

Mechanisms and Machine Science

Stanisław Zawiślak
Jacek Rysiński *Editors*


Graph-Based Modelling in Science, Technology and Art

 Springer

Mechanisms and Machine Science

Volume 107

Series Editor

Marco Ceccarelli , Department of Industrial Engineering, University of Rome Tor Vergata, Roma, Italy

Advisory Editors

Sunil K. Agrawal, Department of Mechanical Engineering, Columbia University, New York, USA

Burkhard Corves, RWTH Aachen University, Aachen, Germany

Victor Glazunov, Mechanical Engineering Research Institute, Moscow, Russia

Alfonso Hernández, University of the Basque Country, Bilbao, Spain

Tian Huang, Tianjin University, Tianjin, China

Juan Carlos Jauregui Correa, Universidad Autonoma de Queretaro, Queretaro, Mexico

Yukio Takeda, Tokyo Institute of Technology, Tokyo, Japan

This book series establishes a well-defined forum for monographs, edited Books, and proceedings on mechanical engineering with particular emphasis on MMS (Mechanism and Machine Science). The final goal is the publication of research that shows the development of mechanical engineering and particularly MMS in all technical aspects, even in very recent assessments. Published works share an approach by which technical details and formulation are discussed, and discuss modern formalisms with the aim to circulate research and technical achievements for use in professional, research, academic, and teaching activities.

This technical approach is an essential characteristic of the series. By discussing technical details and formulations in terms of modern formalisms, the possibility is created not only to show technical developments but also to explain achievements for technical teaching and research activity today and for the future.

The book series is intended to collect technical views on developments of the broad field of MMS in a unique frame that can be seen in its totality as an Encyclopaedia of MMS but with the additional purpose of archiving and teaching MMS achievements. Therefore, the book series will be of use not only for researchers and teachers in Mechanical Engineering but also for professionals and students for their formation and future work.

The series is promoted under the auspices of International Federation for the Promotion of Mechanism and Machine Science (IFTOMM).

Prospective authors and editors can contact Mr. Pierpaolo Riva (publishing editor, Springer) at: pierpaolo.riva@springer.com

Indexed by SCOPUS and Google Scholar.

More information about this series at <http://www.springer.com/series/8779>

Stanisław Zawiślak · Jacek Rysiński
Editors

Graph-Based Modelling in Science, Technology and Art

 Springer

Editors

Stanisław Zawiślak
Faculty of Mechanical Engineering
and Computer Science
University of Bielsko-Biala
Bielsko-Biała, Poland

Jacek Rysiński
Department of Fundamentals of Machine
Building
University of Bielsko-Biala
Bielsko-Biała, Poland

ISSN 2211-0984

Mechanisms and Machine Science

ISBN 978-3-030-76786-0

<https://doi.org/10.1007/978-3-030-76787-7>

ISSN 2211-0992 (electronic)

ISBN 978-3-030-76787-7 (eBook)

© Springer Nature Switzerland AG 2022

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

Graph-based modeling is a well-known technique in many branches of knowledge allowing analysis and synthesis of systems. The present book gives a glimpse of its application to some scientific problems via a series of original chapters.

The book entitled *Graph-Based Modeling in Engineering* published by Springer in 2017, prepared by the present editors, had reached a great success taking into account number of downloaded chapters (top 25%). Therefore, the idea of continuation arose and it is therethrough converted into reality.

Graph theory is an area of discrete mathematics. It is commonly believed that the paper of Leonhard Euler from 1736 on Königsberg Bridges originated scientific investigations in this field. Arthur Cayley and Gustav Kirchhoff are commonly considered as other founding fathers of graph theory. So, taking into consideration the fields from which graphs were originated, we can state that there were logistics, chemistry and electrical engineering. Due to mechanical–electrical analogies, the proposals were issued on a possibility of an analysis of mechanical systems by means of this “tool.” In worldwide known journals such as *Mechanism and Machine Theory* (under IFToMM patronage) and *JME ASME*, many papers dedicated to this area have been published in recent 10 years. The presented book just confirms the further possibilities of utilization of graphs in these fields as well as in some new ones, especially for objects and tasks such as mechanical systems, planetary gears and design activities.

Graph $G(V, E)$ is a pair of the sets, V —nonempty, finite set of vertices and E —finite set of edges. One can ask: What is the mystery in usage of graphs, especially to solve mechanical problems, but also in so many different areas? Till now, such a question has been frequently asked during versatile conferences and/or congresses. The immanent phase of any modeling is simplification, and one way of rephrasing the problem is expressing it in a discrete paraphrase instead of the initial, frequently continuous, version. Naturally, discretization is a basic step in such tasks and/or methodologies like, e.g., numerical integration, finite element method, boundary element method, solving differential equations as well as reverse engineering clouds of points. Therefore, the essential background for graph usage is discretization of the considered system, too. Next steps consist in cross-domain knowledge transfer, i.e., reformulation of mechanical, chemical, electrical, etc., relationships into their graph

theory counterparts. It is a clue of the methodology as well as the most intriguing and essential phase. One can be astonished how different scientific notions are represented via graph-related objects—for example, cycles, cuts, trees, etc. Then, the considered problem is solved via graph-based methodology. Results are converted back into the adequate origin field of knowledge. We are solving problems in such a way because graph modeling gives encoding algebraic structures which can be stored, considered and processed in computer programs. Moreover, graph theory evolved into some new subbranches, for example, algorithmic graph theory, enumerative graph theory and graph drawing methodologies, which allow for visualization, algorithmic calculations and performing tasks by means of reliable and commercial software, e.g., in case of bond graphs. In many cases, the graph algorithms are theoretically proved to be correct, convergent and effective.

The aim of the book is to show some graph application in the above described way. The chapters collected in the book are related to different fields of knowledge, according to its title.

The book is divided into three parts. The first one is dedicated to some mechanical problems related to gears, planetary gears and engineering installations. Two problems related to scheduling and delivery routes are discussed, too. New, original case studies were discussed, applying, e.g., immune algorithms to well-known routing problem. Utilized graphs are: bond, directed, simple and weighted graphs. Moreover, so-called contour graphs were used for modeling of planetary gears what is really rare, like it arises from the references' analysis. However, it is worth for propagating in the opinion of the authors and the editors.

The second part of the book encloses some graph-based methods applied in medical analyses as well as biological and chemical modelling. It confirms that the original areas of graph application are still in development as well as shows new areas of application, especially via utilization of so-called Petri networks.

The third part is related to various topics, e.g., drama analysis, aiding of design activities and network visualization. All these applications show how wide could be the area of graphs' utilization and how discretization activity is fruitful in these approaches.

The last chapter is dedicated to the outstanding scientist who is a founding father of the direction of investigation consisting in graph-based modeling in mechanics in Poland, which he started just 50 years ago. Usually, birthday or anniversary of academic careers is celebrated in this way. Here, via achievements of the outstanding Polish Professor, the 50th anniversary of the direction of investigations is reminded. Simultaneously, the Hero of the chapter, i.e., Prof. Józef Wojnarowski in an active, long-term IFToMM activist. This organization gives a patronage to the present series of books. The wide reference lists give additionally an insight into range of modeling of mechanical systems by means of graphs and fruitful results of this approach.

In one of the chapters, once more, the problem of Königsberg bridges is mentioned. Historical investigations to the problem are still in progress, and they have been recently, several times, related in papers as well as in books. One of the most interesting publications on this topic is a paper of Roman Sznajder entitled “On known

and less known relations of Leonhard Euler with Poland” (*Studia Historiae Scientiarum* 15, 75–110 (2016)). The most shocking information is that Euler had never ever visited town Königsberg but simultaneously it is not completely clear how he received the data to the problem. The paper suggests that the problem was delivered and announced to him by Gdańsk (named also Danzig) mathematicians: Carl G. Ehler and Heinrich Kuehn. The suggestion is confirmed by original letters spotted in libraries, e.g., in Tartu (Estonia). In fact, the solution to the problem was presented at the University of St. Petersburg by Euler in 1735. However, the publication—dated formally for a year 1736—was printed a few years later.

The graph-based modeling is a useful, powerful, handy and multipurpose scientific “tool.” There are only a few books on worldwide market related to real, practical graph application in engineering where particular usages are described in detail—e.g., written by N. Deo (many years ago), L.-W. Tsai – related to mechanisms and A. Kaveh—related to civil engineering. The present book shows a wide spectrum of application where original methods, algorithms and/or graph types are utilized. It shows also that there are new areas of possible usage, e.g., drama analyses what additionally confirms graph theory as beautiful and all-embracing branch of science. In fact, networks are weighted graphs where sometimes geographical position of nodes is additionally taken into account. The position can be stable or in motion. Network theory is a natural generalization of graph theory which is also important nowadays.

So, via the contents of the present book, one can be more familiar with the possibilities of graph-based modelling in versatile areas of knowledge.

Bielsko-Biała, Poland
April 2021

Stanisław Zawiślak
Jacek Rysiński

Contents

Part I Graph-Based Modelling in Technology Especially in Mechanical Engineering and Logistics

1	Application of Bond Graphs in Modelling of the Energy Harvesting Systems from Vibrating Mechanical Devices	3
	Jerzy Margielewicz, Damian Gąska, Grzegorz Litak, and Tomasz Haniszewski	
2	Bond Graph Based Synthesis of Generic Power Split Modelling for Epicyclic Four-Speed Gears	19
	G.-H. Geitner and G. Komurgoz	
3	Chosen Aspects of Analysis and Synthesis of Coupled and Complex Planetary Gears via Search and Contour Graphs Modelling	45
	A. Deptuła, J. Drewniak, J. Kopeć, and S. Zawiślak	
4	Modeling Dynamics of Cored Wire in Molten Steel Using Linear Graph Representation	71
	Kyrylo S. Krasnikov	
5	Methodology of Solving Selected Routing Problems	85
	Bogna Mrówczyńska	
6	Using Graphs for Modeling and Solving Cyclic Flow Shop with Waiting Time Constraints	105
	Czesław Smutnicki and Wojciech Bożejko	

Part II Graph-Based Modelling in Science Especially in Medicine and Chemistry

7	Using Graphs in Processing of Light Microscope Medical Images	127
	M. Ždímalová, A. Chatterjee, M. Kopáni, and H. Svobodová	

8	On the Development of Directed Acyclic Graphs in Differential Diagnostics of Pulmonary Diseases with the Help of Arterial Oscillogram Assessment	157
	V. P. Martsenyuk, D. V. Vakulenko, L. A. Hryshchuk, L. O. Vakulenko, N. O. Kravets, and N. Ya. Klymuk	
9	Bipartite Graphs—Petri Nets in Biology Modeling	175
	Anna Gogolińska and Wiesław Nowak	
10	Labeled Graphs in Life Sciences—Two Important Applications	201
	Piotr Formanowicz, Marta Kasprzak, and Piotr Wawrzyniak	
Part III Graph-Based Modelling in Art, Design and Network Modeling		
11	Graph-Based Analysis of Drama Text—Case Study Based on Shakespeare’s ‘Measure for Measure’	221
	V. Marцениuk, S. Zawiślak, and J. Kopec	
12	A Survey of Different Graph Structures Used in Modeling Design, Engineering and Computer Science Problems	243
	Barbara Strug, Grażyna Ślusarczyk, Anna Paszyńska, and Wojciech Palacz	
13	On the Problem of Visualization of Big Graphs for Infrastructure Engineering	277
	V. Martsenyuk, M. Karpinski, S. Zawiślak, A. Vlasyuk, A. Shaikhanova, and O. Martsenyuk	
Part IV Miscelanous		
14	Professor Józef Wojnarowski—IFToMMist and Pioneer of Graphs’ Application in Mechanics in Poland	291
	S. Zawiślak and J. Kopec	

Part I
Graph-Based Modelling in Technology
Especially in Mechanical Engineering
and Logistics

Chapter 1

Application of Bond Graphs in Modelling of the Energy Harvesting Systems from Vibrating Mechanical Devices



Jerzy Margielewicz, Damian Gąska, Grzegorz Litak,
and Tomasz Haniszewski

Abstract This chapter presents the results of computer simulations of the system of energy harvesting from vibrating mechanical devices. Detailed model tests were carried out for the system of cantilever beams coupled with elastic elements. The end of one vibrating element is loaded with a permanent magnet, which defines a non-linear potential characteristic with the fixed permanent magnets mounted in the housing. Based on the formulated linear mathematical model, the surfaces representing the sensitivity of the system were drawn with regard to its selected parameters. However, in relation to the non-linear system, the results of model tests were presented in the form of an indicator characterizing the efficiency of energy generation. The RMS value of the voltage induced on the piezoelectric electrodes was adopted as an indicator. The method of bond graphs was used to derive mathematical models. The system of equations achieved in this systematic way is elegant and clearly formed, which also induces its correctness. The obtained results of computer simulations indicate that the effective energy harvesting in a non-linear system is almost double that of a linear one.

Keywords Four degree of freedom system · Sensitivity analysis · Efficiency · Non-linear system

J. Margielewicz (✉) · D. Gąska · T. Haniszewski
Faculty of Transport and Aviation Engineering, Silesian University of Technology, Krasinskiego
8, 40-019 Katowice, Poland
e-mail: jerzy.margielewicz@polsl.pl

D. Gąska
e-mail: damian.gaska@polsl.pl

T. Haniszewski
e-mail: tomasz.haniszewski@polsl.pl

G. Litak
Faculty of Mechanical Engineering, Lublin University of Technology, Nadbystrzycka 36, 20-618
Lublin, Poland
e-mail: g.litak@pollub.pl

1.1 Introduction

The object of model research contained in this work is the system of harvesting energy from vibrating mechanical devices. From a technical point of view, energy recovery from the human environment is possible in many ways. Most often it is obtained from the so-called renewable sources as a result of transformation of solar, wind or water energy. However, these sources are not always available, and therefore an alternative approach is to use simple electromechanical systems that transform the energy of mechanical vibrations into electricity [1], via piezoelectric transducers [2, 3], electrostatics [4], or electromagnetic [5]. In our considerations, we used piezoelectric transducers in model research. We are aware that such solutions are characterized by low energy efficiency. Nevertheless, through them it is possible to supply simple electronic devices such as sensors monitoring the technical condition of devices or wireless data transmission systems. In addition, the piezoelectric effect is used in: seismic systems, fuses, microphones, micro drives and elements of active vibration reduction systems.

The piezoelectric effect was first discovered in 1880 by brothers Piere and Jaques Curie, who showed the presence of an electric charge on the surface of a loaded mechanical quartz crystal. However, this phenomenon only takes place in certain solids, characterized by an ordered atomic structure. This phenomenon was also dealt with by Lippman, who in 1881 based on thermodynamic considerations, showed the possibility of the inverse piezoelectric phenomenon. As the name suggests, this phenomenon is characterized by the ability to deform material due to the electric field applied to it. The existence of this phenomenon in laboratory conditions was experimentally demonstrated by the Curie brothers. The term piezoelectricity was proposed in 1881 by Wilhelm Gottlieb Hankel. The first technical application of the piezoelectric effect was in 1917, when Langevin used a quartz crystal to detect submarines.

In the initial period, the basic piezoelectric materials were: quartz, tourmaline, and Rochella salt. They showed a negligible ability to transform mechanical energy into electrical energy. The need to use much more perfect materials has led to the development of technology and the production of ceramic materials such as barium titanate (BaTiO_3), lead titanate (PbTiO_3), lead zirconate (PZT), lead and magnesium niobate. They are most often produced in the form of polycrystals, less often from materials of piezoelectric fibers (MFC), which are characterized by a low value of generated current. Currently, piezoelectric polymers such as (PVDF) and its copolymers are increasingly being replaced in technical applications with ceramic materials. This is mainly due to the fast and cheap technological process, their high mechanical strength and a wide range of transmitted dynamic forces. In addition, these materials are characterized by high susceptibility and a short response to a change in stress.

The issue of harvesting energy from vibrating mechanical devices has been devoted to a lot of work on identifying the amplitude-frequency characteristics of piezoelectric transducers, experimental and model research. In general, the formal basis for conducting computer simulations dedicated to energy harvesting is the

adoption of an equivalent circuit mapping the electrical properties of piezoelectrics. At the same time, such a circuit is most often mapped through a series connection of a capacitor and resistor. We only signal that, in relation to electromagnetic transducers, the replacement electrical circuit consists of a coil connected in series with a resistor. At this point it is also worth mentioning that in electromagnetic transducers the resistance in the substitute electric circuit is negligible and during computer simulations it can be accurately represented by the resistance resulting from the current flowing through the wire. Bearing in mind the simplification of analyses, energy harvesting systems are most often loaded with a resistive element, connected in parallel in the transducer substitute circuit.

From an engineering point of view, the construction of most systems for harvesting energy from vibrating mechanical systems, which is based on a flexible beam, on which piezoelectric transducers are glued. This type of system is described by means of linear differential equations. Due to the fact that during operation of machines, a broad spectrum of harmonics is excited, solutions are used that take into account many transducers connected in parallel, each tuned to a different frequency [6]. The need to tune the transducers to the appropriate harmonics caused that the attention of researchers focused on nonlinear systems in which it is possible to excite harmonics in a wide frequency spectrum. Design solutions of non-linear systems, like linear systems, are based on a flexible beam, and additionally contain additional elements in the form of permanent magnets [7, 8]. Permanent magnets are used to properly shape the energy potential barrier. The potential modelling function is most often mapped through a polynomial function [9]. There are also solutions with two flexible beams coupled to electrical circuits [10] and mechanically via a spring [11], or inertial elements [12]. There are also designs of energy systems based on free and coupled single pendulums [13, 14] and double pendulums [15].

Bearing in mind quantitative and qualitative research, the differential equations of motion were derived using the bond graph method.

1.2 Energy Harvesting in a Linear System

The subject of model research is a system with four degrees of freedom, harvesting energy from vibrating mechanical devices. The considered system consists of four flexible beams *II* and *III*. At the same time, to the beam *II*, on which the piezoelectric transducer is glued, additional beams *III* are attached, at the ends of which load (mass) $m_2 = m_3 = m_4$ elements *V* are placed. During computer simulations different lengths of beams *III* and the same values of load elements were assumed. The whole system is attached to the vibrating device via a non-deformable frame *IV*.

Based on the schematic diagram of the vibrating mechanical subsystem (Fig. 1.1), a bond graph was built. Bearing in mind the possibility of harvesting energy, it is necessary to map the physical properties of the piezoelectric transducer. In general, the external load acting on the piezoelectric causes its deformation, as a result of which positive and negative ions in the crystal lattice are mutually displaced. As a

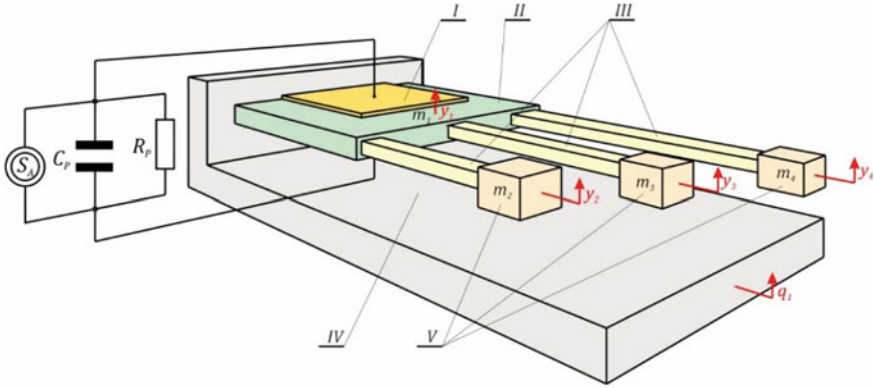


Fig. 1.1 Schematic diagram of the examined energy harvesting system

consequence, an electric charge is induced on its electrodes. An equivalent electrical circuit that mimics the real properties of piezoelectric, in addition to the parallel capacitance C_P and internal resistance R_P can also include parasitic inductance. In model tests, inductance is most often neglected due to its negligible low value. The coupling between the mechanical subsystem and the electrical circuit was mapped using the current source S_A , which in the bond graph was mapped via a transformer (Fig. 1.2).

