 2]. Resulted solution methods and algorithms along with their applications in the form of conforming decision support systems contribute to more effective performance of such systems. Two selected problems from this area are considered in the paper, i.e.: planning of truck closures and rescheduling of trains. The former one consists in the determination of time periods when corresponding tracks as a part of a whole rail network are not available for trains. In a consequence, some trains can change their departure times as well as some trains have to be cancelled or moved to other routes. Thus, the latter problem arises, referred to as train rescheduling, when a new timetable for trains operating at a given railway network should be made; strictly speaking a current one should be modified. In the paper, both problems are treated and formulated as a whole. The train rescheduling attracts substantial researchers' attention. The majority of cases deal with this issue in response to a single, unforeseen event that makes impossible further realization of the transportation plan expressed by the current timetable. The track maintenance planning problem has attracted a little attention so far. The literature studies different models describing the problem as well as a variety of solution methods and techniques. The most common optimization models proposed are Integer Linear Programming [3–6], Mixed Integer Programming [7–10] and Constraint Programming [10]. The models proposed differ mainly in the representation and assumptions on railway infrastructure elements. There are problems dealing with no constrains concerning infrastructure like in [7], taking one [5], two [11] or any number of tracks connecting two railway stations, or even assuming, that a track is composed from blocks [8], which is the closest to real-world railway systems. Apart from the considered number of tracks, the number of trains traveling between two railway stations in the same period of time constitutes different decision making problems [3]. Moreover, different criteria are used to evaluate decisions, e.g.: total delay of all passengers [7], total delay of trains [3, 4, 5, 11, 8, 9, 10, 12, 6], total cost of delays [10]. Some works consider more than one criterion, e.g. [10]. To solve formulated problems, authors used solvers like CPLEX [8], methods and algorithms like Branch and Bound [3], the Lagrange relaxation [4], and heuristics like Genetic Algorithm [11], SAP [12], Simulated Annealing and Tabu Search [6].

More general track maintenance scheduling problem is investigated in [13–16] where the minimization of the difference between current and new timetables is taken into account. The train runs rescheduling in the presence of the track maintenance planning treated as sub-problems is considered in [17, 18]. A simple train scheduling is solved in [17] after converting each track closure into a new additional train run. Both sub-problems are solved jointly in [18] by means of Problem Space Search.

Unlike the majority of existing literature, the joint planning problem is investigated when both track closures and new timetables for trains are to be determined.

The new problem formulation as a nonlinear integer programming problem is given in Sect. 2. The outline of the original solution method is presented in the next section. The results of a simple numerical experiment followed by conclusions complete the paper.

2 Problem Formulation

Let us assume that the railway network under consideration is represented by an undirected multigraph $\mathbf{G} = <\mathbf{V}, E>$ where $\mathbf{V} = \{0, 1, 2, \ldots, V\}$ and $E = [e_{i,j,k}]_{i,j \in \mathbf{V}, k \in \mathbf{K}}$ are a set of railway nodes and an incidence matrix describing tracks among nodes, respectively. The positive valued elements of **V** refer to real-world railway nodes, i.e. railway stations and branch posts outside railway stations. The index '0' denotes a virtual node being the beginning and the end of each train run. The binary element $e_{i,j,k}$ of E is equal 1(0) if there exists track k from i to j (otherwise) where $\mathbf{K} = \{0, 1, 2, \ldots, K\}$ is a given set of track numbers and K is the maximum track number between any pair of nodes ($k = 0$ denotes the possibility of substitute transport). Additionally, we denote by $d_{i,j}$ the distance between i and j, where $d_{0,j} = d_{j,0} = 0$ for every i. Index $p \in \mathbf{P} = \{1, 2, \ldots, P\}$ as an element of P considered train runs, denotes the current train run represented by variable $r_p = <\rho_p, [(\underline{\tau}_{i,p}, \overline{\tau}_{i,p})_{i \in \mathbf{V}}]>$ where $\rho_p = \left[\rho_p^{i,j,k}\right]$ is the path of railway tracks ($\rho_p^{i,j,k} = 1(0)$ means that train p can go from i to j by track k (otherwise)), and $\underline{\tau}_{i,p}$, $\overline{\tau}_{i,p}$ are arrival and departure time at node i, respectively. All variables r_p form a timetable $R = [r_p]_{p \in \mathbf{P}}$.

The set **N** of N planned closures $z_n, n \in \mathbf{N} = \{1, 2, \ldots, N\}$ is given. Among all closures, M obligatory closures z_n, n = 1, 2, …, M and $N - M$ optional closures z_n, $n = M + 1, M + 2, \ldots, N$ are distinguished. Each closure is defined by the track number $\sigma_n^{i,j,k}, i,j \in \mathbf{V}, k \in \mathbf{K} \backslash \{0\}$ (element of matrix $\sigma_n = [\sigma_n^{i,j,k}]$) not available for trains during the closure, the duration of closure T_n as well as three starting times for the closure: $\underline{t}_n$, $\overline{t}_n$ and t_n^* as the earliest, the latest and the suggested one, respectively. The auxiliary notation for variables and data is given in Table 1.

We are seeking for a subset of optional closures to be performed and for a new timetable comprising a new subset of train runs. To this end, the following decision variables are specified:

$y_n, n \in \mathbf{N}$ where y_n = 1(0) if the corresponding optional closure is performed (otherwise), $\Delta t_n, n \in \mathbf{N}$—positive shift of the earliest closure starting time, $x_p, p \in \mathbf{P}$ where x_p = 1(0) if pth train run is present in the new timetable (otherwise), $R' = [r'_p]_{p=1,2,\ldots,P}$, $r'_p = <\rho_p, [(\underline{\tau}_{i,p} + \Delta\underline{\tau}_{i,p}, \overline{\tau}_{i,p} + \Delta\overline{\tau}_{i,p})_{i=0,1,2,\ldots,V}]>$. It is assumed that the time horizon of the considered problem is limited to the interval [0, T] where $T = \max(T_1 + \max_i T_{ARR}(i), T_2)$, and T_1, T_2 are respectively the latest arrival of a train to the destination railway station according to timetable R, the completion time of the latest closure. All mentioned and resulted time variables have to belong to this interval, however time shifts $\Delta\underline{\tau}_{i,p}$, $\Delta\overline{\tau}_{i,p}$ can be negative.

Table 1 Auxiliary notation

$c_i, i \in \mathbf{V}\backslash\{0\}$	Capacity of node i (the maximum number of trains)
$\Delta\underline{\tau}_{i,p}$	Arrival time shift in a new timetable
$\Delta\overline{\tau}_{i,p}$	Departure time shift in a new timetable
$K_{i,j} = \sum_{k\in\mathbf{K}\backslash\{0\}} e_{i,j,k}$	Number of tracks between i and j
$\hat{\mathbf{P}} \subseteq \mathbf{P}$	Subset of train runs obligatory present in a new timetable
$\mathbf{V}_{ARR}, \mathbf{V}_{DEP} \subseteq \mathbf{V}$	Subset of stations with given maximum arrival lateness $T_{ARR}(i), i \in \mathrm{V}_{ARR}$, maximum departure lateness $T_{DEP}(i), i \in \mathrm{V}_{DEP}$, respectively
$\tau_{\max}$	Maximum train standing time (the same for every station and train)
$\tau_{\min,i,p}$	Minimum standing time of pth train at ith station
$\hat{\tau}_{\max}$	Maximum time extension for a distance D of the train run in relation to timetable R (e.g. D = 100 km)
$\tilde{\tau}_{\max}$	Maximum total time extension of the train run in relation to timetable R

Constraints have been imposed on the decision variables to have a feasible solution, which concern, in particular, the new timetable for train runs: obligatory closures have to be performed and all closures starting times are feasible,

$$y_n = 1, \quad n = 1, 2, \ldots, M, \quad \underline{t}_n + \Delta t_n \le \overline{t}_n, \quad \Delta t_n \ge 0, \quad n \in \mathbf{N} \tag{1}$$

train runs from set $\hat{\mathbf{P}}$ are present in the new timetable,

$$x_p = 1, \quad p \in \hat{\mathbf{P}} \tag{2}$$

standing times of trains at stations are limited by the minimum and maximum feasible values,

$$\tau_{\text{min,i,p}} \le \overline{\tau}_{p,i} + \Delta\overline{\tau}_{p,i} - \Delta\underline{\tau}_{p,i} - \underline{\tau}_{p,i} \le \tau_{max}, \quad i \in \mathbf{V},\ p \in\ \mathbf{P} \tag{3}$$

run times between adjacent nodes are fixed,

$$\sum_{k\in\mathbf{K}} \rho_p^{i,j,k} (\Delta\underline{\tau}_{j,p} - \Delta\overline{\tau}_{i,p}) = 0, \quad i,j \in \mathbf{V}, \quad p \in \mathbf{P} \tag{4}$$

departure and arrival latenesses at selected stations are bounded,

$$\begin{aligned} &\Delta\overline{\tau}_{i,p} \le T_{DEP}(i,p), \quad i \in \mathbf{V}_{DEP}, \\ &\Delta\underline{\tau}_{i,p} \le T_{ARR}(i,p), \quad i \in \mathbf{V}_{ARR}, \quad p \in \mathbf{P} \end{aligned} \tag{5}$$

total time extensions of each train run in relation to timetable R is limited,

$$\Delta\underline{\tau}_{0,p} - \Delta\bar{\tau}_{0,p} \leq \tilde{\tau}_{max}, \quad p \in \hat{\mathbf{P}} \tag{6}$$

capacities of stations are limited (only in artificial node '0' capacity is unlimited),

$$\sum_{j\in\mathbf{V},\, k\in\mathbf{K}\backslash\{0\}} x_p \max(\rho_p'^{j,i,k},\ \rho_p'^{i,j,k})(1 + \sum_{q\in\bar{\mathbf{P}}_{i,p}} \left(\sum_{l\in\mathbf{V},\, m\in\mathbf{K}\backslash\{0\}} x_q \max(\rho_q'^{i,l,m}, \rho_q'^{i,l,m}) \right)) \leq c_i,$$
$$i \in \mathbf{V}\backslash\{0\},\ p \in \mathbf{P} \tag{7}$$

where

$$\begin{aligned}\bar{\mathbf{P}}_{i,p} = \{q \in \{1, 2, \ldots, P\} : \underline{\tau}_{i,p} + \Delta\underline{\tau}_{i,p} \leq \bar{\tau}_{i,q} + \Delta\bar{\tau}_{i,q} \leq \bar{\tau}_{i,p} + \Delta\bar{\tau}_{i,p} \\ \vee\ \underline{\tau}_{i,p} + \Delta\underline{\tau}_{i,p} \leq \underline{\tau}_{i,q} + \Delta\underline{\tau}_{i,q} \leq \bar{\tau}_{i,p} + \Delta\bar{\tau}_{i,p}\}\end{aligned} \tag{8}$$

is a subset of train runs waiting at station i in time coincided with the run of train p, at most one train can occupy one track at every time moment,

$$\begin{aligned}&\min\{\rho_p^{i,j,k} x_p(\bar{\tau}_{i,p} + \Delta\bar{\tau}_{i,p}) + \rho_p^{j,i,k} x_p(\underline{\tau}_{i,p} + \Delta\underline{\tau}_{i,p}), \rho_q^{i,j,k} x_q(\underline{\tau}_{j,q} + \Delta\underline{\tau}_{j,q}) \\ &+ \rho_q^{j,i,k} x_q(\underline{\tau}_{i,q} + \Delta\underline{\tau}_{i,q})\} \leq \max\{\rho_p^{i,j,k} x_p(\bar{\tau}_{i,p} + \Delta\bar{\tau}_{i,p}) + \rho_p^{j,i,k} x_p(\bar{\tau}_{j,p} + \Delta\bar{\tau}_{j,p}), \\ &\rho_q^{i,j,k} x_q(\bar{\tau}_{i,q} + \Delta\bar{\tau}_{i,q}) + \rho_q^{j,i,k} x_q(\bar{\tau}_{j,q} + \Delta\bar{\tau}_{j,q})\},\ \ p, q \in \mathbf{P},\ \ i, j \in \mathbf{V},\ \ k \in \mathbf{K}\backslash\{0\}\end{aligned} \tag{9}$$

$$\rho_p^{i,j,k}(\bar{\tau}_{i,p} + \Delta\bar{\tau}_{i,p}) \leq \rho_p^{i,j,k}(\underline{\tau}_{j,p} + \Delta\underline{\tau}_{j,p}),\ \ p, q \in \mathbf{P}, \quad i, j \in \mathbf{V}, \quad k \in \mathbf{K}\backslash\{0\} \tag{10}$$

every pair of closures has to be disjoint,

$$\begin{aligned}&\min\{y_n(\underline{t}_n + \Delta t_n + T_n), y_m(\underline{t}_m + \Delta t_m + T_m)\} \\ &\quad \leq \max\{y_n(\underline{t}_n + \Delta t_n), y_m(\underline{t}_m + \Delta t_m)\} : n, m \in \mathbf{N} : n \neq m \wedge \sigma_n^{i,j,k} = \sigma_m^{i,j,k}\end{aligned} \tag{11}$$

no train can occupy a track during a closure,

$$\begin{aligned}\min\{\rho_p^{i,j,k} x_p \underline{\tau}_{j,p}, y_n(\underline{t}_n + \Delta t_n + T_n)\} \leq \max\{\rho_p^{i,j,k} x_p \bar{\tau}_{i,p}, y_n(\underline{t}_n + \Delta t_n)\}, \\ p \in \mathbf{P},\ \ n \in \mathbf{N},\ \ i, j \in \mathbf{V},\ \ k \in \mathbf{K} : \sigma_n^{i,j,k} = 1\end{aligned} \tag{12}$$

arrival time of every train at every station must not be later than corresponding departure time

$$\underline{\tau}_{i,p} + \Delta\underline{\tau}_{i,p} \leq \bar{\tau}_{i,p} + \Delta\bar{\tau}_{i,p}, \quad i \in \mathbf{V}\backslash\{0\},\ \ p \in \mathbf{P}. \tag{13}$$

We propose the following performance index to evaluate the decisions made:

$$Q = \alpha \sum_{n=M+1}^{N} y_n + \sum_{p \in \mathbf{P} \setminus \hat{\mathbf{P}}} x_p - \beta \sum_{i \in \mathbf{V}, p \in \mathbf{P}} \left|\Delta\bar{\tau}_{\mathrm{i,p}}\right| \tag{14}$$

where $\alpha > P - \hat{P}, \beta < 1/[P(I+1)T]$ and $\hat{P}$ is the size of $\hat{\mathbf{P}}$.