Bearing in mind the efficient conduct of quantitative and qualitative numerical experiments, differential motion equations in explicit form were derived, which were further transformed into dimensionless form. A built graph of bonds, consists of 28 edges and 9 1-junction nodes, but only four of them were assigned generalized coordinates. There is a causal conflict at some 1-junction nodes. However, the introduction of additional edges representing fictitious sources of excitation (mapped in a blue dashed line) results in an unambiguous assignment of *flow* type coordinates. This procedure is most often carried out in relation to 1-junction nodes that are incidental with inertial elements.

When modelling systems of different nature, in particular electromechanical systems by the method of bond graphs, the analogue of current is speed. However, the voltage corresponds to the force or moment of force. We use this analogy in our research. The consequence of adopting such a model assembly causes that the motion equations corresponding to the mechanical subsystem are generated on the basis of 1-junction nodes. However, the corresponding equations characterizing the electric circuit are derived from the incident 1-junction node with C_P capacity and piezoelectric R_P resistance. As in the case of single nodes, this node has a causal conflict. It is eliminated by introducing an additional fictitious edge representing the source of force *effort* (the edge marked with a dashed line in red), which clearly defines the generalized coordinate characterizing the functioning of the electrical circuit.

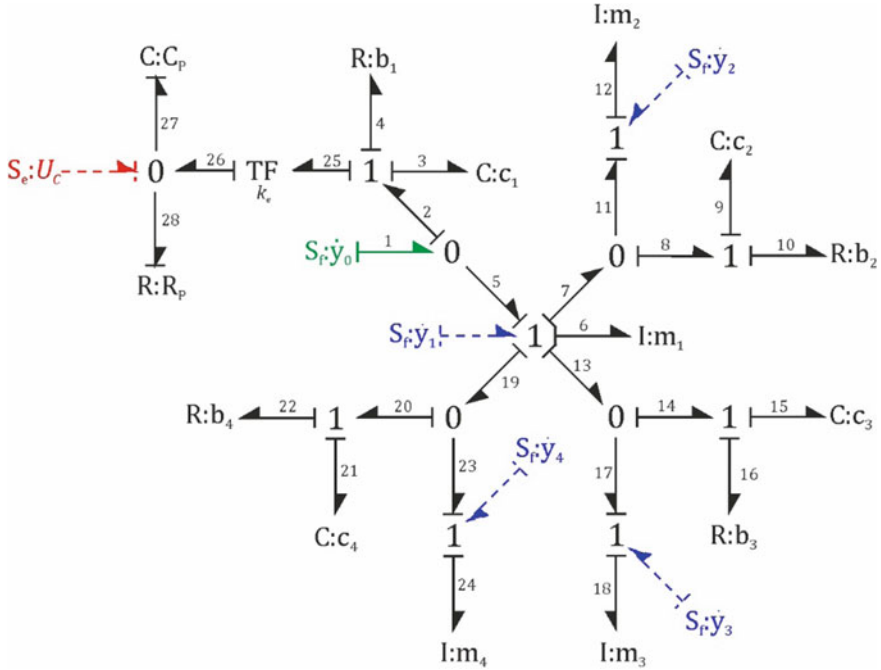


Fig. 1.2 Schematic diagram of the examined energy harvesting system in form of bond graph

In general terms, the essence of deriving the differential equations of motion comes down to recording the cause-and-effect relationships occurring in the elements and nodes of the bond graph. With respect to 1-junction nodes representing the generalized coordinates of the mechanical subsystem, the corresponding relationships take the form:

$$\begin{cases} e_6 + e_7 + e_{13} + e_{19} - e_5 = 0, \\ e_{12} - e_{11} = 0, \\ e_{18} - e_{17} = 0, \\ e_{24} - e_{23} = 0, \end{cases} \quad \begin{cases} f_6 = f_7 = f_{13} = f_{19} = f_5 = \dot{y}_1, \\ f_{12} = f_{11} = \dot{y}_2, \\ f_{18} = f_{17} = \dot{y}_3, \\ f_{24} = f_{23} = \dot{y}_4. \end{cases} \quad (1.1)$$

We remind you that in 1-junction nodes, the stress variables e_i (effort) of the edges of the bond graph are added together. In contrast, the flow variables f_i (flow) of incidental edges with a given 1-junction node assume the same values. The procedure is similar in the case of 0-junction nodes, at this stage of deriving motion equations it is possible to take into account relationships corresponding to flow variables (1.1):

$$\begin{cases} f_2 = f_1 - f_5 = \dot{q}_1 - \dot{y}_1, \\ f_8 = f_7 - f_{11} = \dot{y}_1 - \dot{y}_2, \\ f_{14} = f_{13} - f_{17} = \dot{y}_1 - \dot{y}_3, \\ f_{20} = f_{19} - f_{23} = \dot{y}_1 - \dot{y}_4, \end{cases} \quad \begin{cases} e_2 = e_1 = e_5, \\ e_8 = e_7 = e_{11}, \\ e_{14} = e_{13} = e_{17}, \\ e_{20} = e_{19} = e_{23}. \end{cases} \quad (1.2)$$

In contrast to 1-junction nodes, in the 0-junction nodes of the bond graph, the *effort* variables e_i take the same values, while the *flow* variables f_i are added together. The dependencies describing the relationships in the remaining 1-junction nodes are given below as equations:

$$\begin{cases} e_3 + e_4 + e_{25} = e_2, \\ e_9 + e_{10} = e_8, \\ e_{15} + e_{16} = e_{14}, \\ e_{21} + e_{22} = e_{20}, \end{cases} \quad \begin{cases} f_3 = f_4 = f_{25} = f_2, \\ f_9 = f_{10} = f_8, \\ f_{15} = f_{16} = f_{14}, \\ f_{21} = f_{22} = f_{20}. \end{cases} \quad (1.3)$$

Cause-effect relationships corresponding to inertial, capacitive and dissipative elements are given by equations:

$$\begin{aligned} e_6 &= m_1 \frac{df_6}{dt} = m_1 \frac{d\dot{y}_1}{dt}, & e_3 &= c_1 \int f_3 dt, & e_4 &= b_1 f_4, \\ e_{12} &= m_2 \frac{df_{12}}{dt} = m_2 \frac{d\dot{y}_2}{dt}, & e_9 &= c_2 \int f_9 dt, & e_{10} &= b_2 f_{10}, \\ e_{18} &= m_3 \frac{df_{18}}{dt} = m_3 \frac{d\dot{y}_3}{dt}, & e_{15} &= c_3 \int f_{15} dt, & e_{16} &= b_3 f_{16}, \\ e_{24} &= m_4 \frac{df_{18}}{dt} = m_4 \frac{d\dot{y}_4}{dt}, & e_{21} &= c_4 \int f_{21} dt, & e_{22} &= b_4 f_{22}, \\ & & f_{27} &= C_P \frac{de_{27}}{dt}, & f_{28} &= \frac{1}{R_P} e_{28}. \end{aligned} \quad (1.4)$$

In relation to the element converting mechanical energy into electrical energy, the appropriate cause-effect relationships, taking into account (1.2) and (1.3) take the form:

$$f_{26} = k_e f_{25} = k_e (\dot{q}_1 - \dot{y}_1), \quad e_{25} = k_e e_{26} = k_e U_C. \quad (1.5)$$

As a result of substitution (1.4) and (1.5)–(1.3), relations are obtained representing the mechanical couplings occurring between individual beams of the energy harvesting system:

$$\begin{cases} e_2 = b_1 f_4 + c_1 \int f_3 dt + e_{25} = b_1 (\dot{q}_1 - \dot{y}_1) + c_1 \int (\dot{q}_1 - \dot{y}_1) dt - k_e U_C, \\ e_8 = b_2 f_{10} + c_2 \int f_9 dt = b_2 (\dot{y}_1 - \dot{y}_2) + c_2 \int (\dot{y}_1 - \dot{y}_2) dt, \\ e_{14} = b_3 f_{16} + c_3 \int f_{15} dt = b_3 (\dot{y}_1 - \dot{y}_3) + c_3 \int (\dot{y}_1 - \dot{y}_3) dt, \\ e_{20} = b_4 f_{22} + c_4 \int f_{21} dt = b_4 (\dot{y}_1 - \dot{y}_4) + c_4 \int (\dot{y}_1 - \dot{y}_4) dt. \end{cases} \quad (1.6)$$

The structure of differential equations of mechanical subsystem movement is achieved as a result of substitution of equations (1.4), (1.5) and (1.6) to (1.1):

$$\left\{ \begin{array}{l} m_1 \frac{d\dot{y}_1}{dt} + b_2(\dot{y}_1 - \dot{y}_2) + c_2 \int (\dot{y}_1 - \dot{y}_2)dt + b_3(\dot{y}_1 - \dot{y}_3) + c_3 \int (\dot{y}_1 - \dot{y}_3)dt \\ \quad + b_4(\dot{y}_1 - \dot{y}_4) + c_4 \int (\dot{y}_1 - \dot{y}_4)dt - k_e U_C - b_1(\dot{q}_1 - \dot{y}_1) - c_1 \int (\dot{q}_1 - \dot{y}_1)dt = 0, \\ m_2 \frac{d\dot{y}_2}{dt} - b_2(\dot{y}_1 - \dot{y}_2) - c_2 \int (\dot{y}_1 - \dot{y}_2)dt = 0, \\ m_3 \frac{d\dot{y}_3}{dt} - b_3(\dot{y}_1 - \dot{y}_3) - c_3 \int (\dot{y}_1 - \dot{y}_3)dt = 0, \\ m_4 \frac{d\dot{y}_4}{dt} - b_4(\dot{y}_1 - \dot{y}_4) - c_4 \int (\dot{y}_1 - \dot{y}_4)dt = 0. \end{array} \right. \quad (1.7)$$

With regard to the electric circuit, the corresponding equation is derived based on the cause-and-effect relationships taking place in the incident 1-junction node with the edge of the fictitious effort source of the *effort* variable. In general terms, this equation defines the current distribution in the parallel connection of the capacitive element C_P , the resistance R_P and the current source S_A :

$$k_e(\dot{q}_1 - \dot{y}_1) = f_{27} + f_{28}, \quad e_{26} = e_{27} = e_{28} = U_C. \quad (1.8)$$

The equations characterizing the dynamics of the electric circuit are obtained by substituting the appropriate relationships (1.4) and (1.5)–(1.8):

$$k_e(\dot{q}_1 - \dot{y}_1) = C_P \frac{de_{27}}{dt} + \frac{1}{R_P} e_{28} \rightarrow C_P \frac{dU_C}{dt} + \frac{1}{R_P} U_C - k_e(\dot{q}_1 - \dot{y}_1) = 0. \quad (1.9)$$

Finally, the mathematical model of the system of harvesting energy from vibrating mechanical devices takes the form:

$$\left\{ \begin{array}{l} m_1 \ddot{y}_1 + b_2(\dot{y}_1 - \dot{y}_2) + c_2(y_1 - y_2) + b_3(\dot{y}_1 - \dot{y}_3) + c_3(y_1 - y_3) \\ \quad + b_4(\dot{y}_1 - \dot{y}_4) + c_4(y_1 - y_4) + b_1(\dot{y}_1 - \dot{q}_1) + c_1(y_1 - q_1) - k_e U_C = 0, \\ m_2 \ddot{y}_2 - b_2(\dot{y}_1 - \dot{y}_2) - c_2(y_1 - y_2) = 0, \\ m_3 \ddot{y}_3 - b_3(\dot{y}_1 - \dot{y}_3) - c_3(y_1 - y_3) = 0, \\ m_4 \ddot{y}_4 - b_4(\dot{y}_1 - \dot{y}_4) - c_4(y_1 - y_4) = 0, \\ C_P \dot{U}_C + \frac{1}{R_P} U_C + k_e(\dot{y}_1 - \dot{q}_1) = 0. \end{array} \right. \quad (1.10)$$

Derived dependencies are the formal basis for conducting computer simulations, mapping the efficiency of energy harvesting by the tested system. We assume that the system is affected by harmonic excitation q_1 with amplitude A and frequency

ω_W . Based on the adopted material and geometric dimensions, physical parameters characterizing the examined dynamic system were identified.

1.2.1 The Results of Model Tests of a Linear System

Model tests were carried out in relation to the linear system energy harvesting system from vibrating mechanical devices. We assume that the system is affected by harmonic excitation. The numerical quantities characterizing the tested system, necessary to perform computer simulations, are presented in Table 1.1.

In the first stage of model tests, dynamic characteristics were identified and sensitivity was tested in relation to selected physical parameters characterizing the examined system. The graph (Fig. 1.3) presents its own characteristics, which were plotted assuming that external excitation and system response are recorded on the inertia element m_1 .

The graphic image of dynamic characteristics clearly indicates that the discrete system is characterized by four resonance frequencies: $\omega_1 \approx 2.82$ rad/s, $\omega_2 \approx 4.48$ rad/s, $\omega_3 \approx 6.21$ rad/s and $\omega_4 \approx 8.65$ rad/s. Energy losses occurring in the system significantly reduce the amplitudes of vibrations of higher harmonic components.

Assuming that the stiffness c_1 of the energy harvesting system will increase, with the same values of other stiffnesses (Fig. 1.4a), then all resonance frequencies are shifted towards higher values of harmonic components. The largest shift

Table 1.1 Physical parameters of energy harvesting systems

Name	Symbol	Value
Concentrated masses	m_1	0.012 kg
	m_2	0.003 kg
	m_3	0.003 kg
	m_4	0.003 kg
Energy losses in a mechanical system	b_1	0.001 Nsm ⁻¹
	b_2	0.001 Nsm ⁻¹
	b_3	0.001 Nsm ⁻¹
	b_4	0.001 Nsm ⁻¹
Bending stiffness of beams	c_1	0.2 Nm ⁻¹
	c_2	0.15 Nm ⁻¹
	c_3	0.1 Nm ⁻¹
	c_4	0.05 Nm ⁻¹
Electrical circuit equivalent resistance	R_P	$1 \times 10^6 \Omega$
Electrical circuit equivalent capacity	C_P	72 nF
Electromechanical constant of piezoelectric	k_e	-3.985×10^{-5} N/V

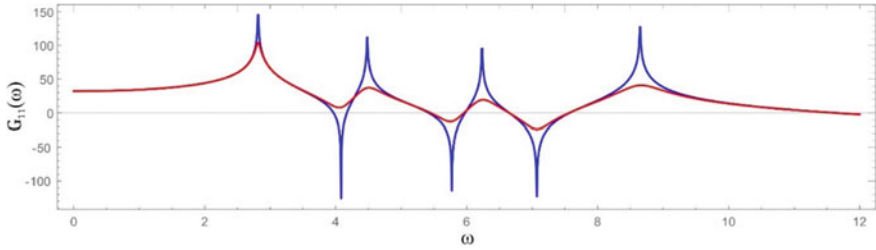


Fig. 1.3 Dynamic characteristics plotted for an undamped system (blue) and a damped system (red)

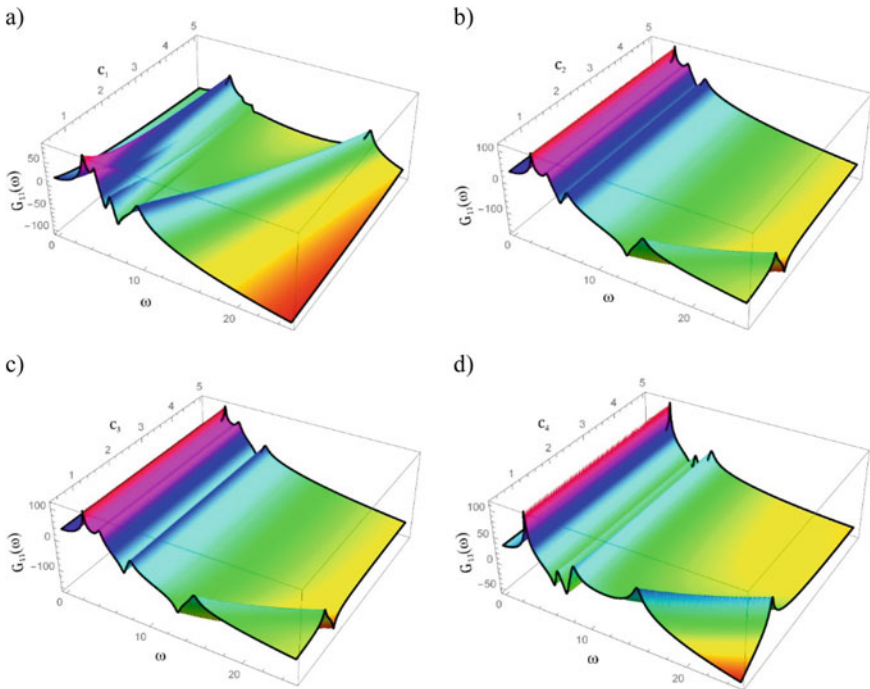


Fig. 1.4 Sensitivity functions illustrating the effect of stiffness on dynamic characteristics: **a** $c_2 = 0.15, c_3 = 0.1, c_4 = 0.05$, **b** $c_1 = 0.2, c_3 = 0.1, c_4 = 0.05$, **c** $c_1 = 0.2, c_2 = 0.15, c_4 = 0.05$, **d** $c_1 = 0.2, c_2 = 0.15, c_3 = 0.1$

occurs in relation to the resonance frequency with the highest harmonic component. The increase in stiffness c_2 has virtually no noticeable effect on the first two resonance frequencies (Fig. 1.4b). Whereas the third and fourth are shifted towards higher values. On a change in c_2 stiffness, the highest sensitivity is demonstrated by the resonance frequency with the highest harmonic component. With regard to the sensitivity of dynamic characteristics (Fig. 1.4c) to the change in stiffness c_3 , it

was found that this parameter has essentially no effect on the resonance frequency with the lowest harmonic component. The greatest impact is observed on the highest resonance frequency.

By changing the stiffness c_4 , it is possible to influence the location in the amplitude-frequency spectrum of all frequencies characterizing the dynamics of the system. Nevertheless, the least sensitivity occurs in relation to the frequency with the lowest harmonic component. To sum up these considerations, it should be stated that the greatest sensitivity to a change in any stiffness of the considered energy harvesting system concerns the highest harmonic component.

1.2.2 Efficiency of Harvesting Energy in a Linear System

As shown below, the subject of numerical research was the efficiency of harvesting energy from vibrating mechanical devices. Particular attention was paid to the voltage induced on piezoelectric electrodes. The results of computer simulations were depicted in the form of diagrams of the RMS voltage (Fig. 1.5). With dashed black lines, the resonant frequencies of the considered linear system with four degrees of freedom are depicted. However, the solid red line represents the effective value of the diagrams in the analysed range of excitation frequency variations.

Diagrams of the RMS voltage induced on piezoelectric electrodes are depicted in relation to dimensionless frequency $\omega = \frac{\omega_w}{\omega_B}$. Whereas, having regard to the assessment of the impact of all resonant frequencies of the system on the efficiency of energy harvesting, the base frequency was chosen $\omega_B = 6.21$ rad/s. In addition,

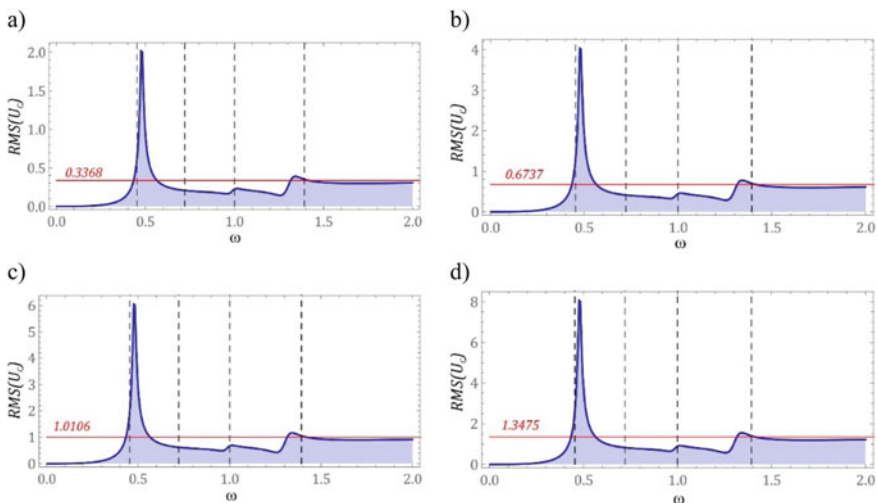


Fig. 1.5 Diagrams of the RMS voltage induced on piezoelectric electrodes obtained at the vibration amplitude of the external source: **a** $A = 0.001$ m, **b** $A = 0.002$ m, **c** $A = 0.003$ m, **d** $A = 0.004$ m

such presentation of the results of model tests will allow direct reference of the results of the linear system to the non-linear system analysed in the next chapter. The impact of the amplitude of the external source of excitation plays an important role on the efficiency of energy harvesting, as evidenced by the effective value of the diagrams (solid red line) (Fig. 1.5). It is worth noting that the increase in the effective value of diagrams is directly proportional to the amplitude of the external source of mechanical vibrations. In addition, regardless of the amplitude of the excitation, the geometric shape of the diagram does not change, which is a typical feature of linear dynamic systems. The tested system exhibits the best efficiency in obtaining energy in a narrow band located in close proximity to the lowest resonance frequency.

1.3 Energy Harvesting in a Non-linear System

This chapter presents the results of model tests of non-linear energy harvesting systems. Particular attention was focused on the two-bar system. Differential equations of motion, as in the case of a linear system, were derived from the topological structure of the bond graph. However, in this case, it was limited to presenting the structure of the bond graph and providing a mathematical model.

The subject of research is the energy harvesting system *DBEH (Double Beam Energy Harvester)*, composed of two flexible cantilever beams *I* and *II* (Fig. 1.6). It is a modification of the design solution of a classic three-wells transducer consisting of two permanent magnets mounted in a non-deformable frame. In the proposed construction solution, the movement of flexible beams was coupled with a linear Kelvin-Voigt *IV* element.

The mechanical properties of vibrating beams are mapped through c_{Bi} stiffness and b_{Bi} damping. At the same time, the stiffness of the beams subject to bending is identified on the basis of the equation $c_{Bi} = EI_{Bi}/L_i$, where I_{Bi} represents the moment

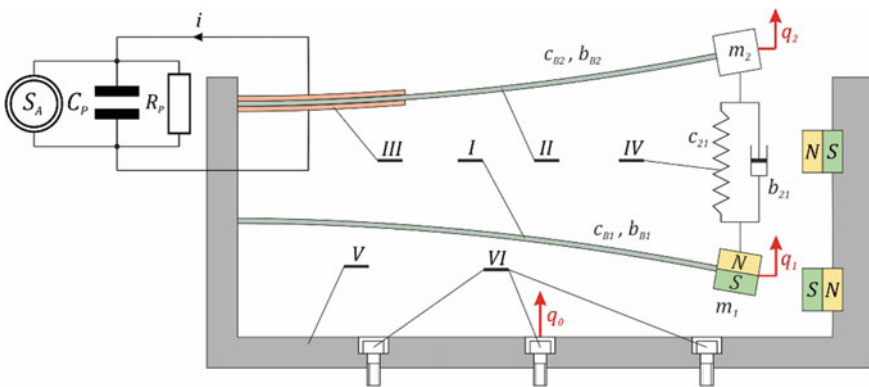


Fig. 1.6 Phenomenological model of the energy harvesting system

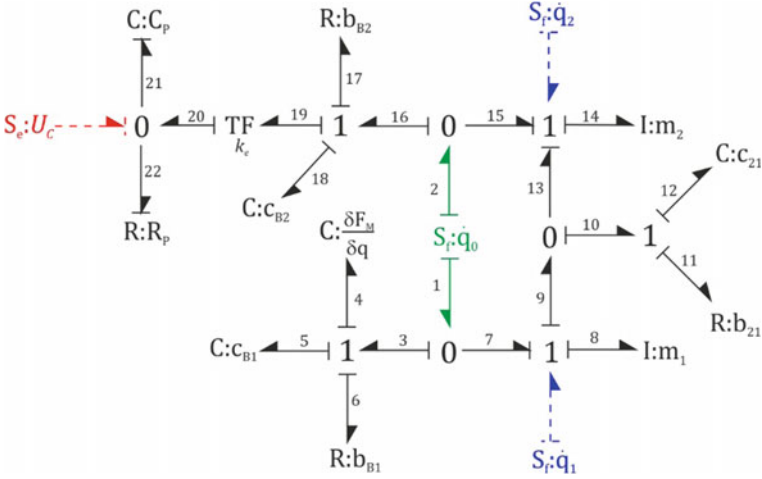


Fig. 1.7 Bond graph of a two-bar nonlinear system

of inertia of the cross-section. However, with respect to energy dissipating elements, additional laboratory tests are necessary due to the presence of internal and structural damping.

On the beam **II**, a piezoelectric transducer **III** is glued, on the electrodes of which, under the influence of elastic deformations, an electric charge is induced. Both deformable elements **I** and **II** are rigidly mounted in the **V** frame, which, through the fastening screws **VI**, creates a fixed connection with the vibrating component of the mechanical system. Fixed permanent magnets were mounted in the frame and arranged symmetrically with respect to the beam position **I**. The bond graph corresponding to the tested two-beam energy harvesting system is shown in Fig. 1.7.

Based on the bond graph, differential motion equations were derived, which are the formal basis for conducting quantitative and qualitative model tests. We note that during computer simulations, kinematic excitation was assumed, mapped by a harmonic function $q_0 = A \sin(\omega_W t)$. The general structure of differential equations of motion, capturing electromechanical couplings take the form:

$$\begin{cases} m_1 \dot{q}_1 + b_{E1}(\dot{q}_1 - \dot{q}_0) + c_{B1}(q_1 - q_0) + b_{21}(\dot{q}_1 - \dot{q}_2) + F_C + F_M = 0, \\ m_2 \dot{q}_2 + b_{B2}(\dot{q}_2 - \dot{q}_0) + c_{B2}(q_2 - q_0) - b_{21}(\dot{q}_1 - \dot{q}_2) - F_C - k_P u = 0, \\ C_P \dot{u} + \frac{u}{R_P} + k_e(\dot{q}_2 - \dot{q}_0) = 0. \end{cases} \quad (1.11)$$

The F_C value represents the excited force in the elastic element **IV**, which couples the movement of both beams. We assume its linear characteristic: $F_C = c_{21}(q_1 - q_2)$. Cause-effect relationships occurring between the forces of magnet interactions and displacements of beams is mapped by a polynomial function $F_M = c_{11}(q_1 - q_0) + c_{12}(q_1 - q_0)^3 + c_{13}(q_1 - q_0)^5$ according to Huang et al. [9].

The formal basis for conducting computer simulations is a system of differential equations that has been transformed to dimensionless form. The numerically effective structure of equations is obtained by introducing new variables defining the differences in displacements of beams **I** and **II**: $x_1 = q_1 - q_0$ and $x_2 = q_2 - q_0$. In addition, we assume that both beam elements are made of the same material and have the same geometrical structure as a consequence of which $c_{B2} = c_{B1} = c_B$ and $b_{B2} = b_{B1} = b_B$. Considering the above assumptions, idealizing the studied system, leads to a mathematical model whose generalized coordinates depend on dimensionless time $\tau = \omega_0 t$:

$$\begin{cases} \ddot{x}_1 + \delta_1 \dot{x}_1 + x_1 + \alpha x_1^3 + \beta x_1^5 - \delta_3 \dot{x}_2 - \eta_1 x_2 = \omega^2 A \sin(\omega \tau), \\ \mu \ddot{x}_2 + \delta_2 \dot{x}_2 + \eta_2 x_2 - \delta_3 \dot{x}_1 - \eta_1 x_1 - \kappa u = \mu \omega^2 A \sin(\omega \tau), \\ \dot{u} + \sigma u + \vartheta (\dot{x}_2) = 0. \end{cases} \quad (1.12)$$

where:

$$\begin{aligned} \mu &= \frac{m_2}{m_1}, c_z = c_{B1} + c_{21} + c_{11}, \alpha = \frac{c_{12}}{c_z}, \beta = \frac{c_{13}}{c_z}, \omega_0^2 = \frac{c_z}{m_1}, \omega_1^2 = \frac{c_{21}}{m_1}, \\ \omega_2^2 &= \frac{c_{B2} + c_{21}}{m_1}, \delta_1 = \frac{b_{B1} + b_{21}}{m_1 \omega_0}, \delta_2 = \frac{b_{B2} + b_{21}}{m_1 \omega_0}, \delta_3 = \frac{b_{21}}{m_1 \omega_0}, \eta_1 = \frac{\omega_1^2}{\omega_0^2}, \\ \eta_2 &= \frac{\omega_2^2}{\omega_0^2}, \sigma = \frac{1}{C_P R_P \omega_0}, \vartheta = \frac{k_e}{C_P}, \kappa = \frac{k_e}{m_1 \omega_0^2}, \omega = \frac{\omega_W}{\omega_0}. \end{aligned}$$

In the case of a non-linear system, it was decided to model research based on a dimensionless model, due to the speed of numerical calculations. The derived system of differential equations (1.12) is the formal basis for model tests of the considered energy harvesting system. The numerical values characterizing the considered dynamic system necessary to perform computer simulations are presented in Table 1.2.

The main goal of the results of computer simulations included in this work is to assess the impact of selected mechanical quantities on the efficiency of energy harvesting. At this point, we signal that all the results of numerical experiments were obtained assuming zero initial conditions. One of the features of nonlinear systems is the possibility of more than one solution, under given conditions characterizing the external source of motion. The impact of initial conditions in terms of co-existing solutions is not the subject of the analysis contained in this work. The results of numerical calculations were depicted in the form of diagrams of the RMS voltage induced on the electrodes of the piezoelectric transducer. Numerical calculations were carried out at the amplitude of the external source of excitation $A = 0.001$ m. In the case of a non-linear system, the base frequency to which the excitation frequency is referred to is $\omega_B = \omega_0 \approx 17$ rad/s.

The obtained results of computer simulations indicate that the system most effectively harvests energy if the rigidity of the coupling element assumes a value in the range $c_{21} = 1 \div 1.5$ N/m (Fig. 1.8). On the graphs (Fig. 1.9), the influence of the

Table 1.2 Physical parameters of energy harvesting systems

Name	Symbol	Value
Concentrated mass loading the beam <i>I</i>	m_1	0.0172 kg
Concentrated mass loading the beam <i>II</i>	m_2	0.006 kg
Energy losses in beam elements	b_{B1} and b_{B2}	0.01 Nsm ⁻¹
Bending stiffness of beam elements	c_{B1} and c_{B2}	0.05 Nm ⁻¹
Energy losses in the coupling spring	b_{21}	0.004 Nsm ⁻¹
Stiffness of the coupling element	c_{21}	1.5 Nm ⁻¹
Characteristics that reflect the forces of magnets	c_{11}	3.818 Nm ⁻¹
	c_{12}	-48,931 Nm ⁻³
	c_{13}	1.12×10^8 Nm ⁻⁵
Electrical circuit equivalent resistance	R_P	1×10^6 Ω
Electrical circuit equivalent capacity	C_P	72 nF
Electromechanical constant of piezoelectric	k_e	-3.985×10^{-5} N/V
Amplitude of forcing mechanical vibrations	A	0.001 m

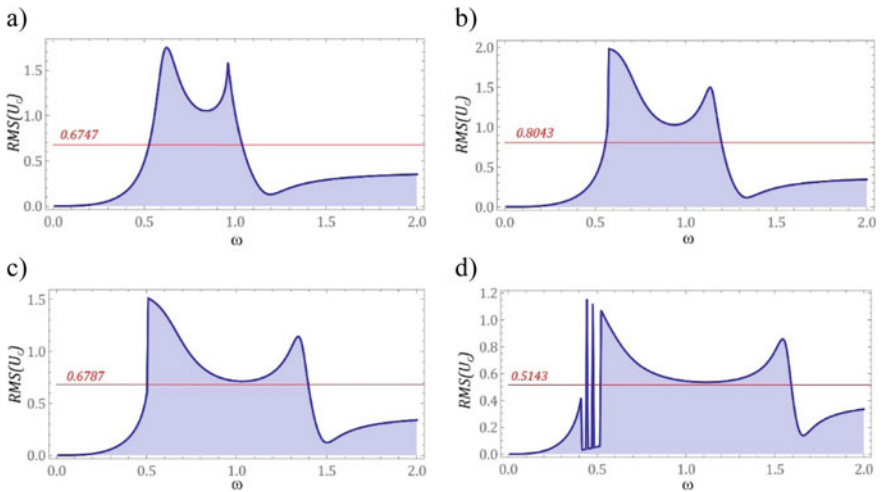


Fig. 1.8 Impact of coupling element rigidity on energy harvesting efficiency: **a** $c_{21} = 0.75$ N/m, **b** $c_{21} = 1.5$ N/m, **c** $c_{21} = 3$ N/m, **d** $c_{21} = 6$ N/m

external forcing source amplitude A on the amount of induced voltage is presented. For the numerical calculations, the stiffness of the coupling element of flexible beams was assumed $c_{21} = 1.2$ N/m.

As in the case of a linear system, the increase in excitation amplitude significantly improves the efficiency of energy harvesting from vibrating mechanical systems. In the case of linear systems, the shape of the function reflecting diagrams of the RMS voltage induced on piezoelectric electrodes does not change geometrically. In the case

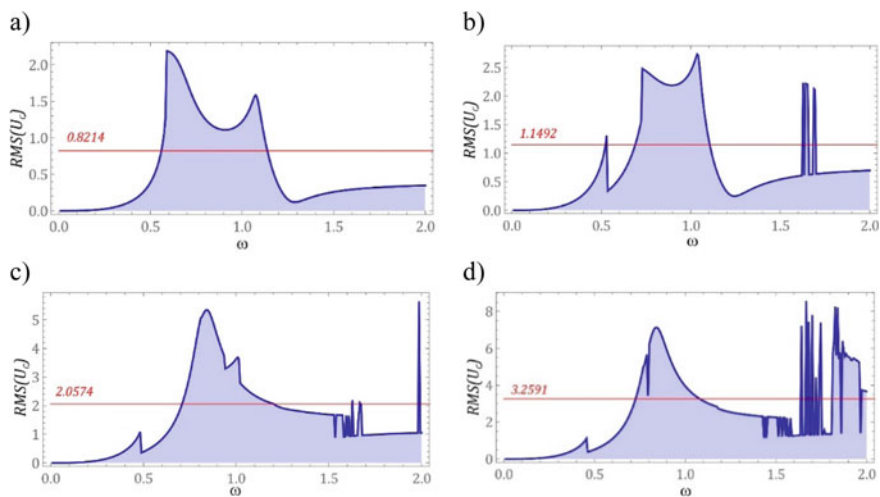


Fig. 1.9 Diagrams of the RMS voltage induced on piezoelectric electrodes obtained at the vibration amplitude of the external source: **a** $A = 0.001$ m, **b** $A = 0.002$ m, **c** $A = 0.003$ m, **d** $A = 0.004$ m

of non-linear systems, the profiles of plotted diagrams differ significantly (Fig. 1.9). The largest differences are observed in the range of high harmonic components of the amplitude-frequency spectrum.

1.4 Conclusions

Based on the conducted model research, it is possible to formulate the following conclusions:

- The amount of harvested energy increases as the vibration amplitude of the excitation source increases. This situation occurs both in relation to linear and nonlinear systems.
- Linear systems are able to efficiently harvest energy from vibrating mechanical devices when the excitation frequency is located in close proximity to the dominant resonant frequency of the linear system.
- Effective energy acquisition in a non-linear system is almost double that of a linear system as evidenced by the effective values of the diagrams drawn.
- Bonds graphs are an effective and fast method for deriving the equations of motion for energy harvesting systems.

Acknowledgements This work was supported by the program of the Ministry of Science and Higher Education in Poland under the project  DIALOG/DIALOG 0019/DLG/2019/10 in the years 2019–2021.

References

1. Zhang, Z., Xiang, H., Shi, Z., Zhan, J.: Experimental investigation on piezoelectric energy harvesting from vehicle/bridge coupling vibration. *Energy Convers. Manage.* **163**, 169–179 (2018)
2. Friswell, M.I., Ali, S.F., Bilgen, O., Adhikari, S., Lees, A.W., Litak, G.: Non-linear piezoelectric vibration energy harvesting from a vertical cantilever beam with tip mass. *J. Intell. Mater. Syst. Struct.* **23**, 1505–1521 (2012)
3. Kim, P., Yoon, Y.J., Seok, J.: Nonlinear dynamic analyses on a magnetopiezoelectric energy harvester with reversible hysteresis. *J. Nonlinear Dyn.* **83**, 182–1854 (2016)
4. Mitcheson, P.D., Miao, P., Stark, B.H., Yeatman, E.M., Holmes, A.S., Green, T.C.: Mems electrostatic micropower generator for low frequency operation. *Sens. Actuators, A* **115**, 523–529 (2004)
5. Kim, Y.S.: Analysis of electromotive force characteristics for electromagnetic energy harvester using ferrofluid. *J. Magn.* **20**, 252–257 (2015)
6. Shahruz, S.M.: Design of mechanical band-pass filters for energy scavenging. *J. Sound Vib.* **292**, 987–998 (2006). <https://doi.org/10.1016/j.jsv.2005.08018>
7. Xie, Z., Kitio Kwuimy, C.A., Wang, T., Ding, X., Huang, W.: Theoretical analysis of an impact-bistable piezoelectric energy harvester. *Eur. Phys. J. Plus* **134**, 190 (2019). <https://doi.org/10.1140/epjp/i2019-12569-2>
8. Younesian, D., Alam, M.R.: Multi-stable mechanisms for high-efficiency and broadband ocean wave energy harvesting. *Appl. Energy* **197**, 292–302 (2017). <https://doi.org/10.1016/j.apenergy.2017.04.019>
9. Huang, D., Zhou, S., Litak, G.: Theoretical analysis of multi-stable energy harvesters with high-order stiffness terms. *Commun. Nonlinear Sci. Numer. Simulat.* **69**, 270–286 (2019). <https://doi.org/10.1016/j.cnsns.2018.09.025>
10. Litak, G., Friswell, M.I., Kitio Kwuimy, C.A., Adhikari, S., Borowiec, M.: Energy harvesting by two magnetopiezoelectric oscillators with mistuning. *Theor. Appl. Mech. Lett.* **2**, 043009 (2012)
11. Granados, A.: Invariant manifolds and the parametrization method in coupled energy harvesting piezoelectric oscillators. *Physica D* **351–352**, 14–29 (2017). <https://doi.org/10.1016/j.physd.2017.04.003>
12. Kim, I.H., Jung, H.J., Lee, B.M., Jang, S.J.: Broadband energy-harvesting using a two degree-of-freedom vibrating body. *Appl. Phys. Lett.* **98**, 214102 (2011). <https://doi.org/10.1063/1.3595278>
13. Kadje, A.N., Wofo, P.: Effects of springs on a pendulum electromechanical energy harvester. *Theor. Appl. Mech. Lett.* **4**, 063001 (2014)
14. Malaji, P.V., Ali, S.F., Adhikari, S., Friswell, M.I.: Analysis of harvesting energy from mistuned multiple harvesters with and without coupling. *Procedia Eng.* **144**, 621–628 (2016)
15. Kumar, R., Gupta, S., Ali, S.F.: Energy harvesting from chaos in base excited double pendulum. *Mech. Syst. Signal Process.* **124**, 49–64 (2019)

Chapter 2

Bond Graph Based Synthesis of Generic Power Split Modelling for Epicyclic Four-Speed Gears



G.-H. Geitner and G. Komurgoz

Abstract Planetary gears are applied in a wide range of applications. For example power supply of airplane auxiliaries or hybrid electric vehicles benefit from such-like gears. Thus heavy truck hybrid propulsion based on epicyclic four-speed gears guarantees internal combustion engine optimal working point. Assuming that generic power split modelling for this gear type marks a research focus for some time already. The Bond Graph method as a member of a very closely related power flow modelling tool family is especially very well suitable to meet this aim. The paper argues this message not via common analysis but via synthesis, introduces folding operation as natural effective means for getting vectorial models, shows variants and illustrates generalization via simulation results and numerical examples. Synthesis approach suggested can also serve as a starting point for generation of more detailed models or research and comparison of new epicyclic gear constructions.