Two first elements of (13) evaluate respectively the number of closures and the number of train runs inserted into the new timetable. The last element assesses the total difference between departures in both timetables. Such a form of the performance index ensures the hierarchy of the decisions, i.e. the acceptance of optional closures is the most important, whereas the differences of departure times in the timetables are the least important.

In a consequence, the following optimization problem **P0** is solved. For given: $\mathbf{G}, \mathbf{N}, \hat{\mathbf{P}}, \mathbf{V}_{ARR}, \mathbf{V}_{DEP}, \mathbf{K}, \mathbf{P}, R, M, D, \tau_{\max}, \hat{\tau}_{\max}, \tilde{\tau}_{\max},$ $T_{\mathrm{ARR}}(i,\ p)$, $T_{DEP}(i,\ p)$, $\tau_{\min,i,p}$, ρ_p, $i \in \mathbf{V}, p \in \mathbf{P}$, T_n, $\underline{t}_n$, $\bar{t}_n$, $\bar{t}_n^*$, σ_n, $n \in \mathbf{N}$, decision variables Δt_n, $y_n, n \in \mathbf{N}$, x_p, $\Delta\underline{\tau}_{i,p}$, $\Delta\bar{\tau}_{i,p}, i \in \mathbf{V}, p \in \mathbf{P}$ are to be determined with respect to constrains (1)–(13) in such a way to maximize (14).

3 The Solution Algorithm

The decision making problem **P0** formulated in the previous section is a complex, multi-variable, multi-constraint optimization problem. This complexity encourages approaches which forgo obtaining an exact solution and instead focus on providing satisfactory results in reasonable time. For this purpose, a series of smaller and simpler supplemental problems is introduced. These problems form a hierarchy for which the solution to a parent problem is dependent on the solutions of its child problems. Conversely, the formulation of each child problem depends on the determination of decision variables of its parent.

The highest level problem **P1** is responsible for determining the set of accepted (active) closures including all obligatory closures. Next problem **P2** consists in choosing a sub-set of train runs for inclusion into the new timetable. Problem **P3** deals with the calculation of decision variables $\Delta t_n, n \in \mathbf{N}$, $\Delta\underline{\tau}_{i,p}$, $\Delta\bar{\tau}_{i,p}$,. These variables are not attained straightforwardly, but are a result of applying additional lower-level decompositions. Firstly, the railway network is divided into smaller sub-networks (sub-problems **P3.*i***, i = 1, 2, …, η where η is a number of sub-networks). The aforementioned decision variables are sought separately for individual sub-networks (individual sub-problems **P3.*i***). Additionally, every sub-problem **P3.*i*** is divided into three lower-level parts consisting of a track ordering sub-problem (**P3.*i*.1**), a station ordering sub-problem (**P3.*i*.2**) and a time shift resolution sub-problem (**P3.*i*.3**) with the following contents of the respective resulting decisions: the order of runs and closures, the order of entering stations by trains and the values of time moments when closures begin as well as when trains leave/arrive at the station nodes. Sub-problems **P3.*i*.1**, **P3.*i*.2** and **P3.*i*.3** are respectively Mixed Integer Linear Programming (MILP) and Linear Programming (LP). Solution algorithms for **P1** and **P2** are presented in this paper in a general form unlike the procedure for decomposing **P3** into sub-networks which is described more thoroughly.

A multi-level algorithm which gives, as its final result, a solution to **P0** is designed as a series of algorithms for solving supplemental problems **P1**–**P3** described as follows. The greedy algorithm **A1** solving **P1** sequentially accepts individual closures for launching and adds them one by one to a set of accepted closures in such a way to ensure the generation of feasible timetables. The feasibility of the current closure is checked by algorithm **A2** solving **P2**. Algorithm **A1** decides on the inclusion of the current closure into the set of accepted closures according to information returned by **A2**. However, it is necessary to cover all obligatory closures. On the analogous basis, algorithm **A2** sequentially, i.e. one by one, forms the set of accepted trains for the new timetable. It takes into account information returned by algorithm **A3** solving **P3** which, in turn, is launched at every iteration of **A2**. Algorithm **A3** assumes the decomposition of the railway network performed by the Track Grouping Algorithm (**TGA**). Finally ordering problems **P3.*i*.1**, **P3.*i*.2** are solved with the combined use of a cheapest insertion algorithm with an LP problem solved in every iteration. Those LP problems as well as the problem **P3.*i*.3** are solved with the use of a numerical solver 'lpsolve' in the algorithm solving problem **P3.*i*** for the *i*th subnetwork (Fig. 1).

Algorithm **TGA** is parameterized and can work in three modes. In the first mode, the solution is build based on the association of the tracks to the railway line category, in the second mode, the association of the tracks to the management area, and in third mode, the aggregation method is used.

The aggregation method allows making groups of tracks which may be seen as a railway subnetwork. It is proposed to apply the agglomerative aggregation algorithm [19]. It is assumed that each track constitutes a separate group at the beginning. The distance function of tracks is given by the following formula:

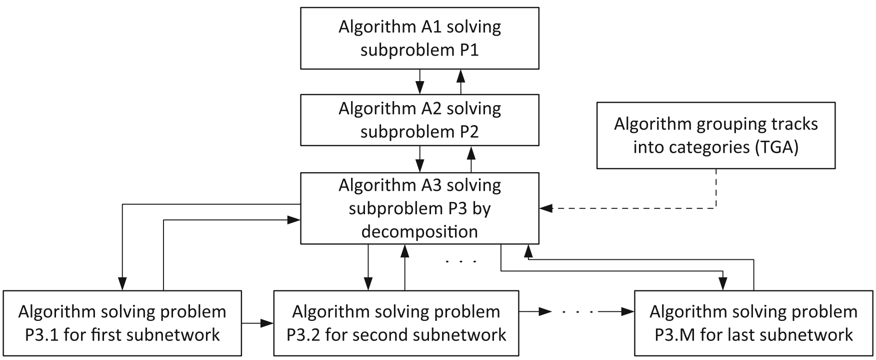

Fig. 1 Relation of sub-problems in the solution algorithm

$$\delta(k_1,k_2) = \begin{cases} w_1(\delta_1(k_1,k_2))^{-1} + w_2(\delta_2(k_1,k_2))^{-1}, & \text{if} \quad k_1 \text{ and } k_2 \text{ have a common vertex,} \\ \infty, \quad \text{otherwise} \end{cases}$$

where k_1, k_2—any two tracks, $\delta_1(k_1, k_2)$—number of train runs using both tracks (i.e. k_1 and k_2), $\delta_2(k_1, k_2)$—number of train runs which neither use track k_1 nor k_2 and $w_1 > 0$ and $w_2 > 0$—arbitrary chosen weights (in particular $w_1 = 1$, $w_2 = 0$).