Keywords Power split · Planetary gear · Equivalent dynamics · Tooth contact · Modelling variants

2.1 Introduction

Modern vehicle drive concepts include hybrid drives as required intermediate stage from conventional combustion engines too pure electric traction systems [7]. Power split and power addition again is a crucial issue in hybrid traction solutions. Although there are electric components under development [6] typically planetary gear variants are standard solutions for this issue. Especially heavy trucks [20] but also passenger cars may take advantage from compound epicyclic gears with four shafts. Such

G.-H. Geitner (✉)
TU Dresden, Dresden, Germany
e-mail: gert-helge.geitner@tu-dresden.de

G. Komurgoz
Istanbul Technical University, Istanbul, Turkey
e-mail: komurgoz@itu.edu.tr

constructions permit to apply diverse power flow control strategies between combustion engine driven at point of lowest losses, two electrical machines optionally driven as motor or generator and load. Thus, flexible general power flow models of such devices are of current interest.

The Bond Graph as modelling tool is a promising approach, not only, but also for mechanism and machine science regarding analysis of kinematics [9, 26] and dynamics [8]. It is a perfect choice for power flow description based on generalized Kirchhoff's rules [8], for simulation of dynamic systems and for energy efficiency calculation based on integration and cycle time [19]. This is true also and especially for planetary gears. Please compare sections "Kinematical Analysis of Variants of Wind Turbine Drive by Means of Graphs" and "Graph-Based Algorithm for the Evaluation of the Mechanical Efficiency of Epicyclic Gear Drive in Hybrid Scooters" in [33] for other methods to model planetary gears. Please see section "Bond Graphs in System Modelling" in [33] for introduction and definitions regarding BG's. For simplification we will neglect modulation of sources and only apply source symbols "SE" or "SF". In order to facilitate simulation based on models introduced in this section we restrict suggestions to integral causality.

Typical methods to get BG models are as following. Starting from circuit diagrams, sketches or similar specifications the BG model immediately may be created without equations. This approach works fine for easy systems of relatively limited extent. As a precondition it is of course necessary to be really familiar with this method. Only after BG model construction equation establishing takes place. On the contrary, however, as result of an analysis the more general approach starts with a complete set of mathematical equations and only then this mathematical description is graphically programmed via BG icons.

In this section, it is planned to implement a rather hybrid approach. Starting from basic considerations regarding the operating principle of planetary gears, it is intended to develop a BG model synthesis that takes place step by step.

Modelling by means of BG may take benefit using vectorial connections instead of scalar connections between elements or at least realization of mixed connections. This requires power variable definitions of type vector and parameter definitions of type matrix or vector. Such approaches are very well suited for systems featuring similar parallel functionalities. It will be shown that planetary gears provide a good example for this purpose. Please note, we will not use different bond symbols depending on scalar bonds or vectorial bonds as may be found in some references [23], but mark relevant power variables via underlined characters. Suchlike modelling symbolism emphasizes structural identity at simultaneous clearness and eases the application of folding operation as natural tool for BG modelling. In general there are two possibilities to simulate BG models. Special BG software offers direct customization for bidirectional connections and causality settings for instance. In contrast general software solutions have to be enhanced to meet features necessary for BG modelling.

The latter option, which is usually based on some kind of add-on, is planned to be implemented. Whereas block diagrams obtained from BG models should be restricted to very easy examples for first training only. We suggest to prefer standard software enhanced via add-on solutions, just as for instance Simulink and add-on block

library BG ver. 2.1 [13]. Such an approach is planned to be implemented because it ensures access to known tools as transfer function generation, data processing or control design. However, a menu-driven add-on may manage simple customization to specific BG needs.

2.2 Simplified Equivalent Dynamics

In order to develop the BG model via synthesis a suitable starting point is necessary. Any dynamics or friction is still neglected. It is known that a basic planetary gear set features a fixed determined linkage between the three external speeds. Two speeds ω_x and ω_y involve the third speed ω_z . This is true for any specific construction (2.1a). Moreover, exactly one correlation between the speeds of sun, ring and carrier and the speeds ω_1 till ω_3 has to meet equation structure (2.1b). Hence geometrically determined parameter n_1 , which is identically to parameter R in [25], defines K_{a1}^* and K_{b1} in very simple fashion. This Willis Equation (2.1b) has to be rearranged as presented in (2.1c) if the model computes ω_3 but the power flow reference direction is positive. The first BG fragment Fig. 2.1a symbolizes graphically (2.1c) based on definitions (2.1d). Figure 2.1a demonstrates that any minus sign may be handled either by reverse power flow direction or parameter sign. Speeds are considered to be power variables of type flow as common BG practice according to Maxwell analogy [8]. Consequently introduction of torque T_3 as power flow variable of type effort

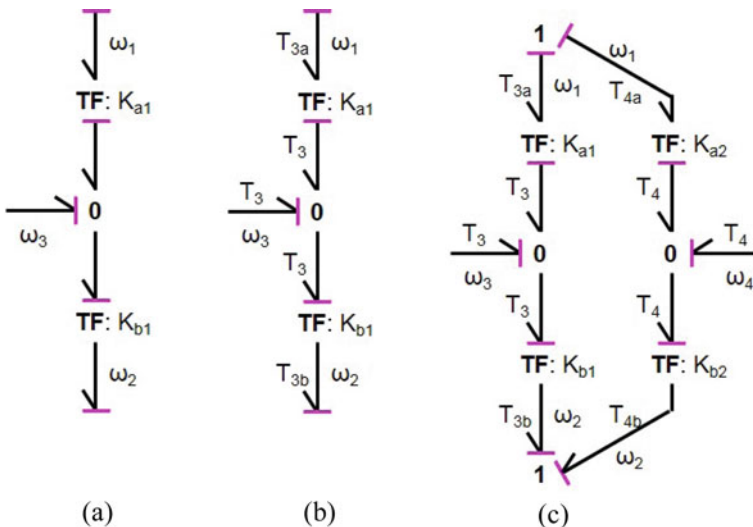


Fig. 2.1 Bond graph fragments for generic power split modelling of epicyclic four-speed gears based on synthesis ignoring spring damper tooth contacts: **a** BG equivalent of Willis equation; **b** introduction of torques via power balance; **c** introduction of a fourth speed

results in Fig. 2.1b. In doing so the 0-node acts as torque distributor and TF-elements naturally provide with torques (2.1e) based on power balance.

$$\omega_z = K_{a1}^* \cdot \omega_x + K_{b1} \cdot \omega_y; \quad (z, x, y) \in \{(3, 1, 2), (1, 2, 3), (2, 3, 1)\} \quad (2.1a)$$

$$\omega_3 = n_1 \cdot \omega_1 + (1 - n_1) \cdot \omega_2 \quad (2.1b)$$

$$-\omega_3 = +(-n_1) \cdot \omega_1 - (1 - n_1) \cdot \omega_2 \quad (2.1c)$$

$$K_{a1} = -K_{a1}^* = -n_1; \quad K_{b1} = 1 - n_1 \quad (2.1d)$$

$$T_{3a} = K_{a1} \cdot T_3; \quad T_{3b} = K_{b1} \cdot T_3 \quad (2.1e)$$

In the next step one can imagine any epicyclic four-speed gear to be composed of two basic planetary gear sets. Two rigid mechanical “connections” of both sets reduce the number of different speeds from six to four. Practical realisation of such “connections” causes common components as for instance common ring and common carrier, which is true for Ravigneaux gear. Anyway, since one planetary gear set yields one holonomic constraint constructions including two sets yield two holonomic constraints. This means two speeds ω_1 and ω_2 cause two more speeds ω_3 and ω_4 now and modelling needs two parallel paths in terms of Fig. 2.1b. As a result follows Fig. 2.1c. The 1-nodes act as flow distributors and ensure that speeds ω_1 and ω_2 may be “used” twice each with.

Similar to the supply of torques T_3 and T_4 to 0-nodes also both 1-nodes require the supply of speeds ω_1 and ω_2 via bonds. Finally addition of sources completes the first BG model specified in Fig. 2.2a. This model is valid for static torque and speed balance and ignores any friction or dynamics. Assuming analog definitions of K_{ai} , K_{bi} , T_{ja} and T_{jb} as well as $i = \in \{1, 2\}$ and $j = \in \{3, 4\}$ it is possible to prove the total power balance (2.2a) of this model by means of (2.2b). Please take special note of different source causalities in BG Fig. 2.2a as a consequence of static modelling of a system featuring dynamics and holonomic constraints in fact. Nevertheless, for some tasks such a kind models may be sufficiently [21].

$$T_1\omega_1 + T_2\omega_2 + T_3\omega_3 + T_4\omega_4 \stackrel{!}{=} 0 \quad (2.2a)$$

$$\begin{aligned} [-n_1T_3\omega_1 - n_2T_4\omega_1] & \quad +[-(1 - n_1)T_3\omega_2 - (1 - n_2)T_4\omega_2] + \dots \\ [+n_1\omega_1T_3 + (1 - n_1)\omega_2T_3] & \quad +[+n_2\omega_1T_4 + (1 - n_2)\omega_2T_4] & = 0 \end{aligned} \quad (2.2b)$$

The next synthesis step changes torque balances both at node 1_a and node 1_b via introduction of friction torques ΔT_{iF} and dynamic torques ΔT_{id} and thus allows all sources for effort causality (2.3a/b). Consequently both scalar torques ΔT_i have to

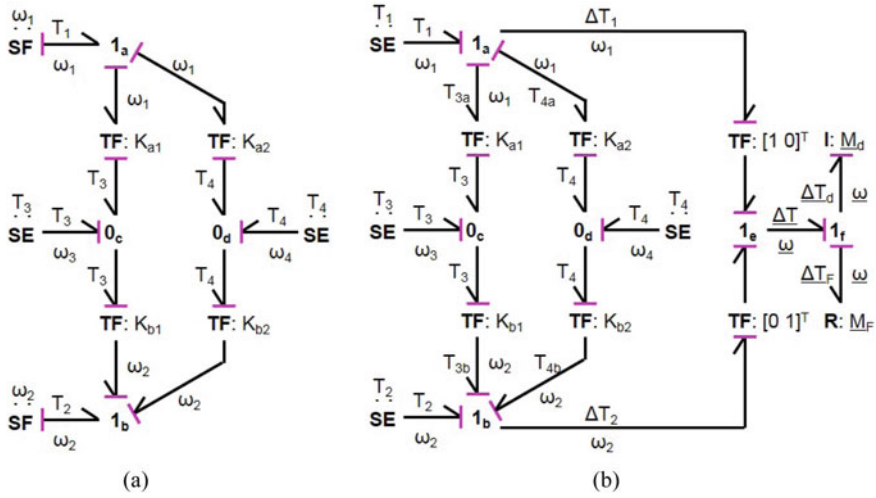


Fig. 2.2 Bond graph models for generic power split modelling of epicyclic four-speed gears taking Fig. 2.1c as starting point: **a** pure statically scalar BG model; **b** introduction of equivalent dynamics via hybrid scalar and vectorial modelling

drive a dynamic subsystem which now has to determine ω_1 and ω_2 . This subsystem again represents a substituted mechanical system via two fictive shafts considering friction and inertia of four real shafts. The reason for this are mentioned holonomic constraints. Furthermore, for clearness purposes, this new part of the model should be a vectorial one in order to generally imply component coupling of the equivalent dynamical system. Based on definitions (2.4a/b) and vectorial equation of motion (2.5) arises Fig. 2.2b. TF-elements connect scalar and vectorial model parts, I-type energy storage operates in integral mode as desired and matrices \underline{M}_F and \underline{M}_d contain fictive friction resp. inertia elements (2.6). Nodes 1_e and 1_f could be combined, but separated modelling permits access to $\underline{\Delta T}$ and improves representation.

$$T_1 - T_{3a} - T_{4a} = \Delta T_1 = \Delta T_{1d} + \Delta T_{1F} \neq 0 \quad (2.3a)$$

$$T_2 + T_{3b} + T_{4b} = \Delta T_2 = \Delta T_{2d} + \Delta T_{2F} \neq 0 \quad (2.3b)$$

$$\underline{\Delta T} = \begin{bmatrix} \Delta T_1 \\ \Delta T_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \cdot \Delta T_1 + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \cdot \Delta T_2 \quad (2.4a)$$

$$\omega_1 = [1 \ 0] \cdot \underline{\omega} = [1 \ 0] \cdot \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix}; \quad \omega_2 = [0 \ 1] \cdot \underline{\omega} = [0 \ 1] \cdot \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix} \quad (2.4a)$$

$$\underline{\Delta T} = \underline{\Delta T}_F + \underline{\Delta T}_d = \underline{M}_F \cdot \underline{\omega} + \underline{M}_d \cdot \frac{d\underline{\omega}}{dt} \quad (2.5)$$

$$\underline{\underline{M}}_F = \begin{bmatrix} K_{F11} & K_{F12} \\ K_{F21} & K_{F22} \end{bmatrix}; \quad \underline{\underline{M}}_d = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \quad (2.6)$$

There are two essential preconditions for easy identification of matrix elements in (2.6). Firstly all real four shafts are modeled by means of a motion equation structure like (2.7), whereat T_j symbolizes total external torque and K_{Fj} or J_j symbolize total friction coefficient resp. total inertia of the particular shaft. Secondly total parameters also have to include parts internal moved only. If applicable then (2.1e) and (2.3a/b) result in (2.8a/b). Potentially elastic shafts must be modeled separately.

$$T_j = K_{Fj} \cdot \omega_j + J_j \cdot \frac{d\omega_j}{dt} \Rightarrow T_j = (K_{Fj} + sJ_j) \cdot \omega_j = K_{\Delta j} \cdot \omega_j; j \in \{1, \dots, 4\} \quad (2.7)$$

$$T_1 - K_{a1}T_3 - K_{a2}T_4 = \Delta T_1 = K_{F11}\omega_1 + K_{F12}\omega_2 + sJ_{11} \cdot \omega_1 + sJ_{12} \cdot \omega_2 \quad (2.8a)$$

$$T_2 + K_{b1}T_3 + K_{b2}T_4 = \Delta T_2 = K_{F21}\omega_1 + K_{F22}\omega_2 + sJ_{21} \cdot \omega_1 + sJ_{22} \cdot \omega_2 \quad (2.8b)$$

Moreover, consideration of (2.1b), (2.1d) and (2.7) in (2.8a/b) gives (2.9a/b). Analysis of (2.9a/b) directly produces needed elements of matrix $\underline{\underline{M}}_F$ (2.10). Please notice a full structural identity regarding friction and inertia formulas. Therefore (2.10) gives results for friction matrix $\underline{\underline{M}}_F$ elements only. Inertia matrix $\underline{\underline{M}}_d$ elements follow by analogy.

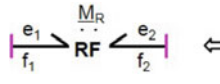
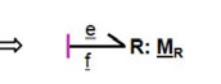

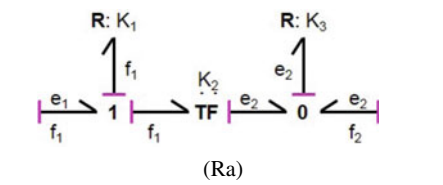
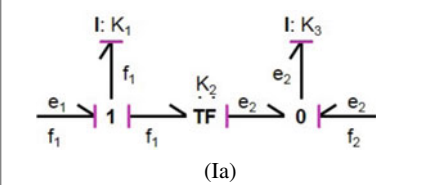
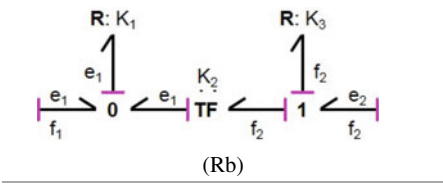
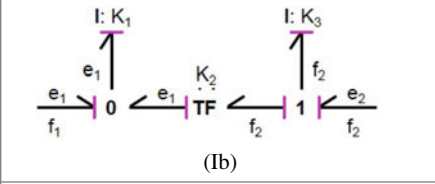
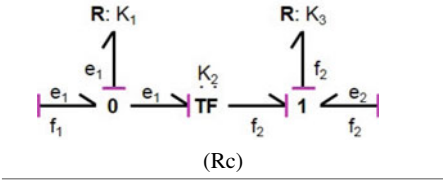
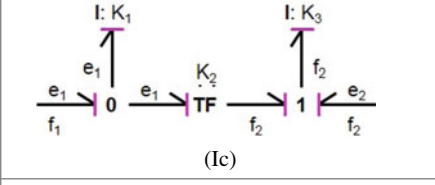
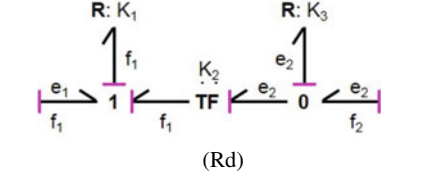
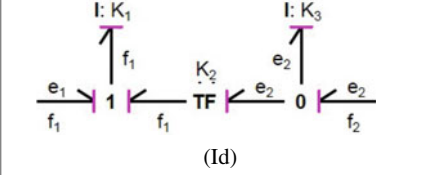
$$K_{\Delta 1}\omega_1 - K_{a1}K_{\Delta 3} \underbrace{(K_{b1}\omega_2 - K_{a1}\omega_1)}_{=\omega_3} - K_{a2}K_{\Delta 4} \underbrace{(K_{b2}\omega_2 - K_{a2}\omega_1)}_{=\omega_4} = \Delta T_1 \quad (2.9a)$$

$$K_{\Delta 2}\omega_2 + K_{b1}K_{\Delta 3} \underbrace{(K_{b1}\omega_2 - K_{a1}\omega_1)}_{=\omega_3} + K_{b2}K_{\Delta 4} \underbrace{(K_{b2}\omega_2 - K_{a2}\omega_1)}_{=\omega_4} = \Delta T_2 \quad (2.9b)$$

$$\begin{aligned} K_{F11} &= +K_{F1} + K_{F3}K_{a1}^2 + K_{F4}K_{a2}^2; & K_{F12} &= -K_{F3}K_{a1}K_{b1} - K_{F4}K_{a2}K_{b2} \\ K_{F12} &= -K_{F3}K_{a1}K_{b1} - K_{F4}K_{a2}K_{b2}; & K_{F22} &= +K_{F2} + K_{F3}K_{b1}^2 + K_{F4}K_{b2}^2 \end{aligned} \quad (2.10)$$

Hybrid scalar and vectorial model Fig. 2.2b may be modified in different ways. Since the right part in Fig. 2.2b also may be modelled via non-basic BG elements of type field, compare [18], it is possible to convert this part into scalar representation. The general conversion of n-port fields was discussed in [4], whereas [5] presented solutions for 3-port fields. Table 2.1 arranges solutions for 2-port fields needed here. It makes clear conversion is not unique, gives an analytical junction structure only without physical meaning [4] but exposes essential transformational couplings and state variables may be associated with each scalar storage element [5]. Mathematical feature of symmetric matrices is a precondition. Many technical systems meet this condition just as gear model Fig. 2.2b or most electric machine BG models [12]

Table 2.1 Equivalent substitution variants (Ra) till (Rd) for BG R-fields and (Ia) till (Id) for BG I-fields with two scalar ports, symmetrical matrices sine qua non; for parameters K_1 till K_3 equivalent to parameters a, b and d see Table 2.2

Flow causal R-field		Integral causal I-type storage field
$\underline{e} = \underbrace{\begin{bmatrix} a & b \\ b & d \end{bmatrix}}_{=M_R} \underline{f};$ $e_1 = a f_1 + b f_2;$ $e_2 = b f_1 + d f_2;$	$\underline{e} = \begin{bmatrix} e_1 & e_2 \end{bmatrix}^T;$ $\underline{f} = \begin{bmatrix} f_1 & f_2 \end{bmatrix}^T;$	$\underline{f} = \underbrace{\begin{bmatrix} a & b \\ b & d \end{bmatrix}}_{=M_I} \int \underline{e} dt;$ $e_1 = a \int f_1 dt + b \int f_2 dt$ $e_2 = b \int f_1 dt + d \int f_2 dt$
		
 <p style="text-align: center;">(Ra)</p>	 <p style="text-align: center;">(Ia)</p>	
 <p style="text-align: center;">(Rb)</p>	 <p style="text-align: center;">(Ib)</p>	
 <p style="text-align: center;">(Rc)</p>	 <p style="text-align: center;">(Ic)</p>	
 <p style="text-align: center;">(Rd)</p>	 <p style="text-align: center;">(Id)</p>	

but care has to be taken for exceptions such as synchronous reluctance machine BG model considering stator iron losses.