Let us denote the *i*th group of tracks as Γ_i. Assume a distance function between groups of tracks, e.g.:

(a) $\Delta(\Delta\Gamma_1, \Gamma_2) = \min\{\delta(k_1,k_2) : k_1 \in \Gamma_1, k_2 \in \Gamma_2\}$,
(b) $\Delta(\Delta\Gamma_1, \Gamma_2) = \max\{\delta(k_1,k_2) : k_1 \in \Gamma_1, k_2 \in \Gamma_2\}$,
(c) $\Delta(\Gamma_1, \Gamma_2) = \frac{1}{|\Gamma_1||\Gamma_2|} \sum\limits_{k_1 \in \Gamma_1, k_2 \in \Gamma_2} \delta(k_1,k_2)$.

Let us assume the following auxiliary notation: Γ_i^j—*i*th group of tracks obtained in the *j*th algorithm iteration, Ξ—maximal size of the group. Now, the aggregation algorithm is as follows:

1. Initiate value M with a number of tracks in the whole railway network.
2. Create M groups: $\Gamma_1^0, \Gamma_2^0, \ldots, \Gamma_M^0$ in the following way: Γ_i^0 for $i = 1, 2, \ldots, M$.
3. Initiate values $j{:} = 0$.
4. If any two groups may not be aggregated and the stop condition is not satisfied, go to 5.
5. Find indexes of two nearest groups, i.e. find indexes q_1^* and q_2^* such that $q_2^* > q_1^*$ and the value $\Delta(\Gamma_{q_1^*}^j, \Gamma_{q_2^*}^j)$ is the smallest possible: $\Delta(\Gamma_{q_1^*}^j, \Gamma_{q_2^*}^j) = \min\limits_{q_1, q_2 : \left|\Gamma_{q_1}^j\right| \le \Xi, \left|\Gamma_{q_2}^j\right| \le \Xi} \{\Delta(\Gamma_{q_1}^j, \Gamma_{q_2}^j)\}$.
6. Set: $j{:} = j + 1$, $\Gamma_q^j{:} = \Gamma_q^{j-1}$ for $q < q_2^*$ and $q \neq q_1^*$, $\Gamma_{q_1^*}^j := \Gamma_{q_1^*}^{j-1} \cup \Gamma_{q_2^*}^{j-1}$, $\Gamma_q^j{:} = \Gamma_{q+1}^{j-1}$ for $q_2^* \le q \le M, M := M - 1$.
7. Go to 3.

Number of groups or maximal distance between two nearest groups can serve as the stop condition. Such decomposition algorithm may be executed once at the beginning of the procedure solving track closures planning problem or each time when sub-problem **P3** is going to be solved.

4 Numerical Experiment

Let us now provide an empirical evaluation of the innermost routine of the railway track maintenance algorithm (i.e. P3.i). We operate on a set of 47 nodes derived from the “Kielce” region of the Polish Railway Lines with 95 runs within a single day. We consider two single-closure experiments with closures added to two distinct tracks. In the first experiment, we simulate a maintenance of a fairly remote track Bukowa-Kielce with merely 4 runs per day. In the second experiment we close a Berezów-Łączna with 22 closely intertwined runs. For the purpose of this experiment, it is further assumed that there are no requirements on a single-station arrival/departure $\mathbf{V}_{ARR}, \mathbf{V}_{DEP} = \emptyset$ or maximum time extension $\hat{\tau}_{\max} = \tilde{\tau}_{\max} = \infty$, and that stations have a sufficient capacity to receive all the runs $c_i = \infty$. It is however required of all the runs to begin and finish within the whole day $T = 1440$ min., and of all standing times to belong to a fixed period $\tau_{\min,i,p} = 0$, $\tau_{\max} = 30$. The closures in both experiments have the following parameters $\underline{t}_1 = 0$, $\overline{t}_1 = 360$, $T_1 \in \{60, 120, \ldots, 1440\}$. Since for a fixed set of closures and runs, the quality criterion only changes in at most one term, we use the following resulted from (13) quality criterion to evaluate the algorithm

$$\tilde{Q} = \sum_{i \in \mathbf{V}, p \in \mathbf{P}} \left|\Delta\overline{\tau}_{i,p}\right|. \tag{15}$$

Results are gathered in Tables 2 and 3 for Experiment 1 and 2, respectively.

In both experiments for small values of T_1, we obtain results with no changes in the timetable. In fact, since all runs originally enter the closed track significantly past the midnight (522 min for the earliest run in experiment 1, 310 in experiment 2), the whole closure can be performed before the transportation begins. As the duration of the closure increases, it is necessary to change the timetable and, in some cases, to modify the order of the closure-run execution. In either case it becomes apparent that the increasing time shift propagation through the whole network quickly brings the total time shift (as defined in 14) up to values impossible to obtain over just a single track.

It is worth noting that the heuristic properties of the algorithm can lead to situations such as in experiment 2 for which the case with $T_1 = 660$ does not yield a feasible result even though the more constraining case of $T_1 = 720$ does. This is a result of the local optimization procedure for the problem **P3.*i*.1** providing different orderings for both cases.

Table 2 Dependence of $\tilde{Q}$ on T_1 for Bukowa-Kielce closure

T_1	60–480	540	600	660	720	780	840
$\tilde{Q}$	0	52	232	412	673	1247	1978
T_1	900	960	1020	1080	1140	1200	≥1260
$\tilde{Q}$	3336	5025	7350	8497	1083	13,177	∞

Table 3 Dependence of $\tilde{Q}$ on T_1 for Berezów-Łączna closure

T_1	60–300	360	420	480	540
$\tilde{Q}$	0	2092	8544	18,052	32,186
T_1	600	660	720	≥780	–
$\tilde{Q}$	52,786	∞	82,577	∞	–

5 Final Remarks

The complex maintenance planning problem for an operating railway network under a given set of obligatory and facultative track closures has been considered. The problem has been divided into several connected sub-problems due to the complexity of a railway traffic conducted along real-world railway networks. The presented result of the numerical example, performed for the railway network covering the surroundings of Kielce, Poland, has confirmed the usefulness of proposed heuristic solution approach based on the decomposition.

Details of this approach as well as results of more detailed computer simulation experiments of the algorithm are presented in other parallel publications.

This solution method needs the development. In particular, it is very important to consider detours of trains in new timetables as a remedy for closures. Moreover, the elaboration of heuristic solution algorithms, based on metaheuristics, e.g. on tabu search, is planned. Such solution algorithms may ensure shortening the execution times of algorithms.

Acknowledgements The research presented in this paper was partially supported by the European Union within the European Regional Development Fund program Number POIG.01.03.01-02-079/12.

References

1. Lidén, T.: Survey of railway maintenance activities from a planning perspective and literature review concerning the use of mathematical algorithms for solving such planning and scheduling problems. Technical report, Linköpings universitet (2014)
2. Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelentruf, L., Wagenaar, J.: An overview of recovery models and algorithms for real-time railway rescheduling. Transp. Res. Part B Methodol. **63**, 15–37 (2014)
3. Schobel, A.: Capacity constraints in delay management. J. Public Transp. **1**, 135–154 (2009)
4. Brannlund, U., Lindberg, P.O., Nou, A., Nilsson, J.E.: Railway timetabling using Lagrangian relaxation. Transp. Sci. **32**(4), 358–369 (1998) (Informs)
5. Petersen, E., Taylor, A.: Line block prevention in rail line dispatch and simulation models. In: Information Systems and Operations Research, vol. 21(1), pp. 46–51. University of Toronto Press, Toronto (1983)
6. Tornquist, J., Persson, J.: Train traffic deviation handling using Tabu Search and Simulated Annealing. In: Proceedings of 38th Hawaii International Conference on System Sciences, pp. 73a, IEEE (2005)
7. Schobel, A.: A model for the delay management problem based on Mixed Integer Programming. In: Electronic Notes in Theoretical Computer Science, vol. 50, pp. 1–10, Elsevier, Amsterdam (2001)
8. Tornquist, J., Persson, J.: N-tracked railway traffic rescheduling during disturbances. Transp. Res. Part B **41**, 342–362 (2007)
9. Dessouky, M.M., Lu, Q., Zhao, J., Leachman, R.C.: An exact solution procedure to determine the optimal dispatching times for complex rail networks. IIE Trans. **38**(2), 141–152 (2006)

10. Acuna-Agost, R., Michelon, P., Feillet, D., Guaye, S.: Constraint programming and mixed integer linear programming for rescheduling trains under disrupted operations a comparative analysis of models, solution methods, and their integration. In: Lecture Notes in Computer Science 5547/2009, pp. 312–313, Springer, Heidelberg (2009)
11. Ping, L., Axin, N., Limin, J., Fuzhang, W.: Study on intelligent train dispatching. In: IEEE Intelligent Transportation Systems Conference Proceedings, pp. 949–953, Oakland, USA (2001)
12. Acuna-Agost R.: Mathematical modeling and methods for rescheduling trains under disrupted operations. Dissertation, Ph.D. thesis, Universitéd'Avignon et des Pays de Vaucluse (2010)
13. Dekker, R.: Applications of maintenance optimization models: a review and analysis. Reliab. Eng. Syst. Saf. **51**(3), 229–240 (1996)
14. Higgins, A., Ferreira, L., Lake, M.: Scheduling rail track maintenance to minimise overall delays. In: Proceedings of 14th International Symposium on Transportation and Traffic Theory, Jerusalem, Israel (1999)
15. Budai, G., Huisman, D., Dekker, R.: Scheduling preventive railway maintenance activities. J. Oper. Res. Soc. **57**(9), 1035–1044 (2006)
16. Higgins, A.: Scheduling of railway track maintenance activities and crews. J. Oper. Res. Soc. **49**(10), 1026–1033 (1998)
17. Jovanovic, D., Harker, P.T.: Tactical scheduling of rail operations: the SCAN I system. Transp. Sci. **25**(1), 46–64 (1991)
18. Albrecht A.R., Panton D.M., Lee D.H.: Rescheduling rail networks with maintenance disruptions using problem space search. Comput. Oper. Res. **40**(3), 703–712 (2013)
19. Lior, R., Maimon, O.: Clustering methods. In: Data Mining and Knowledge Discovery Handbook, pp. 321–352. Springer, US (2005)

Computer Aided Scheduling and Routing of Vehicle-Carrier Fleet—A Case Study

Andrzej Kozik, Robert Koziarski, Paweł Świątek and Patryk Schauer

Abstract This paper presents a case study of development of an information system for a large enterprise from vehicle-transport industry. A novel optimization model and algorithms developed to solve scheduling and routing of vehicle-carrier fleet are presented and discussed. Carried out qualitative experiments confirmed the usability of developed user-guided search paradigm and high performance of autonomous algorithmic solution search methods.

Keywords Vehicle routing problem · Time windows · Algorithm

1 Introduction

The core of Vehicle Routing Problem is to design the optimal set of routes for fleets of vehicles serving a given set of customers [1]. The problem gained a considerable attention in the literature as it affect both economy and environment. Its variants include time windows, capacity constraints, backhauls and many more; for a reviews see [1–5]. Nevertheless, the problems appearing in industry resembles only parts of problems tackled in the literature, especially they demand to consider special conditions, e.g., EC 561/2006 regulation [6] which drastically changes the model of vehicle driving time.

A. Kozik (✉)
Institute of Mathematics and Informatics, Opole University, Opole, Poland
e-mail: akozik@math.uni.opole.pl

R. Koziarski · P. Świątek · P. Schauer
Institute of Informatics, Wroclaw University of Technology, Wroclaw, Poland
e-mail: robert.koziarski@pwr.edu.pl

P. Świątek
e-mail: pawel.swiatek@pwr.edu.pl

P. Schauer
e-mail: patryk.schauer@pwr.edu.pl

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*,
Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9_20

Moreover, the industrial information systems are not only on algorithms, but also on integration of routing software with other key functions such as billing, inventory tracking, and forecasting [7], integration with accurate geographic information systems and easy-to-use software interfaces.

In this paper we present design and implementation of information system for one of the largest enterprises in the vehicle-transport industry. The main goal for the developed system is to efficiently support human operators in evaluation and verification of feasibility of solutions. Additional features include fast algorithmic improvement of solutions, but rather as an local optimality verification tool than autonomous decision making utility. Since there is no trust in algorithmically generated solutions (as usual), the sophisticated algorithms are developed and included in the system as a test suite and a presentation of capabilities of modern algorithmics.

The developed information system is focused on human-algorithm interaction, and utilizes a Human Guided Search paradigm [8], where algorithm plays a supporting role to the human operator and performs relatively easy tasks as greedy neighborhood exploration or fast search for better solution in the close neighborhood of a given solution. The algorithmic support is viewed rather as a verifier of human choices than a standalone problem solver.

The rest of the paper is organized as follows. The next section precisely describes the optimization model and formulates the problem. Section 3 presents construction details of data structures and algorithms for the considered problem. In Sect. 4 some details of developed information system are presented. The last Section presents results from the industry perspective and concludes the paper.