However, Fig. 2.3 gives a possible complete scalar model originating from Fig. 2.2b. Table 2.2 delivers corresponding parameters related to conversions Table 2.1. Joining of nodes 1_a and 1_h resp. 1_b and 1_g should not be implemented. It would reduce readability and violate symmetrical representation.

Then again it could be of interest to covert Fig. 2.2b into a compact complete vectorial BG representation as part of larger models. The BG method involves a powerful tool to reach this aim called folding operation. Certain symmetry provides the essential precondition. Hybrid scalar and vectorial models or different reference

Fig. 2.3 Scalar BG model of right part Fig. 2.2b based on Tables 2.1 and 2.2 variant Rb/Id

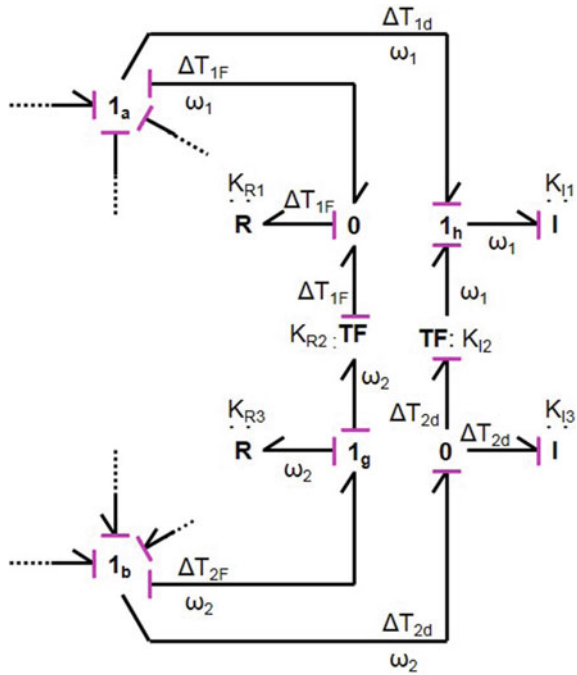
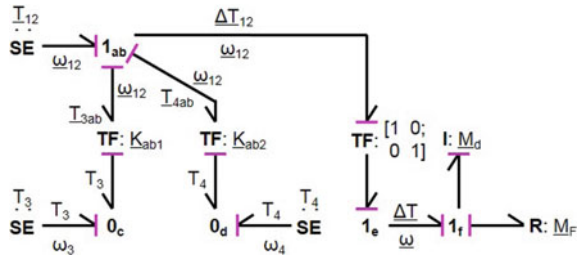


Table 2.2 Parameters for equivalent substitution variants Table 2.1; I-field features: sign of K_2 is positively if preferred power transfer direction is “off” regarding 0-node connection with TF, parameter at 1-node side is unmodified applied; this is true vice versa regarding R-field node types

Variants Table 2.1	K_1	K_2	K_3	R-field	I-field
				Node structure	Reference direction
Rb, Id	a	+b/a	$d - \frac{b^2}{a}$	0 ... 1: to	1 ... 0: off
Rc, Ia		-b/a		0 ... 1: off	1 ... 0: to
Rd, Ib	$a - \frac{b^2}{d}$	-b/d	d	1 ... 0: off	0 ... 1: to
Ra, Ic		+b/d		1 ... 0: to	0 ... 1: off

Fig. 2.4 Vertical folding of BG Fig. 2.2b



directions do not prevent practicability. The latter case only causes reversal of the signs. Thus, vertical folding of BG Fig. 2.2b gives Fig. 2.4 at a first step. Parameters and power variables associated with may be learnt from (2.11 a/b).

Simplification rules allow it to cancel nodes 0_c , 0_d and 1_e as well as TF-element at right side and permit to summarize nodes 1_{ab} and 1_e . As intermediate result of this arises Fig. 2.5a. Subsequently horizontal folding gives the final compact vectorial BG Fig. 2.5b based on parameters and power variables (2.12). Please note \underline{K}_{ab} definition refers to reference direction. Special case pure static relationship of speeds and torques, i.e. $\underline{M}_F = 0$ and $\underline{\Delta T}_d = 0$, symbolizes Fig. 2.5c which corresponds to Fig. 2.2a.

$$\underline{K}_{ab1} = [-n_1 - (1 - n_1)]; \underline{K}_{ab2} = [-n_2 - (1 - n_2)]; \underline{\omega}_{12} = \begin{bmatrix} \omega_1 \\ \omega_2 \end{bmatrix}; \underline{T}_{12} = \begin{bmatrix} T_1 \\ T_2 \end{bmatrix} \quad (2.11a)$$

$$\underline{T}_{3ab} = \underline{K}_{ab1}^T T_3; \underline{T}_{4ab} = \underline{K}_{ab2}^T T_4; \omega_3 = \underline{K}_{ab1} \underline{\omega}_{12}; \omega_4 = \underline{K}_{ab2} \underline{\omega}_{12} \quad (2.11b)$$

Care has to be taken regarding partial inversion of reference direction as applied for first vectorial folding Fig. 2.4. In order to obtain correct BG modelling it is

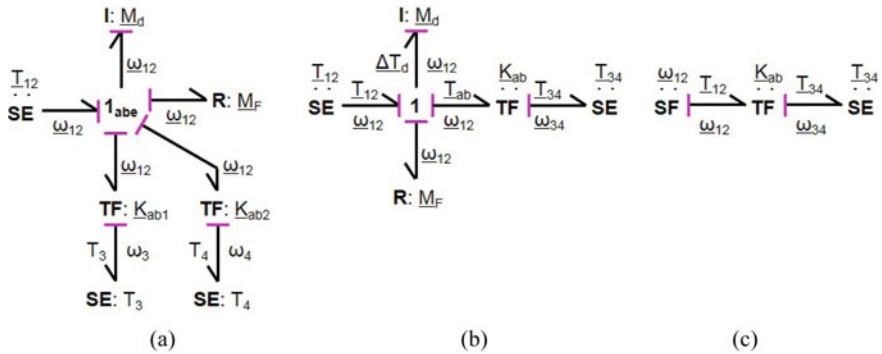


Fig. 2.5 Compact vectorial BG variants ignoring spring damper tooth contacts: **a** intermediate step via simplification of Fig. 2.4; **b** reconfigured final result after horizontal folding of Fig. 2.5a—with dynamics and friction; **c** simple static conversion

necessary to preserve same sign of the vectorial parameter for both operation directions of TF or GY elements. As a rule it may be stated to keep the signs for nodes implementing subtractions, cf. Fig. 2.2a left above, and to modify some signs for nodes implementing additions, cf. Fig. 2.2a left below. Otherwise reversed signs of particular power variables may occur.

$$\underline{K}_{ab} = \begin{bmatrix} -n_1 - (1 - n_1) \\ -n_2 - (1 - n_2) \end{bmatrix} = \begin{bmatrix} K_{a1} - K_{b1} \\ K_{a2} - K_{b2} \end{bmatrix}; \underline{T}_{ab} = \underline{K}_{ab}^T \underline{T}_{34}; \underline{\omega}_{34} = \underline{K}_{ab} \underline{\omega}_{12} \quad (2.12)$$

Direct continuation of synthesis based on Fig. 2.5b in order to introduce elastic tooth contacts is not possible. This is caused by inherent consideration of holonomic constraints. On the other hand it has to be emphasized that modelling introduced in this subsection is universally valid for any epicyclic four-speed gear. Customization to any specific realization only needs to determine n_1 and n_2 .

2.3 Spring Damper Tooth Contact

Modelling introduced in subsection before will be sufficiently in many cases. Anyway, some application cases will require more detailed analysis of power flow. If there are some significant dead zones or torsions for instance special gear type dependent scalar modelling will be necessary. But reducing tooth contacts to general elastic connections based on spring and damper in parallel allows generation of generic vectorial BG models similarly to section before. Two conditions must be met: firstly total inertia and total friction for each type of shaft has been defined; secondly, each kind of tooth contact and planet type has been considered just once. Consequently minimum number $n_{st} = 6$ of shaft types and minimum number $n_{tc} = 4$ of tooth contacts result if the epicyclic four-speed gear is built from number $n_{gs} = 2$ standard planetary gear sets. This is explained by number $n_{be} = 3$ of basic elements sun, ring and carrier per set, number $n_p = 1$ of planet types in between sun and ring and number $n_{tc} = 2$ of different kind tooth contacts per set as well as number $n_{rc} = 2$ of rigid connections in between sets (2.13a/b).

$$n_{st} = n_{gs} \cdot n_{be} - n_{rc} + n_{gs} \cdot n_p = 2 \cdot 3 - 2 + 2 \cdot 1 = 6 \quad (2.13a)$$

$$n_{tc} = n_{gs} \cdot n_{tc} = 2 \cdot 2 = 4 \quad (2.13b)$$

In this manner spring and damper elements decouple the shafts. Hence, basic equations of motion can be merged vectorial. Thereby inertia matrix \underline{M}_{dyn} and friction coefficient matrix \underline{M}_{Fric} are main diagonal matrices only, whereas torque vectors for input \underline{T} , dynamics $\underline{\Delta T}_{Dyn}$, friction \underline{T}_{Fric} and load \underline{T}_{SD} as well as speed vector $\underline{\omega}$ feature just as much elements as different shaft types n_{st} are defined (2.14).

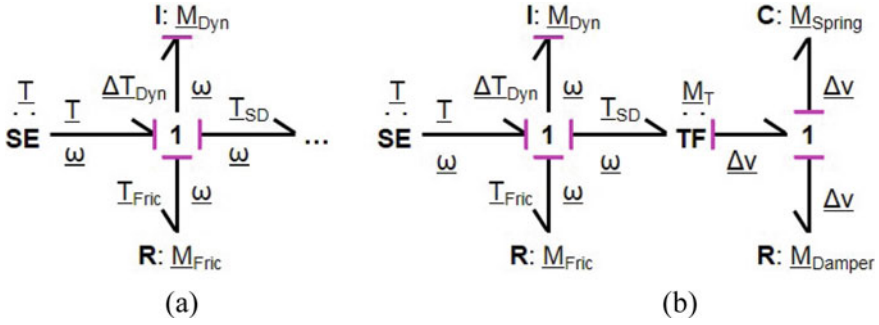


Fig. 2.6 Compact vectorial BG model inclusively of spring damper tooth contacts: **a** BG fragment of unconnected motion equations—see (2.14); **b** complete compact vectorial BG model

$$\underline{\Delta T}_{Dyn} = \underline{T} - \underline{T}_{Fric} - \underline{T}_{SD}; \underline{T}_{Fric} = \underline{M}_{Fric} \underline{\omega}; \underline{\omega} = \underline{M}_{Dyn} \int \underline{\Delta T}_{Dyn} dt + \underline{0} \quad (2.14)$$

In a certain sense BG fragment Fig. 2.6a as symbolization of (2.14) may be understood as further synthesis of Fig. 2.5b. Concerning Fig. 2.5b this means cut off the right side torque source, means increased dimensions for left side elements, i.e. all torque sources of external shafts plus zero sources for planet shafts are merged in vector \underline{T} , and means inverse reference direction for load, i.e. $\underline{T}_{ab} = -\underline{T}_{SD}$. Henceforth, the new TF-element parameter matrix \underline{M}_T joins dynamic spring and damper forces at one side and shaft torques at the other side. So TF-element is the crucial connection element. It connects rotational and translational movements. Due to the construction principle of a standard planetary gear set always three particular rotational speeds $\omega_x \ x = \{1 \dots n_{st}\}$ contribute to one particular translational speed $\Delta v_y \ y = \{1 \dots n_{tc}\}$. Thus elements of matrix \underline{M}_T may be found via superposition principle or based on Willis equation. For the latter case remodeled static Willis equation has to be thought having a dynamic term in sense of (2.3a) or (2.3b) implied for speeds. Since the specific tooth contacts are decoupled as well like shafts, damping coefficient matrix \underline{M}_{Damper} and spring coefficient matrix \underline{M}_{Spring} again are main diagonal matrices only, whereas speed vector $\underline{\Delta v}$ features just n_{tc} elements. Figure 2.6b shows resulting complete compact vectorial BG model with spring damper tooth contacts.

Decomposition of Fig. 2.6b into two rotational vectorial branches would better convey principle construction of epicyclic four-speed gears based on two standard planetary gear sets. This possible synthesis step may be named as unfolding operation. Right side 1-node with appended spring and damper modelling serves as a symmetry axis. But 1-nodes are not able to add any flow variables. However, just this is necessary if vector $\underline{\Delta v}$ is subdivided into two subcomponents. Insertion of a 0-node solves the problem and Fig. 2.7 presents corresponding BG. Both sides are fed with same spring damper forces via 0-node. Decomposition of \underline{M}_T into \underline{M}_{T1} and \underline{M}_{T2} ensures correct assignment of forces. Vectors $\{\underline{T}_1, \underline{\omega}_1, \underline{\Delta v}_1\}$ and $\{\underline{T}_2, \underline{\omega}_2, \underline{\Delta v}_2\}$ do not necessarily have to feature same number of elements. Grouping $\{\underline{T}_i, \underline{\omega}_i, \underline{\Delta v}_i\}$ $i \in \{1,2\}$ may

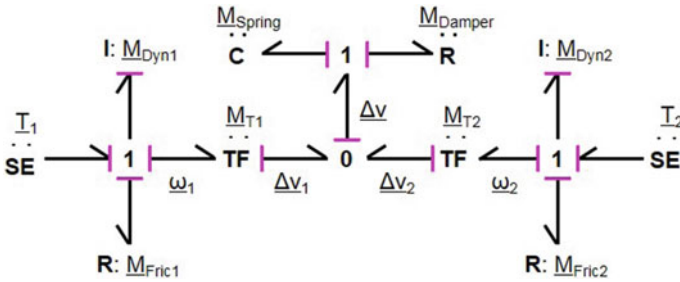


Fig. 2.7 Vectorial BG model inclusively of spring damper tooth contacts decomposed into two rotational vectorial branches

follow any reason. Anyway, TF parameter sign problems, as mentioned at the end of previous subsection, may not occur because unfolding keeps reference directions.

For complete scalar modelling all vectorial chains of structure “1 – TF – 0” have to be unfolded, i.e. parallelized operations have to be presented in detail. The number of such chains is equal to the number of elements in vector $\underline{\omega}$ —see Fig. 2.6b. In contrast the number of elements in vector $\underline{\Delta v}$ —see Fig. 2.7—determines the number of identical 0-nodes for junctions of three “1 – TF – 0” chains each with. Moreover, there are some additional 1-nodes necessary or usefully. Firstly, this particularly refers to shared translational speeds avoiding parallel TF-elements of same functionality and secondly this refers to more clarity in case of shared rotational speeds. These comments already hint at the problems of such scalar modelling, as getting very complex and rather complicated models. It should be applied exceptionally. Anyhow, as an advantage structure of the mechanical construction directly translates into structure of scalar BG. Further modelling details let these statements become still more important.

Vectorial modelling summarizes causal relations clearly without drawbacks. Any partial power is available via products of vector elements. In the following a specific example is needed in order to exemplify scalar modelling with consideration of spring damper tooth contacts. Suchlike general scalar models are impossible due to dependence on construction. Figure 2.8 introduces Renault IVT epicyclic four-speed gear as an example [22]. Table 2.3 specifies dynamic chains of effect and Table 2.4 summarizes basic equations for this example.