2 Problem Formulation

In the vehicle transportation industry the Vehicle Routing Problem consist of dispatching vehicle-carriers to execute given set of orders, where each order concerns a transport of given set of vehicles (including passenger cars, light and heavy commercial vehicles) between manufacturer and dealer. On the basis of historical data, durations of pickup and delivery activities, according to location, quantities and types of transported vehicles, can be estimated. Consideration of EC 561/2006 regulation forces timing to be important for the problem. The considered problem belongs to the PDPVRPTW class, i.e., Vehicle Routing Problem with paired pickups and deliveries and Time Windows.

2.1 *Input Data*

Vehicles. There is given a set $V = \{v_1, \ldots, v_n\}$ of n vehicle-carriers (called vehicles for short). Each vehicle $v_i \in V$ is characterized by type, average travel cost per km, average travel speed (given in km/h), initial location and has its associated driver. Each driver is characterized by ready-time and initial state of driving time and rest periods according to EC 561/2006 regulation.

Orders. There is given a set O = {o_1, …, o_m} of m orders. Each order $o_i \in$ O is characterized by:

- pickup and delivery locations,
- logistics operation times of pickup and delivery, defining duration of corresponding activities,
- pickup time-window,
- maximum service time limit, counted from the pickup start-time to the delivery end-time,
- list of feasible vehicle-carrier types, able to realize the order,
- expected route length, distance needed to complete the order in the case of additional delivery points between given pickup and delivery locations,
- revenue.

All orders are considered FTL (full-truck load) and cannot be served simultaneously by one vehicle.

Distances. For each ordered pair of locations appearing in input data (vehicle initial locations and locations of pickups and deliveries), there is given a distance in km. Distances were read from a popular mapping service provider based on geographical coordinates of locations.

2.2 The Problem

A solution to considered problem consist of an assignment of each order to be performed by at most one of vehicles, and a schedule for each vehicle, i.e., pickup and delivery start-times for each assigned order. Orders not assigned to vehicles are assumed to be performed by contractors and did not contribute to the objective function.

Let a profit generated by vehicle be a sum of revenues of its assigned orders reduced by a travel cost of distance needed to perform assigned orders. Let a total profit be a sum of profits generated by vehicles. The problem is to find a solution that maximizes the total profit and satisfies additional constraints. Let a order processing time be a time between start of pickup and end of delivery. A solution is said to be feasible if:

C1. each order is assigned to at most one of vehicles,
C2. each order is assigned to vehicle of suitable type,
C3. processing times of no two orders assigned to the same vehicle overlap (full-truck load condition of each order is satisfied),
C4. for each order its processing time is less than service time limit,

C5. for each order its pickup end-time falls into pickup time-window,
C6. each vehicle can travel a distance between each consecutive locations according to its schedule,
C7. for each vehicle (driver) driving times and rest periods are set according to EC 561/2006 regulation.

3 Algorithms

For the considered Vehicle Routing Problem we constructed data structures and algorithms to be included in developed information system. The need for manual scheduling and routing and adaptation of rules of EC 561/2006 regulation forces to divide a solution into two parts. The first part is a discrete representation of a solution, which can be manipulated either by human user or by algorithms. The second part represents a continuous time variables and is evaluated algorithmically according to given discrete part.

3.1 Discrete Part

A combinatorial representation of a solution is given as a table of $n + 1$ lists of orders. First n lists represents assignment of orders to corresponding vehicles, the last list contains unassigned orders (assigned to contractors). The order of elements on each list determines the order of execution of orders, and thus the route of corresponding vehicle. It is assumed that each order from the set O appears in the representation exactly once, on one of lists—this guarantees that constraint C1 is always satisfied.

3.2 Continuous Part

A discrete representation of a solution is algorithmically evaluated to contain continuous part associated with time variables (a schedule). To this end, each list from a representation is decoded into route data structure, precisely describing a route and a schedule of a corresponding vehicle. The route is described by its start time, end time, corresponding vehicle and a list of stations. Each station has its location, arrival time, departure time and a list of tasks performed on this stop. Each task is either pickup, delivery or rest. Pickup and delivery tasks are described by location, corresponding order, start time and end time; rest tasks are described by type of a rest period (e.g., short break, daily break, weekly break, etc.) and its start time and end time. An example route data structure is presented in Fig. 1. The first

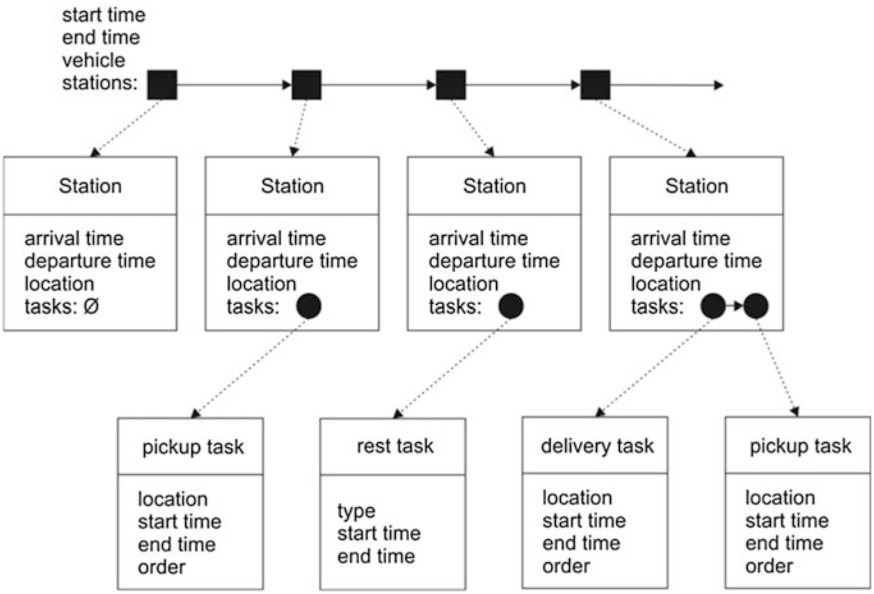

Fig. 1 Example of a route data structure

station is at vehicle initial location and has no tasks; second station has one task—pickup; third station is anonymous and has a rest task; fourth station has two task: delivery followed by a pickup at the same location. Decoding of representation into corresponding route data structures is performed by a simulation procedure.

Simulation procedure. Given a representation, a detailed schedule for each vehicle can be obtained algorithmically by solving optimization problem. Observe, without considering the EC 561/2006 regulation, the corresponding problem is trivial, and all time variables are sequentially derived based on distances, vehicle average speed and given logistics operation times for each order. On the other hand, considering the EC 561/2006 regulation forces the decoding procedure to carefully check all possible combinations of rest periods, their possible shortenings and, foremost, finding a start time for each route, as it can impact the overall duration of a route. Note, that starting a route too early forces a driver to take an unnecessary rest, which in turn can result in delayed pickup or delivery at some later point in the route.

We developed a simulation procedure such that it guarantees that no pickup is early, constraints C3, C6 and C7 are satisfied, minimizes route end time and maximizes route start time in the case of many solutions with the same end time. If the simulation procedure needs to insert a rest task between a two stations, an anonymous station (without localization set) is inserted into the route.

After a representation is decoded into set of routes by a simulation procedure, a solution is evaluated, i.e., feasibility is checked against constraints C2, C4 and C5, and objective function value is computed.

3.3 Operators

Insert move. The solution representation is subject to manipulation performed either by human or by algorithms. As a basic operator we choose an insert move of a given order, that removes the order from the representation and re-insert it into other position. The random insert move selects destination position at random, whereas greedy insert move performs evaluation of the whole neighborhood, evaluating all possible destination positions of a given order and selects the one with the best objective value.

Crossover. The crossover operator we designed constructs a new discrete solution based on two parent solutions, preserving their individual features. The operator bases on the solution construction procedure, in which a solution is constructed from an order list and a solution shape. The shape of solution is a tableaux of $n + 1$ rows, composed of m bins in total. The solution shape is similar to Young tableaux [9], with the exception that rows can be empty. Given the order list, the solution is obtained in *round-robin* fashion, i.e., the solution construction procedure sequentially picks orders from the order list and assigns them to the first empty bin (from the left side) on the *actual row*. Procedure starts from the first row having empty bins, and after placement of each order, changes the actual row to the next row having empty bins. In the first stage of crossover operator, for each parent solution a corresponding order lists and shaper are constructed. Then, shapes of parents are averaged, fractional numbers of bins are randomly rounded (up or down) such that the total number of bins equals m. The order lists are subject to classic one point crossover, to construct an order list of the outcome. The new solution is constructed from averaged shape and outcome order list. The example of crossover procedure is presented in Fig. 2.

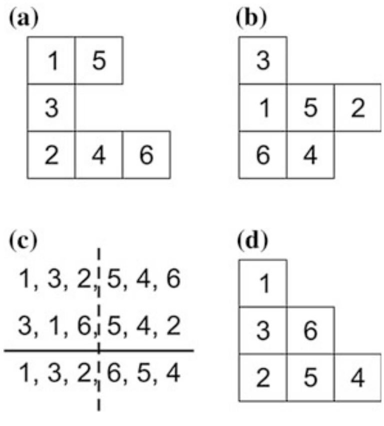

Fig. 2 Example of crossover: **a** and **b** parent solutions and their tableaux, **c** parents order lists and application of one point crossover, **d** outcome solution

3.4 Simulated Annealing Algorithm

As a basic optimization algorithm executed by information system we choose the simulated annealing (SA) [10]. The search process of SA is a semi-random trajectory in the search space, biased towards the "good region" of solution space. In its basic form SA combines an intensification strategy (an iterative local search using a neighborhood concept) with a form of diversification (by so-called cooling-schedule allowing uphill moves to escape from the local minimum). Its efficiency, i.e. the criterion value of delivered solution, strongly depends on the number of visited solutions in the search space. This is especially important in the case when search space is filled with infeasible solutions, where SA performs a lot of futile iterations.

The SA starts the search process from a given initial solution. Then, at each iteration of SA, a solution S′ is obtained from the actual solution S by performing an insert move of a random order. We considered either random insert move or greedy insert move as an operator. Let F(S) be a criterion function to be maximized. The new solution S′ replaces the old solution S either with a probability computed following the Boltzmann distribution $\exp(F(S') - F(S)/\tau)$ if $F(S') < F(S)$, or without a draw if $F(S') \geq F(S)$. The temperature τ is decreased after each iteration iter = 1, 2, …, MaxIter by a geometric cooling schedule, i.e., $\tau iter = \alpha \tau iter - 1$, $\alpha \in (0, 1)$. As the criterion function we considered a total profit of the solution decreased by a static penalty in the case of unfeasible solution. We considered a tardiness time to be a penalty in the case of unfeasible C4 and C5, and 100 for each order assigned to unfeasible vehicle, according to C2 constraint. It was also assumed that a feasible solution always outperforms an unfeasible one.

3.5 Memetic Search

As an advanced solution search metaheuristic we designed a Memetic Search [11] algorithm, which combines the diversification strength of evolutionary methods [12], with intensification provided by local search trajectory method. The pseudo-code of memetic search procedure with Lamarckian learning is given below:

```
P ← GenerateInitialPopulation()
Evaluate(P)
while not satisfied StopCondition do
      P′ ← Recombination(P)
      P′′ ← Mutation(P′)
      for each solution s ∈ P′′:
           s′ = LocalSearch(s)
           replace score of s by score of s′ in P′′
      P ← Selection(P, P′′)
end while
```

The initial population is constructed by applying solution construction procedure described in Sect. 3.3 to randomly generated shapes and order lists. The stop condition was set to maximal number of iterations. The recombination procedure constructs an offspring population using crossover operator applied to randomly sampled parent solutions. Mutation is performed with low probability as a random insert move of random order. As a local search procedure we applied simulated annealing algorithm. The selection is based on roulette method [10, 12].

3.6 Implementation

The considered problem and corresponding algorithms were implemented in the Algorithm Composition Engine presented in [13]. The Pill object was constructed such that that the discrete part was implemented as a Representation, continuous part as a Solution, simulation procedure as a translator and insert moves and crossover as operators on representation. Constraints C2, C4, C5 and objective function were implemented as separate entities and included in the Pill object. Simulated annealing and Memetic Search algorithms were implemented in the engine to explore the possibility to construct hybrid algorithms [14] from custom and universal metaheuristics.

4 Information System

Algorithms developed in Sect. 3 were incorporated into information system built for vehicle-transport enterprise. The main goal was to enable a manual decision making by logistics with algorithmic methods only as a supplementary helpers functions. To this end we developed a graphical user interface with drag and drop functionality, allowing to manually assign orders to vehicles and to determine order of their execution. An example view on the interface is presented in Fig. 3, showing an assignment of 7 orders to three vehicles: Vehicle 1 performs Order 1 followed by Order 0, Vehicle 2 performs Orders 3 and 4, and Vehicle 3 performs Orders 5, 6, and 7. Each order is represented as a square containing its id (name) in the first line, pickup window (as on-screen information essential for human operators), pickup location on the left and delivery location on the right (in the third line). To enhance user experience and to give a visual feedback, each order square contains parameters of evaluated solution, i.e., pickup start-time (left) and delivery end-time (right). In the case of unfeasible solution, each order that contributes to unfeasibility has a red border and the amount of tardiness is given with the red font. As an example, the Order 0 in Fig. 3 has a tardy pickup by 18 min. Each vehicle in Fig. 3 is described by its id, initial location (in the second line) and driver availability time (in the third line). Each order can be moved to any other position in vehicle orders lists.