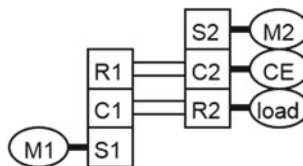


Fig. 2.8 Renault IVT—schematic reduction by symmetry [22]: CE—combustion engine, Cx—carrier, Rx—ring, Sx—sun, Mx—electric motor/generator, x = ∈{1, 2} gear set number resp. machine number

Table 2.3 Bidirectional chains of effect according Fig. 2.8

Gear set internal	Sun 1 \Leftrightarrow spring SP 1	\Leftrightarrow	Carrier 1/ring 2
		\Leftrightarrow	Planet 1
	Sun 2 \Leftrightarrow spring SP 2	\Leftrightarrow	Carrier 2/ring 1
		\Leftrightarrow	Planet 2
Between gear sets	Carrier 1/ring 2 \Leftrightarrow spring PR 1	\Leftrightarrow	Carrier 2/ring 1
		\Leftrightarrow	Planet 1
	Carrier 2/ring 1 \Leftrightarrow spring PR 2	\Leftrightarrow	Carrier 1/ring 2
		\Leftrightarrow	Planet 2

Table 2.4 Basic equations for Fig. 2.8, r_{yx} —radius, $y \in \{R, S\}$, P—planet, v —circumferential velocity; for more abbr. see Fig. 2.8

Willis equation $\frac{\omega_{Sx} - \omega_{Cx}}{\omega_{Rx} - \omega_{Cx}} = -\frac{r_{Rx}}{r_{Sx}}$ and example Renault IVT		$\omega_{R1} = \omega_{C2}$ $\omega_{R2} = \omega_{C1}$
Dynamics with equivalent system, cf. Sect. 2.2, planets implicitly	Dynamics with spring and damper, planets explicitly	
Reconfiguration for one basic planetary gear set ($\omega \cdot r = v$)		
$\omega_S \cdot r_S - \omega_C \cdot r_S = \omega_C \cdot r_R - \omega_R \cdot r_R$	$\omega_S \cdot r_S - \omega_C \cdot r_S = \omega_C \cdot r_R - \omega_R \cdot r_R = -\omega_P \cdot r_P$	
Renault IVT related definition for both sets, $x \in \{1, 2\}$		
$n_1 = -\frac{r_{R1}}{r_{S1}}; n_2 = 1 + \frac{r_{R2}}{r_{S2}}$ $\omega_{Sx} = (1 - n_x) \cdot \omega_{C1} + n_x \cdot \omega_{C2}$	$\omega_{Sx} \cdot r_{Sx} - \omega_{Cx} \cdot r_{Sx} + \omega_{Px} \cdot r_{Px} = 0 \equiv \Delta v_{SPx}$ $\Rightarrow \Delta v_{SPx} = v_{Sx} - v_{C1} + v_{Px}; i = 2x - 1$ $\omega_{Cx} \cdot r_{Rx} - \omega_{Rx} \cdot r_{Rx} + \omega_{Px} \cdot r_{Px} = 0 \equiv \Delta v_{PRx}$ $\Rightarrow \Delta v_{PRx} = v_{Cj} - v_{Rx} + v_{Px}; j = 2x$	

Tables 2.3 and 2.4 based on Fig. 2.8 enable it to draw scalar BG Fig. 2.9 if 6 scalar equations of rotational motion, in sense of (2.14), and 4 scalar translational equations of spring and damper motion, in sense of (2.15), are taken into account. In the latter case particular dynamic circumferential speeds Δv have to be integrated as well as spring forces F_S and damper forces F_D have to be added.

$$F_{SD} = F_S + F_D; F_D = d_{SD} \cdot \Delta v; F_S = K_{SD} \int \Delta v dt + 0 \quad (2.15)$$

Both four-speed gear constructions based on two standard planetary gear sets and bidirectional chains of effect according Table 2.3 are directly visible in Fig. 2.9. Equations in Table 2.4 are graphically presented just as well. Please note that nodes 1_{Z1} and 1_{R1} resp. 1_{Z2} and 1_{R2} could be merged but clearness would suffer. That applies also to a possible changing over of nodes 1_{M1} resp. 1_{M2} to direct links of nodes 1_{U1} and 0_{M1} resp. 1_{U2} and 0_{M2} via additional TF-elements with parameter r_{Px} .

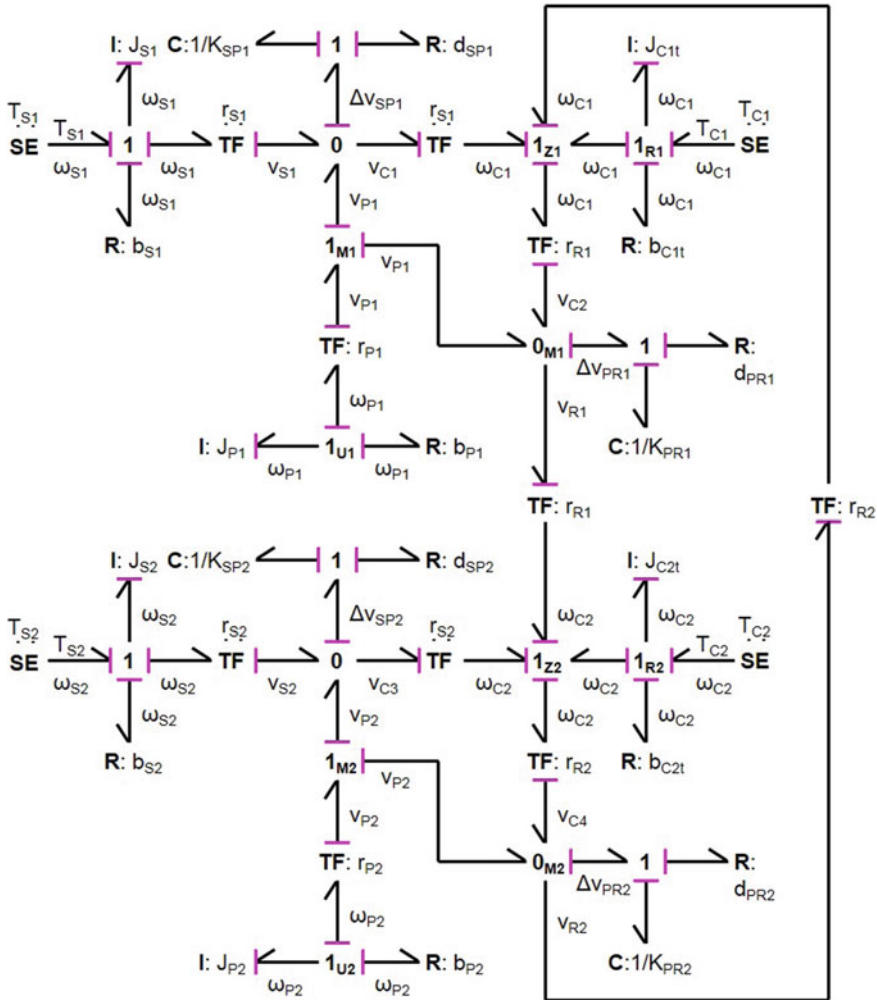


Fig. 2.9 Scalar BG model example Renault IVT (Fig. 2.8) inclusively of spring and damper tooth contacts; J_{Cxt} resp. b_{Cxt} —total inertia resp. total friction coefficient, K_{yx} resp. d_{yx} $y \in \{SP, PR\}$ —spring resp. damper coefficients, for more abbr. see Table 2.3

2.4 Simulation Hints and Examples of Numerical Results

For reason given in introduction, this subsection deals with Simulink[®] freeware add-on block library BG V2.1 [13–15]. In contrast with Simulink[®] standard Simscape libraries as Physical Modeling family which are component based libraries from the outset this add-on block set provides the user with real basic power flow elements. The next step only will define components based on subsystem technique if useful from the user’s perspective. Thus the user is free to develop and study diverse variants of

component models, to study inner component power flow, to choose scalar, vectorial or hybrid component configuration or to easily implement user in-depth knowledge for instance.

2.4.1 *Simulation Hints*

Moreover, since BG modelling method is a member of a specific power flow modelling tool family some parameter definitions may be applied to other members of this family such as Energetic Macroscopic Representation (EMR) or Power Flow Graph (POG). Again it has to be emphasized that a so-called zoomed view representation inside BG icons not at all may serve as an alternative to modelling via BG icons. At best this may support familiarization with BG method. However, POG is a real alternative. With this in mind one selected example will be given modeled by BG, POG and EMR in order to inspire the reader to get heavily involved with this modelling tool family. Furthermore, close relations between these tools also hint at fact that scalar, vectorial or mixed representation may be applied to all three methods [20, 32]. Add-on block library BG V2.1 for graphical programming of BG's by Simulink[®] shows some specific characteristics. Selected features are as follows.

- Bidirectional connection of blocks via visible forward direction and invisible backward direction. Whereat forward direction means propagation of a power variable from antecessor block to successor block. Thus this statement may be related to flow or effort depending on context. Port names are adapted to particular block settings.
- Unidirectional standard blocks must not be connected to BG block structure. This applies to all bidirectional power flow connections. Exceptions are related to unidirectional parameter inputs and to source BG blocks and measurement BG blocks which define interfaces to standard Simulink[®] blocks.
- Menu driven customization limits number of BG blocks to 9 types featuring buttons such as causality, optional power output, variable parameter input or scalar/vectorial function. Representative cases for bringing together modes of operation in one block type are nodes, sources and storages. Accidentally change of menu settings may be locked.
- Automatic detection of easy sources of error such as misconnection of flow and effort ports or menu mismatch of operation mode and internal parameter is implemented. All power variables are available, directly or via measurement nodes. About 25 examples partially with variants demonstrate use of blocks and library.

Graphical programming of BG Fig. 2.2b by means of add-on block library BG V2.1 demonstrates Fig. 2.10. Structural identity is evidently. Additional node blocks named “splitter” serve for making available power variables ω_3 and ω_4 . Please note this is not necessary regarding ω_1 and ω_2 because gauge blocks do not modify node power balances. All additional gauge blocks serve as interface outward to standard blocks whereas ports “S” of source blocks torque serve as interface inwards to BG

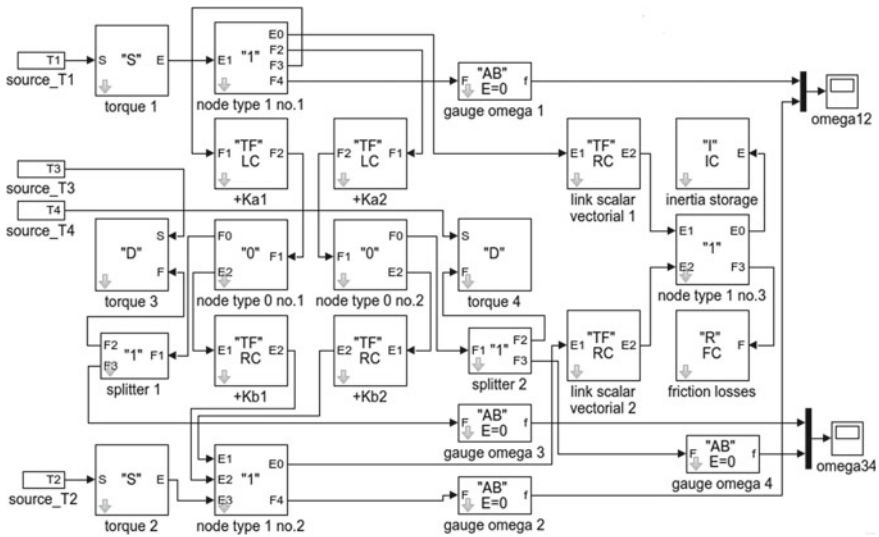


Fig. 2.10 Graphical programming of BG Fig. 2.2b by means of add-on block library BG V2.1

structure. Port names “E” and “F” mirror causality. For simplification Fig. 2.10 “node type 1 no. 3” combines operation of Fig. 2.2b nodes “1_e” and “1_r”. Table 2.5 summarizes block icons applied, explains preset customizations and gives selected possible menu driven transformations for block type in question.


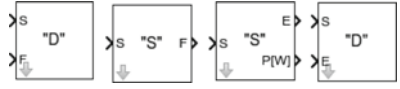










Vectorial BG modelling for this example illustrates Fig. 2.11a. Again structural identity of Figs. 2.5b and 2.11a is evidently. Because of vectorial operation two gauge blocks only are added, no splitter is necessary. Please note that vectorial operation may not be seen before any menu interaction aside from standard source and sink blocks.

Same issue model vectorial POG model Fig. 2.11b and vectorial EMR model Fig. 2.11c. All three modelling variants are graphically programmed based on same parameter definitions. Specific POG related characteristics are as follows. See [28, 29, 31] e.g. for more details.

- Working block and connection block represent the only two basic blocks. Interfaces, mixing points and connectors complete basic blocks. Working block appears in three different variants depending on losses and/or energy storage. Connection block visualizes transpose operation.
- All mathematical operations are directly visible. Power variables of type flow and effort may be swapped between upper and lower path. Tool is very good suitable for introduction to power flow modelling idea and for state space models based on energy matrix. Neither a block library nor an empty icon library is necessary.

Specific EMR related characteristics are as follows. See [1, 3, 17] e.g. for more details.

Table 2.5 Selected elements of Simulink® Bond graph library V.2.1 [13–15], (BG internal bond connections are symbolized via Simulink related forward connections either by effort E or flow F, opposite direction is invisibly and automatically established; optional outputs: P[W] power, M momentum, D displacement; optional inputs: NL non-linear parameter)

Icon used in examples	Generalized name (colour)	Operation just for preset customization	Subset menu driven customization variants
	Source/destination (orange)	Input unidirectional signal “s” and output as effort “E” of a bidirectional bond	
	Gauging, activated bond (pink)	Gauging flow F, unchanged power balance via E = 0	
	Node (yellow)	$E_0 = + E_1 - E_2 - E_3 - E_4$, $F_0 = F_1 = F_2 = F_3 = F_4$	
	Energy storage (cyan)	$F = \frac{1}{K} \int E dt + E_0$ IC: integral causality K: parameter	
	Losses (blue)	$E = K \cdot F$ FC: flow causality K: parameter	
	Transformer (magenta)	$F_2 = K \cdot F_1$ $E_1 = K \cdot E_2$ LC: left causality K: parameter	

- This tool makes use of an empty icon library. This means icons as used in Fig. 2.11c define empty subsystems of special relevance which have to be filled by the user. In easy case of Fig. 2.11 energy accumulation is just identically with POG working block and mono-physical converter is identically with POG connection block.
- Power variables of type flow and effort may be swapped between upper and lower path. Tool is very good suitable for visualization of power distribution and

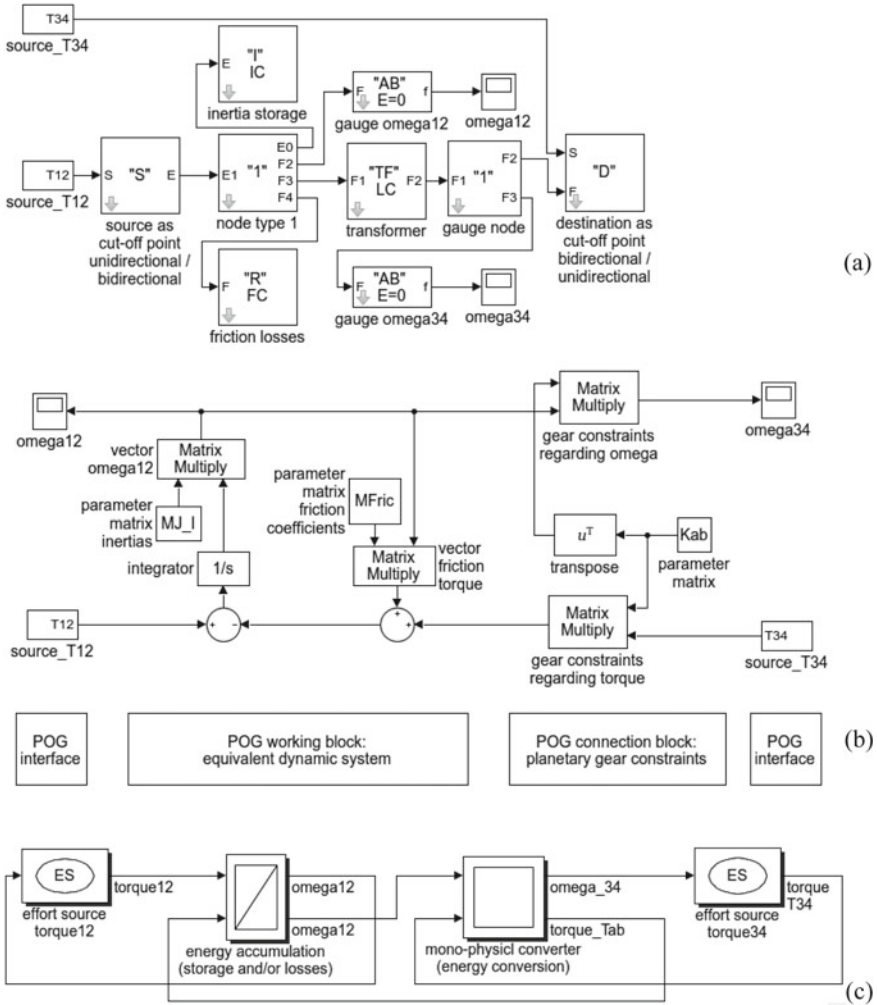


Fig. 2.11 Graphical programming of example Fig. 2.5b via BG V2.1 (a), POG (b) and EMR (c)

junction as well as identification of control structures which also addresses the inverse problem [8].

See e.g. [2, 13, 16, 30] for more examples regarding comparison of modeling tools BG, POG and EMR. There is no problem with addition of control structures for any of these tools. By way of example [19] expounds DC machine BG model inclusive of elastic shaft and parameterized state control structure.

2.4.2 Examples of Numerical Results

Table 2.6 gives parameters for simulation of models Fig. 2.10 and Fig. 2.11 following simple standard three-speed planetary gear [32]. In order to realize epicyclic four-speed gear simulation example Renault IVT was chosen [22]. For this purpose three-speed parameter set was applied twice and constraints were considered. Since simulation task is targeted only on illustration of same results for different model variants shaft related parameter determination is strongly simplified. As might be expected all four models output same time response Fig. 2.12a regarding ω_1 till ω_4 if torques Fig. 2.12b are applied. Because the equivalent dynamic system includes component cross coupling time behavior also can feature more or less distinct non-minimum-phase behavior—see ω_4 step response on T_2 step input. Any transmission gears, wheels or operation modes are not considered here.

Another example illustrates the influence of the choice of state variables, i.e. input and output variables, on BG parameters. Gear Fig. 2.13 shows a variant of Fig. 2.8. It is taken from [10], desists from detailed assignment of machines to gear shafts and features one common sun body and one common ring body. Thus there exists only one sun angular velocity and one ring angular velocity each with.

As already discussed in detail parameters n_1 and n_2 of TF matrix \underline{K}_{ab} given in (2.12) may be found via superposition principle or Willis equation, see Table 2.4. Table 2.7

Table 2.6 Parameter definitions for simulation

<i>Simple standard three-speed planetary gear—element parameters following [32]</i>				
	Sun (S)	Ring (R)	Carrier (C)	Planet (P)
Inertia (kg m ²)	0.6	2.5	1.8	0.08
Friction (Nms/rad)	0.15	6.5	2.8	0.24
Radius (m)	$r_S = 0.1$	$r_R = 0.25$	n.s.	n.s.
<i>Epicyclic four-speed gear Renault IVT [22]—simplified composed of two gear sets following [32]</i>				
Geometrically determined [11]	$n_1 = -\frac{r_{R1}}{r_{S1}} = -2.5$		$n_2 = 1 + \frac{r_{R2}}{r_{S2}} = +3.5$	
TF parameter (2.1d)	$K_{a1} = 2.5$	$K_{b1} = 3.5$	$K_{a2} = -3.5$	$K_{b2} = -2.5$
<i>Assignment/equation of motion—parameters (2.7), simplified specification</i>				
Index assignment	1: C1/R2	2: R1/C2	3: S1	4: S2
Friction (Nms/rad)	$K_{F1} = 9.3 = K_{F2}$		$K_{F3} = 1.11 = K_{F4}$	
Inertia (kg m ²)	$J_1 = 4.3 = J_2$		$J_3 = 0.92 = J_4$	
<i>Equivalent dynamic system—matrix elements (2.10), following parameters above</i>				
Friction matrix \underline{M}_F	$K_{F11} = 29.8$	$K_{F12} = -19.4 = K_{F21}$		$K_{F22} = 29.8$
Inertia matrix \underline{M}_d	$J_{11} = 21.3$	$J_{12} = -16.1 = J_{21}$		$J_{22} = 21.3$

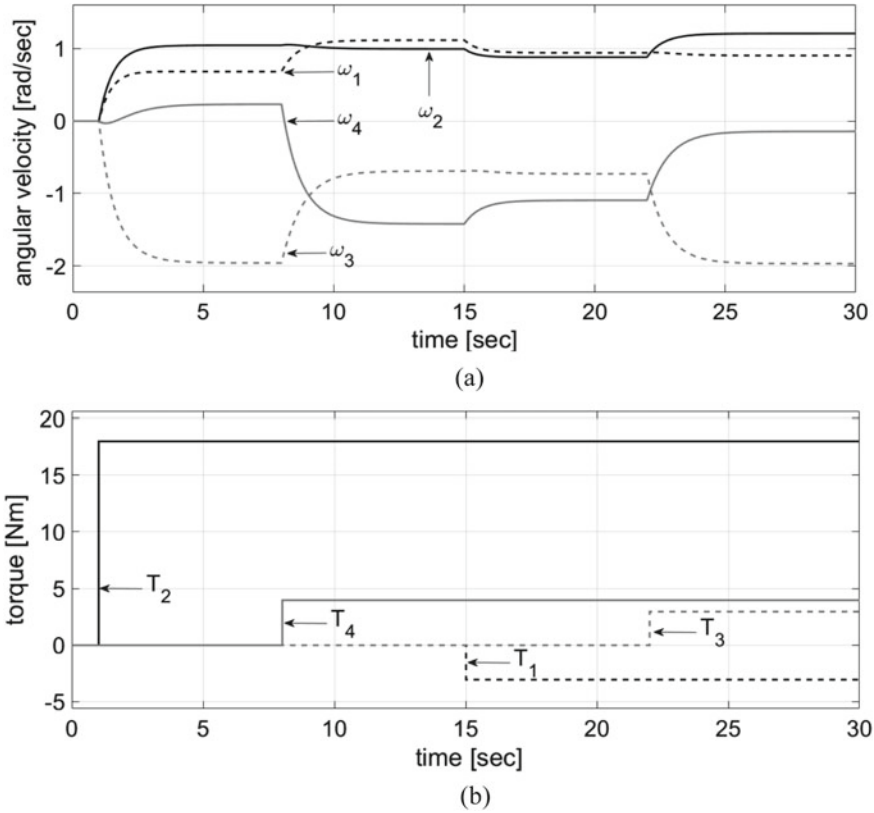


Fig. 2.12 Exemplary time response of models Figs. 2.10 and 2.11 angular speeds ω_1 till ω_4 (a) if torques are given (b) and parameters Table 2.6 are applied

summarizes suchlike definitions for gear Fig. 2.13 supplemented by numerical values based on parameter sets defined in [10]. However, depending on construction and application state variable choice may be different and Table 2.8 exemplarily presents two arbitrary selected alternatives. Since this choice is conform to [10], Table 2.9 provides proof that results obtained via BG modelling method in question do not differ from those results found out by means of the Linear-Graph or the Contour-Graph based models in [10]. Definitions in Table 2.7 do not allow assignment of gear in question to these two gear groups defined in [11]. In order to answer this issue it is necessary to modify definitions n_1 and n_2 in such a way that it meets variant 3, Table 9 in [11]. Table 2.10 summarizes results. New definitions $n_1 = 6$ and $n_2 = -20$ lead to $\alpha > 1$ and $\beta < 1$ and thus to assignment to group G2 in [11] just as Simpson gear or Renault IVT gear.

Care has to be taken for application of formulas [11] Table 9 if sequence of inputs and outputs doesn't meet desired sequence. As Table 2.10 illustrates state variable choice II in Table 2.8 yields a direct correspondence with variants [11] Table 9, but

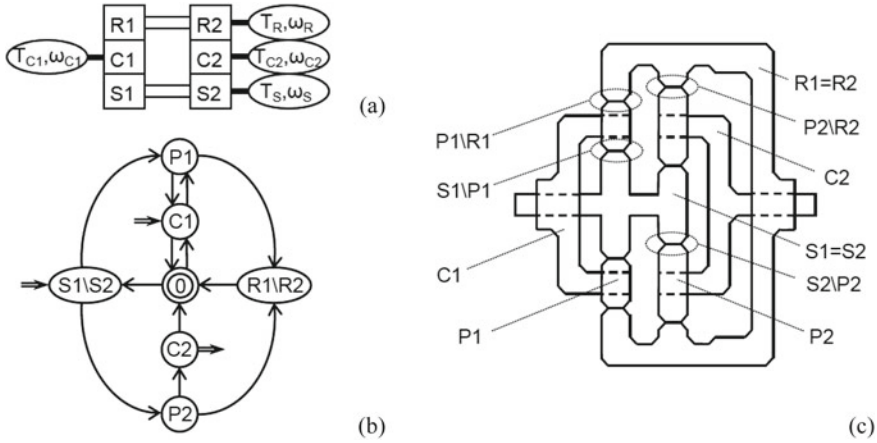


Fig. 2.13 Exemplary planetary gear following [10]; Cx—carrier, Px—planet, Rx—ring, Sx—sun; 0—support for the axes S, R and C; $x \in \{1, 2\}$ gear set number: **a** simplified sketch; **b** Contour Graph; **c** detailed drawing with common mechanical body (\equiv), tooth contact (\wedge) and planets

Table 2.7 Parameter definitions for gear Fig. 2.13; Z_{Sx}, Z_{Rx} $x \in \{1, 2\}$ teeth numbers of suns or rings assuming equivalence of radii ratios and teeth ratios

Equations for Fig. 2.13 directly based on superposition principle or Willis method, considering: $\omega_{S1} = \omega_{S2} = \omega_S;$ $\omega_{R1} = \omega_{R2} = \omega_R$	Choice of definitions (correlation with [11], Table 9, variant 6)	Parameter set 1, see [10] p. 419: $Z_{S1} = 15,$ $Z_{R1} = 63,$ $Z_{S2} = 18,$ $Z_{R2} = 60$	Parameter set 2, see [10] p. 425: $Z_{S1} = 18,$ $Z_{R1} = 60,$ $Z_{S2} = 15,$ $Z_{R2} = 63$
$\omega_{C1} = \frac{r_{S1}}{r_{S1}+r_{R1}} \omega_S + \frac{r_{R1}}{r_{S1}+r_{R1}} \omega_R$ $\omega_{C2} = \frac{r_{S2}}{r_{S2}+r_{R2}} \omega_S + \frac{r_{R2}}{r_{S2}+r_{R2}} \omega_R$	$n_1 = \frac{1}{1+\frac{r_{R1}}{r_{S1}}}; n_2 = \frac{1}{1+\frac{r_{R2}}{r_{S2}}}$ $\omega_{C1} = n_1 \omega_S + (1 - n_1) \omega_R$ $\omega_{C2} = n_2 \omega_S + (1 - n_2) \omega_R$	$n_1 = 0.1923$ $n_2 = 0.2308$	$n_1 = 0.2308$ $n_2 = 0.1923$

Table 2.8 Two more state variable variants for gear Fig. 2.13, based on Table 2.7

No.	Choice of state variables	Correlation	Parameter set 1	Parameter set 2
I	$\omega_{C1} = \frac{n_1-1}{n_2-1} \omega_{C2} + \frac{n_2-n_1}{n_2-1} \omega_S$ $\omega_R = \frac{1}{1-n_2} \omega_{C2} + \frac{n_2}{n_2-1} \omega_S$	With [11], Table 9, variant 5	$\omega_{C1} = +1.05 \cdot \omega_{C2}$ $-0.05 \cdot \omega_S$ $\omega_R = +1.30 \cdot \omega_{C2}$ $-0.30 \cdot \omega_S$	$\omega_{C1} = +0.952 \cdot \omega_{C2}$ $+0.048 \cdot \omega_S$ $\omega_R = +1.24 \cdot \omega_{C2}$ $-0.24 \cdot \omega_S$
II	$\omega_R = \frac{1}{1-n_1} \omega_{C1} + \frac{n_1}{n_1-1} \omega_S$ $\omega_{C2} = \frac{n_2-1}{n_1-1} \omega_{C1} + \frac{n_1-n_2}{n_1-1} \omega_S$	With [11], Table 9, variant 4	$\omega_R = +1.24 \cdot \omega_{C1}$ $-0.24 \cdot \omega_S$ $\omega_{C2} = +0.952 \cdot \omega_{C1}$ $+0.048 \cdot \omega_S$	$\omega_R = +1.30 \cdot \omega_{C1}$ $-0.30 \cdot \omega_S$ $\omega_{C2} = +1.05 \cdot \omega_{C1}$ $-0.05 \cdot \omega_S$

Table 2.9 Numerical values for operating points and ratios based on Tables 2.7 and 2.8, cp. [10]

Input	Parameter set 1		Parameter set 2	
	Output	Ratios	Output	Ratios
Operating point 1, [10] Table 1: $\frac{\omega_S}{\omega_{C2}} = \frac{157}{87.5}$	$\frac{\omega_R}{\omega_{C1}} = \frac{66.65}{84.03}$	$i_{SR_C2} = 2.36$ $i_{SC1_C2} = 1.87$	$\frac{\omega_R}{\omega_{C1}} = \frac{70.95}{90.81}$	$i_{SR_C2} = 2.21$ $i_{SC1_C2} = 1.73$
Operating point 2, [10] Table 2: $\frac{\omega_S}{\omega_{C1}} = \frac{157}{30}$	$\frac{\omega_R}{\omega_{C2}} = \frac{-0.238}{36.05}$	$i_{SR_C1} = -659.4$ $i_{SC2_C1} = +4.36$	$\frac{\omega_R}{\omega_{C2}} = \frac{-8.1}{23.65}$	$i_{SR_C1} = -19.38$ $i_{SC2_C1} = +6.64$

Table 2.10 Modified definitions n_1 and n_2 corresponding to Table 9 in [11] for α and β based gear grouping according [11], Tables 11 and 13

Correlation with [11], Table 9, variant 3 implies	New modified definitions follow and parameter set 2 gives	Relevant benchmarks to compare with gears in [11]
$\omega_{C1} = \frac{n_1}{n_1 - n_2} \omega_S + \frac{n_2}{n_2 - n_1} \omega_R$ $\omega_{C2} = \frac{n_1 - 1}{n_1 - n_2} \omega_S + \frac{n_2 - 1}{n_2 - n_1} \omega_R$	$n_1 = \frac{1 + \frac{r_{R2}}{r_{S2}}}{\frac{r_{R2}}{r_{S2}} - \frac{r_{R1}}{r_{S1}}}; n_2 = \frac{1 + \frac{r_{R2}}{r_{S2}}}{1 - \frac{r_{R2}}{r_{S2}} \frac{r_{S1}}{r_{R1}}}$ $= 6 \qquad \qquad \qquad = -20$	$\alpha = \frac{n_1(n_2 - 1)}{n_2(n_1 - 1)} = 1.26$ $\beta = \frac{n_2}{n_2 - 1} = 0.9524$
Choice of state variables according Table 2.8 now leads to modified formulas but same numerical values		Remarks to the correlation of “inputs” and “outputs” with [11], Table 9
$\omega_{C1} = \frac{n_2}{n_2 - 1} \omega_{C2} + \frac{1}{1 - n_2} \omega_S = 0.952 \cdot \omega_{C2} + 0.048 \cdot \omega_S$ $\omega_R = \frac{n_2 - n_1}{n_2 - 1} \omega_{C2} + \frac{n_1 - 1}{n_2 - 1} \omega_S = 1.24 \cdot \omega_{C2} - 0.24 \cdot \omega_S$		Variant 5 [11]—no direct correlation for “outputs”, thus second equation Table 9 in [11] determines first “output” and vice versa
$\omega_R = \frac{n_2 - n_1}{n_2} \omega_{C1} + \frac{n_1}{n_2} \omega_S = 1.30 \cdot \omega_{C1} - 0.30 \cdot \omega_S$ $\omega_{C2} = \frac{n_2 - 1}{n_2} \omega_{C1} + \frac{1}{n_2} \omega_S = 1.05 \cdot \omega_{C1} - 0.05 \cdot \omega_S$		Variant 1 [11]—direct correlation, thus direct application formulas Table 9 in [11] is possible

state variable choice I in Table 2.8 does not. In this case the order of outputs is reverse. Hence equation 1 gives output 2 and vice versa. In case of a reverse input sequence multiplier formulas [11] Table 9 for inputs have to be swapped. The reason is reduction of Table 9 in [11] to avoid variants with repetitions.

2.4.3 Discussion

The discussion of the numerical data in Tables 2.7, 2.8, 2.9 and 2.10 results in the following. The choice of state variables has a significant effect on the numerical

values of the weighting factors for the angular velocities. The numerical values can all be positive and relatively close to each other (Table 2.7), but the signs can also be different and a relatively large difference in the absolute value can occur (Table 2.8). The state variable selection can also lead to the special case that the reverse specification of the number of teeth leads to exactly reversed weighting factors (Table 2.7). However, the practical application of the gear, i.e. the assignment of the external elements, precludes a completely free choice of the state variables. As expected, all three methods, i.e. Linear Graph, Contour Graph and Bond Graph, lead to consistent results with respect to the ratios of angular velocities (Table 2.9). For the assignment of a certain gear to one of the groups $\{\alpha < 1, \beta > 1\}$ or $\{\alpha > 1, \beta < 1\}$, respectively, a modification of the definition of n_1 and n_2 may be necessary (Tables 2.7 and 2.10). The choice of state variables then leads to modified formulas but the numerical values of the weighting factors will remain definitely unchanged (Tables 2.8 and 2.10).

Finally it should be pointed out that standard efficiency is not significantly in case of dynamic systems. Therefore various suggestions were made based on integral criteria regarding input power P_{in} and output power P_{out} , e.g. (2.16) given in [19, 27]. Cyclic or approximately cyclic operation mode over time period T is an application precondition. Add-on library BG V2.1 supports such considerations via unlock button for power output if applicable, see icons energy storage or losses Table 2.5 column 4 customization variants. Thus also check of partial powers against input power(s) or illustration of internal storages operation mode get very easily possible [19]. Tools POG and EMR need extra power computation.

$$\varepsilon = \frac{\int_0^T P_{out}(t) dt}{\int_0^T P_{in}(t) dt} \quad (2.16)$$

Table 2.6 utilizes Renault IVT [22] and Tables 2.7, 2.8 and 2.9 utilize exemplary gear [10] as examples of epicyclic four-speed gears. In doing so one or three particular link-ages resulting from constraints are valid. This means that one or three of six possible linkages of indices and planetary gear components are considered. More linkages, i.e. n_1 and n_2 definitions Tables 2.6 and 2.7, more examples of epicyclic four-speed gears as well as general considerations for speed and power ratios gives [11]. Especially diagrams based on such ratios and generalized formulas provide with a valuable tool to analyze meaningful operation ranges and give hints for comparison of different gear constructions.

Models subsection 3 can be simulated simply by analogy to suggestions given in this subsection. Furthermore, intermediate step Coates Graph as a result of BG conversion and subsequently the implementation of Coates Formula would pave the way for transfer function generation if desired [24].

The use of these planetary gears in hybrid vehicles has a long technical history already. For example, a city bus was developed by Timken-Detroit in 1941, although it was still equipped with DC engines. The design used a mechanical double planetary

body still without using the rings. A Bond Graph model and a comparison with a standard planetary gear can be found in [18]. Apart from the admittedly meritorious but repeatedly cited Toyota Hybrid System, which is still based on only one planetary gear, there are now numerous hybrid vehicles introduced with planetary gearboxes similar to the design examined here. Some typical examples are the GM-Allison hybrid bus, the aforementioned Renault IVT or the Timken eVT. These planetary gearboxes also use brakes and clutches to enable structural switching for different modes of operation. Detailed studies of these examples can be found in [22]. The investigation here has been limited to one mode of operation in order to focus on the principle procedure of the proposed model building.

2.5 Conclusions

The paper emphasizes the Bond Graph method as power flow modelling tool of choice for dynamic systems in general and epicyclic four-speed gears in particular. This graph method allows not only for model development based on analysis but also for synthesis as applied here. It is very well suitable both for modelling of different abstraction levels and for implementation of any scalar, vectorial or mixed models.

It is clearly shown that the synthesis of generic power split models for epicyclic four-speed gears is not only possible, but also beneficial for understanding the subject of modelling. The physical background of the interaction of physical quantities by branching and joining can thus be plausibly led step by step to a model creation. The basic structure of the model can be found safely. In addition, causality plays an important role in avoiding model building errors. The synthesis also shows that, depending on the level of detail of the models, different technical realizations can lead to the same structure and thus this structure can be reused for a whole class of comparable physical tasks.

Based on a specific example model synthesis is explained step by step. Starting with static relationships definition of dynamic torques is introduced. Constraints lead to a general equivalent dynamic system. Modelling variants are discussed. Folding operation is highlighted as natural graphical BG tool to get vectorial models. Spring and damper elements serve as starting point for more detailed modelling.

Since Bond Graphs are a member of a power flow modelling tool family two more tools of this family are briefly presented. As it has to be expected simulation results show completely equivalent time characteristics for all three methods. All examples meet integral causality requirement. Although each tool possesses specific interesting features Bond Graphs provide with most flexibility and possibilities such as optional power outputs in order to check partial powers and to demonstrate internal storage operation modes or to study causality.

In this study, the synthesis of generic power split modelling for epicyclic four-speed gears based on two basic planetary gear sets has been clearly specified by taking into account variants, simulation examples and numerical examples. Thus, for example, it may be interesting for future work to investigate more sophisticated

structures based on three or more basic planetary gear sets. Suchlike design might benefit from analogies, maybe as epicyclic five-speed gear.