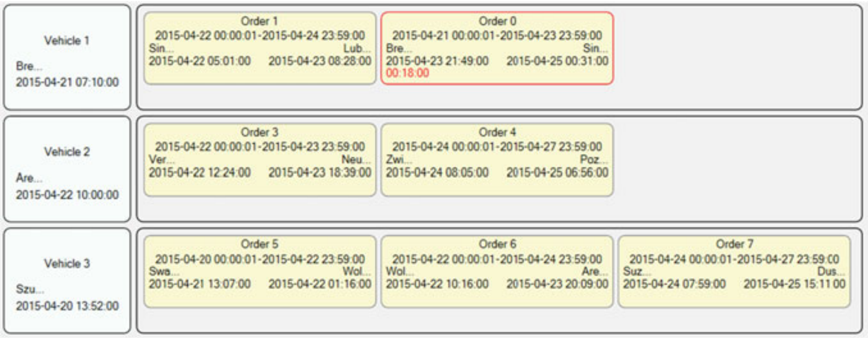

Fig. 3 View of the user interface of the manual scheduler

The algorithmic support of the manual scheduling was primarily a feedback on criterion value and feasibility status of the current solution (reevaluated after each manual move of an order), but it also included a suite of supporting algorithms:

- greedy rescheduling of manually chosen order, performing an insert neighborhood search of the order picked by the user,
- greedy rescheduling of the best order, performing an insert neighborhood search of all orders and selects the one with better improvement of the objective function,
- fast SA, starting with low initial temperature ($\tau = 100$) and small number of iterations ($MaxIter = 100$) and performing a greedy moves of random orders,
- detailed SA, starting with high initial temperature ($\tau = 10{,}000$) and big number of iterations ($MaxIter = 50{,}000$) and performing a random moves of random orders.
- Memetic Search with population sixe set to 50,500 iterations, and internal SA with $\tau = 500$ and $MaxIter = 1000$.

The intent of first two rescheduling strategies was to efficiently help with a manual design of solution, the third strategy is used for checking if the current solution can be quickly improved, and the last two for the automatic solving of the considered scheduling problem.

The second aspect of developed information system was reporting capabilities about details of solutions. The report presents information on three levels: solution level, vehicle level and order level. The order level presents revenue, cost and profit generated by each order, together with its important characteristics: distance and empty distance (measured as a distance from actual location to pickup location), responsible for cost generation. The vehicle level gathers characteristics of orders assigned to particular vehicle, and solution level presents the overall measures. Additionally, for each vehicle, a time-line graph, presenting driver activities (pickups, deliveries, rest periods) are presented. The report was implemented using Timeline Web Widget [15].

While solution measures are important decision-making tools for human-operators, helping them to understand where the costs are generated, the detailed time-line driver activity charts was a mandatory tools, enabling manual verification of schedule feasibility.

5 Results

The performance of developed information system has been subject to qualitative evaluation by logisticians worked for the enterprise. The system has been filled by orders collected by a one-week period and confronted with the actual solution used by the enterprise as an reference.

The presented information system proved to be important tool in solving vehicle-carrier routing and scheduling problem, especially in automation of the process of evaluation of objective value and verification of feasibility of solutions. One of the most important features (as expressed by human-operators) was the ability to deliver a proof of unfeasibility of evaluated solutions in the form of visual feedback during manual scheduling—it helped human-operators to resolve impossible assignments. The computed solution measures also helped to understand the source of costs and optimize the assignments. Moreover, implemented user-guided search paradigm enabled users to quickly obtain a high-quality solutions, by using algorithmic support methods like simple greedy moves of hand-pointed orders. It is worth to note, that such obtained solutions was better than provided referential ones.

The developed detailed SA and Memetic Search algorithms was able to find solution of the same quality as ones obtained by human users (with algorithmic help), what is considered a proof of efficiency of autonomous methods of solving routing and scheduling problems, and results in gaining industry trust to solutions generated autonomously by algorithms. Therefore, our future research direction is design, implementation and quantitative evaluation of hyper-heuristic methods [16], able to autonomously tune their behavior to solved instances.

Acknowledgements The research presented in this paper was partially supported by the Polish Ministry of Science and Higher Education and the European Union within the European Regional Development Fund, Grant No. POIG.01.03.01-02-079/12.

References

1. Toth, P., Vigo, D.: An overview of vehicle routing problems. In: SIAM Monographs on Discrete Mathematics and Applications, pp. 1–26 (2002)
2. Gendreau, M., Laporte, G., Potvin, J.-Y.: Metaheuristics for the VRP. In: Toth, P., Vigo, D. (eds.) The Vehicle Routing Problems. SIAM Monographs on Discrete Mathematics and Applications, pp. 129–154 (2002)

3. Parragh, S., Doerner, K., Hartl, R.: A survey on pickup and delivery problems. Journal für Betriebswirtschaft **58**(1), 21–51 (2008)
4. Parragh, S., Doerner, K., Hartl, R.: A survey on pickup and delivery problems. Journal für Betriebswirtschaft **58**(2), 81–117 (2008)
5. Suresh, N.K., Ramasamy, P.: A survey on the vehicle routing problem and its variants. Intell. Inf. Manage. **4**, 66–74 (2012)
6. EC 561/2006 regulation. http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32006R0561
7. Partyka, J., Hall., R.: On the road to connectivity. OR/MS today **37**(1), 41–49 (2010)
8. Klau, G., Lesh, N., Marks, J., Mitzenmacher, M.: Human-guided search. J. Heuristics **16**(3), 289–310 (2010)
9. Fulton, W.: Young Tableaux with Applications to Representation Theory and Geometry. Cambridge University Press, New York (1997)
10. Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput. Surv. **35**(3), 268–308 (2003)
11. Moscato, P., Cotta, C.: A Gentle introduction to memetic algorithms. In: Glover, F., Kochenberger, G.A. (eds.) Handbook of Metaheuristics, International Series in Operations Research and Management Science, vol. 57. Springer, Heidelberg (2003)
12. Michalewicz, Z., Schoenauer, M.: Evolutionary algorithms for constrained parameter optimization problems. Evol. Comput. **4**(1), 1–32 (1996)
13. Kozik, A., Drwal, M., Gonczarek, A.: Algorithms composition engine for domain-specific information systems. In: Świątek, J., Borzemski, L., Grzech, A., Wilimowska, Z. (eds.) Information Systems Architecture and Technology: Systems Analysis Approach to the Design, Control and Decision Support, pp. 43–52. Oficyna Wydawnicza Politechniki Wrocławskiej, Wrocław (2014)
14. Blum, C., Aguilera, M., Roli, A., Sampels, M.: Hybrid metaheuristics—an emerging approach to optimization. Stud. Comput. Intell. **114** (2008)
15. Timeline Web Widget for Visualizing Temporal Data. http://www.simile-widgets.org/timeline/
16. Burke, E., Gendreau, M., Hyde, M., Kendall, G., Ochoa, G., Ozcan, E., Qu, R.: Hyper-heuristics: a survey of the state of the art. J. Oper. Res. Soc. **64**, 1695–1724 (2013)

Author Index

L. Borzemski et al. (eds.), *Information Systems Architecture and Technology: Proceedings of 36th International Conference on Information Systems Architecture and Technology – ISAT 2015 – Part I*, Advances in Intelligent Systems and Computing 429, DOI 10.1007/978-3-319-28555-9