References

1. Barre, P.J., et al.: Inversion-based control of electromechanical systems using causal graphical descriptions. In: Proceedings of the 32th Annual Conference of the Industrial Electronics Society IEEE-IECON'2006, Paris, pp. 5276–5281 (2006)
2. Bouscayrol, A., et al.: Different energetic descriptions for electromechanical systems. In: Proceedings of the European Power Electronics Conference EPE'05, Dresden, 10 pp (2005)
3. Bouscayrol, A.: EMR Website—Energetic Macroscopic Representation and Summer School Contacts. <http://www.emrwebsite.org> (05.02.2020)
4. Breedveld, P.C.: Decomposition of multiport elements in a revised multibond graph notation. *J. Franklin Inst.* **318**, 253–273 (1984)
5. Brown, F.T.: *Engineering System Dynamics—A Unified Graph-Centered Approach*, 2nd edn. Taylor & Francis (2007)
6. Chau, K.T.: *Electric Vehicle Machines and Drives—Design, Analysis and Application*. Wiley, USA (2015)
7. Chen, K., et al.: Global modeling of different vehicles. *IEEE Veh. Technol. Mag.* **4**, 80–89 (2009)
8. Drewniak, J., Kopec, J., Zawiślak, St.: Kinematic and efficiency analysis of planetary gear trains by means of various graph-based approaches. In: Goldfarb, V., Barmina, N. (eds.) *Theory and Practice of Gearing and Transmission* (Springer series 34: Mechanism and Machine Science), pp. 263–284 (2016)
9. Drewniak, J., Kopec, J., Zawiślak, St.: Graph models of automobile gears—kinematics. *Int. J. Appl. Mech. Eng.* **19**, 563–573 (2014)
10. Drewniak, J., Zawiślak, St.: Linear-graph and contour-graph-based models of planetary gears. *J. Theor. Appl. Mech.* **48**, 415–433 (2010)
11. Geitner, G.-H., Komurgoz, G.: Generic power split modelling for compound epicyclic four-speed gears. *Mech. Mach. Theory* **116C**, 50–68 (2017)
12. Geitner, G.-H., Kömürköz, G.: Generalised power flow model for electric machines. In: Proceedings of the 42th Annual Conference of the Industrial Electronics Society IEEE-IECON'2016, Firenze, pp. 1662–1669 (2016)
13. Geitner, G.-H., Kömürköz, G.: *Power Flow Modelling of Dynamic Systems—Introduction to Modern Teaching Tools* (2015). Available via Cornell University Library. <https://arxiv.org/abs/1505.06828>
14. Geitner, G.-H.: Add-on library BG V.2.1 Homepage—Graphical Programming of Bond Graphs by Means of Simulink. https://etiema.et.tu-dresden.de/ae2_files/ae_8_1e.htm (05.02.2020)
15. Geitner, G.-H.: Bond Graph Add-on Block Library BG V.2.1. MATLAB central file exchange. <https://de.mathworks.com/matlabcentral/fileexchange/11092-bond-graph-add-on-block-library-bg-v-2-1> (05.02.2020)
16. Grossi, F., Lhomme, W., Zanasi, R., Bouscayrol, A.: Modelling and control of a vehicle with tire-road interaction using energy-based techniques. In: Proceedings of the Vehicular Power and Propulsion conference IEEE-VPPC'09, Dearborn, pp. 1842–1848 (2009)
17. Horrein, L., Bouscayrol, A., Cheng, Y., El Fassi, M.: Dynamical and quasi-static multi-physical models of a diesel internal combustion engine using energetic macroscopic representation. *Energy Convers. Manag.* **46**, 280–291 (2015)
18. Kömürköz, G., Geitner, G.-H.: Unified power flow based modelling of power split for Hybrid Electric Vehicles assuming three connections. In: Proceedings of the International Conference on Consumer Electronics, Communications and Networks IEEE-CECNet2011, XianNing, pp. 5464–5469 (2011)

19. K m rg z, G., Geitner, G.-H.: Power flow oriented modelling for teaching modelling of dynamical systems. *Int. J. Knowl. Sci. Technol.* **1**, 15–21 (2010)
20. LHomme, W., et al.: Energy savings of a hybrid truck using a Ravigneaux gear train. *IEEE Trans. Veh. Technol.* **66**, 8682–8692 (2017)
21. Mayet, C., et al.: Comparison of different models and simulation approaches for the energetic study of a subway. *IEEE Trans. Veh. Technol.* **63**, 556–565 (2014)
22. Miller, J.M.: Hybrid electric vehicle propulsion system architectures of the e-CVT Type. *IEEE Trans. Power Electron.* **21**, 756–767 (2006)
23. Mukherjee, A., Karmakar, R.: *Modelling and Simulation of Engineering Systems through Bondgraphs*. Alpha Science International Ltd. (2000)
24. Orlikowski, C.: Symbolic analysis of bond graphs by application of the coates rule. *Mech. Mach. Theory* **30**, 1019–1026 (1995)
25. Pennestri, E., Freudenstein, F.: A systematic approach to power-flow and static-force analysis in epicyclic spur-gear trains. *J. Mech. Des.* **115**, 639–644 (1993)
26. Romero, G., et al.: Kinematic analysis of mechanism by using bond-graph language. In: *Proceedings of the European Conference on Modelling and Simulation ECMS2006, Bonn*, 11 pp (2006)
27. Sch nfeld, R., et al.: *Assessment of quality of motion systems and controlled sequences of motion (DIN—VDI/VDE 3547)*. VDE publishers (2003)
28. Zanasi, R., Grossi, F.: The power-oriented graphs for modeling mechanical systems with time-varying inertia. In: *Proceedings of the Vehicular Power and Propulsion conference IEEE-VPPC'14, Coimbra*, 6 pp (2014)
29. Zanasi, R., Grossi, F., Giuliani, N.: Extended and reduced POG dynamic model of an automatic corking machine for threaded plastic caps. *Mechatronics* **24**, 1–11 (2014)
30. Zanasi, R., Grossi, F.: Differences and common aspects of POG and EMR energy-based graphical techniques. In: *Proceedings of the Vehicular Power and Propulsion conference IEEE-VPPC'11, Chicago*, 6 pp (2011)
31. Zanasi, R., Grossi, F.: Modeling and control of power-split hybrid electric vehicles. In: *Proceedings of the Vehicular Power and Propulsion conference IEEE-VPPC'10, Lille*, 6 pp (2010)
32. Zanasi, R., Grossi, F.: The POG technique for modeling planetary gears and hybrid automotive systems. In: *Proceedings of the Vehicular Power and Propulsion conference IEEE-VPPC'09, Dearborn*, pp. 1301–1307 (2009)
33. Zawi slak, S., Rysi nski, J. (eds.): *Graph-Based Modeling in Engineering (Series 42: Mechanisms and Machine Science)*. Springer, Cham (2017)

Chapter 3

Chosen Aspects of Analysis and Synthesis of Coupled and Complex Planetary Gears via Search and Contour Graphs Modelling



A. Deptuła, J. Drewniak, J. Kopeć, and S. Zawiślak

Abstract The paper presents kinematic analysis of a coupled and planetary gear with double bevel planetary wheels. The method is based on the concept of contour graphs assignment. Exemplary 1-dof and 2-dof epicyclic gear trains, and a complex planetary gear train are used to illustrate the application of this method. The well-known Willis' formulas were utilized for comparisons. When designing planetary gears, it is difficult to determine all the parameters at the beginning of the design, in particular some parameters are selected on the basis of preliminary calculations. Graph methods allow to create a model of the analyzed gear, then determine the detailed ratios and algorithmically determine other quantities and their parameters. Isomorphism identification in the synthesis process and enumeration of 1- dof epicyclic gear train graphs can be found in cited bibliography. The signal flow graphs were utilized for choice of acceptable intervals for number of teeth. All together gives a spectrum of graph application for investigation of complex and coupled planetary gears.

Keywords Kinematical analysis · Algorithmic approach · Teeth numbers · Ratio · Holistic approach

A. Deptuła

Faculty of Production Engineering and Logistics, Opole University of Technology, 45-316 Opole, Poland

e-mail: a.deptula@po.edu.pl

J. Drewniak · J. Kopeć · S. Zawiślak (✉)

Faculty of Machine Building and Computer Science, University of Bielsko-Biala, 43-300 Bielsko-Biała, Poland

e-mail: szawislak@ath.bielsko.pl

J. Drewniak

e-mail: jdrewniak@ath.bielsko.pl

J. Kopeć

e-mail: jkopec@ath.bielsko.pl

© Springer Nature Switzerland AG 2022

S. Zawiślak and J. Rysiński (eds.), *Graph-Based Modelling in Science, Technology and Art*, Mechanisms and Machine Science 107,

https://doi.org/10.1007/978-3-030-76787-7_3

3.1 Introduction

Solving contemporary complex technical problems requires integral design [13] and scientific foundations of all related areas. Cooperation between the designer and representatives of various scientific fields of investigations becomes a necessity in creation of appropriate models to the given design tasks.

In design methodology, modeling can be done in many ways. One can, for example, talk about physical models built in accordance with the theory of similarity. The mathematical models describing static and dynamic or determining the importance of structural and/or operational parameters have two main features: structure and parameters. Then one can determine a certain measure of agreement between the hypothetical (theoretical) relation/behavior and the empirical relation/behavior of such a model. Phase of verification is always needed [18, 22].

Among the tools supporting the decision in designing, the following notions can distinguish e.g.: decision trees, dendrites, tree classifiers, graph grammars and graphs [5]. The set of system particles or parts or elements (and the relationship between them) is represented graphically based upon the mathematical model. These activities are the main framework for implementation of the decision-making process which allows for making the adequate decisions to solve the considered problem [6–12].

The evolution in design aided systems has resulted in the use of solutions based on artificial intelligence and machine learning, as well [26, 36]. In particular, [38] describes the transformation of knowledge related to gears from the field of mechanics to the field of graph theory. The transformation of knowledge involves, among others, the expression of facts, relations, laws, etc., using graph concepts, i.e. facts or equations expressed by means of algebraic structures associated with a graph being a gear model.

Methods of graphs and structural numbers have been known in mechanics since 70-ties of the previous century [2, 19, 30]. Many practical applications of graph theory have been published for the following tasks: system dynamics tests [3, 37], analysis [14, 15, 23] and synthesis [4, 16, 17, 27, 28] of complex mechanical systems etc. In graph methods, such graph classes are considered - like e.g.: linear graphs, mixed graphs, polar graphs, flow graphs [32], hybrid graphs, bond graphs [1, 25, 37] as well as hypergraphs [20] and structural numbers [31, 33, 38]. Additionally, via graph approach, such problem as determining gear efficiency [24] was solved.

The reviews of an application of graphs in modelling of gears have been published: a short one [35], the widest [34] enclosing extremely wide lists of papers, giving a comprehensive overview with many detailed examples as well as [38]—the dissertation printed by the university printing office.

In the chapter, we would like to give practical examples how versatile graphs can be utilized for modelling and solving practical design problems related to design of planetary gears. The rough methodology is described related to the task of teeth numbers assessment and calculations of the ratios via traditional approach, as well. In some other problems, the reader has to relay on the cited bibliography.

3.2 Graph Theory Methods in the Analysis of Planetary Gears

Gears are one of the most commonly used part of machines—used for transfer power and rotational motion. They are widely used in many types of transport, including in aviation, where they are very heavily loaded. Destruction or misuse of transmission-related components may result in an aviation accident. Therefore, particularly high requirements must be met regarding their size, capability, reliability and durability [22, 23].

The most widely used graph representation for geared systems is a linear graph model [2, 21] in which vertices represent parts and edges represent joints or kinematical pairs (in general). This linear graph representation of geared system has been used extensively for modelling of kinematics of gear trains and in the design and synthesis of gear trains [34, 35, 38, 39]. In particular, there are many papers describing the use of graph methods in the analysis of planetary gears [14, 15, 19–21].

In normal gears as well as planetary gears, it is not possible to determine all parameters at the beginning of designing process, some values are selected on the basis of the results of preliminary calculations. It is unavoidable to use computer programs (in particular those based on graph methods) to solve such tasks as: (i) pre-determine the number of gear stages, (ii) required gear ratios, (iii) velocities and (iv) the number of wheel teeth – taking into account some initial assumptions such as maximum dimensions or weight of the gear [17, 25–27]. There are already author's and co-author's papers (for example [9, 10]) in which the most important construction parameters (e.g. the optimal number of teeth) have been determined using decision trees and graph search algorithms. The present chapter presents—in details—the use of modified search or dependency graphs to determine the allowable intervals for number of teeth. The second issue described in the present chapter is related to utilization of contour graphs which are used almost solely by the authors of this chapter—therefore their publications are cited for this topic.

3.2.1 Contour Graph Model of the Planetary Gear

Planetary gears, unlike ordinary gears, are characterized by the fact that the centers of certain wheels, hereinafter referred to as satellites, always move smoothly along circular trajectories around the geometric axis. The gear wheels, the centers of which are placed precisely in the gear axis, are called sun wheels, while the main part on which the satellites are mounted is called an arm. Figure 3.1 shows exemplary planetary gear unit.

Graph in the sense of graph theory is immanently associated with many algebraic structures such as matrices, matroids, structural numbers, linear spaces of cuts and

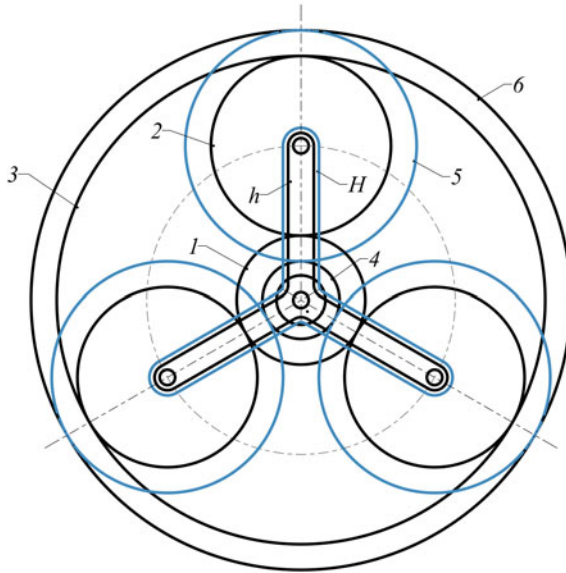


Fig. 3.1 Schematic front view of the considered planetary gear which simplified scheme is given underneath, 1–4—sun wheel with two toothings; h , H —carriers; 2,5—planetary wheels, 3–6 ring gear with two toothings

cycles, and many others [5, 38]. These objects enable encoding of the gear structure, which in turn allows to utilize the advanced artificial intelligence algorithms: evolutionary, ant [36] or immunological etc.

In the present chapter, the analysis of planetary gears is performed by means of the contour graph method which is rarely used. The analyzed artifact had been analyzed by authors in [14], but here the approach is revised and in general deeper explained. Moreover, the new figures were added. Furthermore, the contour-graph approach is worth popularizing due to its algorithmic nature. The idea of this method is based on distinguishing a number of subsequent rigid links of mechanisms that form a closed loop. Such a loop is called a contour. All independent contours should be taken into account. The comprehensive and handy Marghitu's book [29] concerns—among other—contour graphs and their application. The general algorithm for gear modeling with graphs can be described as follows:

- *Step 1.* Selecting a problem for kinematic analysis or synthesis and considering elements, rules, laws related to the selected problem (abstracting),
- *Step 2.* Determining the relationships for the specified elements (e.g. listing kinematic rotational pairs and meshings), after discretization and temporary neglecting other aspects (which are taken into account in virtual and laboratory analyses of a prototype),
- *Step 3.* Listing selected subgraphs based on the basic transmission graph and saving the codes,

- *Step 4.* Generating of equations describing the gearing on the basis of codes—thus in an orderly (algorithmic) way and solution of the obtained system of equations.

Another but similar general approach to modelling is described in [31]. Figure 3.2 shows an exemplary planetary gear with 2-dof (DOF) in the form of a functional diagram and its contour graph. The following notation rules were assumed: 1, 2, ..., 6 sun, planetary and rings wheels with internal teeth [14]—see Fig. 3.2, underneath. This is the simplest coupled planetary gear analyzed several times. Here it is used to present contour graph methodology as well as way of analysis of the 2-dof system.

The method of contour graphs modelling is described widely in [29] by Dan Marghitu, but also in [15, 16]. Here, we develop the gear discussed by us in [14]. The method is referred to as “contour graph approach”. This method is totally different in terms and rules for cuts and cycles comparing to the classical graph theory. Moreover,

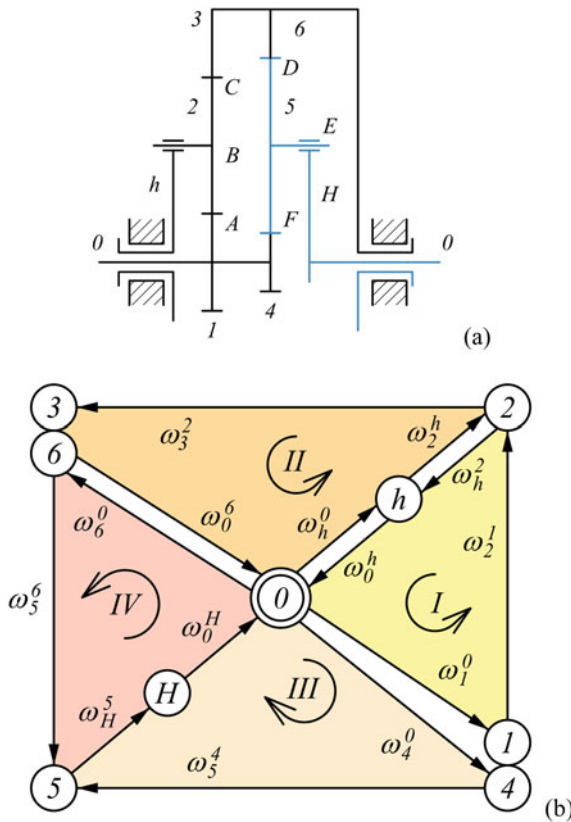


Fig. 3.2 Exemplary planetary gear: **a** functional diagram (simplified), where: 1, 2, ..., 6 represent sun wheels, planetary wheels and ring wheels; *h*, *H* arms (carriers)—as above; *A*, *B*, ... *F* characteristic points; ω —rotational speeds of adequate parts of the gear unit; **b** contour graph where Roman numerals are utilized for description of the consecutive contours

the further essential differences is that directed graph is considered, and orientation of every contour is considered, as well as the relative quantities are utilized. In the result, we define the contour as the cycle in digraph, where additionally orientation is considered and the discussed contour has an essential influence on the form of the generated equations. The rules for creating a contour are as follows: a single contour begins in the support link of the considered system and it passes through the next rotating gear element, each of which transmits the rotational movement to the next one. The contour is considered closed when we reached the support system in the course of analysis—so it is represented by the close loop of mutually connected and movable elements, where the exception is the support (base, waypoint), which if fixed to the ground (i.e. reference frame/point). In order to create a full graph, all contours should be analyzed. For the graph from Fig. 3.2, the contour codes are as follows: **I**: $0 \rightarrow 1 \rightarrow 2 \rightarrow h \rightarrow 0$; **II**: $0 \rightarrow h \rightarrow 2 \rightarrow 3 \rightarrow 0$; **III**: $0 \rightarrow 4 \rightarrow 5 \rightarrow H \rightarrow 0$, **IV**: $0 \rightarrow 6 \rightarrow 5 \rightarrow H \rightarrow 0$. Every contour allows for derivation of two vector Eqs. (3.1).

Based on the above listed contours (shown in Fig. 3.2) it is possible to generate a system of Eqs. (3.1):

$$\left\{ \begin{array}{l} \omega_1^0 + \omega_2^1 + \omega_h^2 + \omega_0^h = 0 \\ \mathbf{r}_{OA} \times \omega_2^1 + \mathbf{r}_{OB} \times \omega_h^2 = 0 \\ \omega_h^0 + \omega_2^h + \omega_3^2 + \omega_0^6 = 0 \\ \mathbf{r}_{OB} \times \omega_2^h + \mathbf{r}_{OC} \times \omega_3^2 = 0 \\ \omega_4^0 + \omega_5^4 + \omega_H^5 + \omega_0^H = 0 \\ \mathbf{r}_{OF} \times \omega_4^5 + \mathbf{r}_{OE} \times \omega_H^5 = 0 \\ \omega_6^0 + \omega_5^6 + \omega_H^5 + \omega_0^H = 0 \\ \mathbf{r}_{OD} \times \omega_5^6 + \mathbf{r}_{OE} \times \omega_H^5 = 0 \end{array} \right. \quad (3.1)$$

$$\left\{ \begin{array}{l} \omega_1^0 + \omega_2^1 + \omega_h^2 + \omega_0^h = 0 \\ r_1 \cdot \omega_2^1 + (r_1 + r_2) \cdot \omega_h^2 = 0 \\ \omega_h^0 + \omega_2^h + \omega_3^2 + \omega_0^6 = 0 \\ (r_1 + r_2) \cdot \omega_2^h + (r_1 + 2 \cdot r_2) \cdot \omega_3^2 = 0 \\ \omega_4^0 + \omega_5^4 + \omega_H^5 + \omega_0^H = 0 \\ r_4 \cdot \omega_4^5 + (r_4 + r_5) \cdot \omega_H^5 = 0 \\ \omega_6^0 + \omega_5^6 + \omega_H^5 + \omega_0^H = 0 \\ (r_4 + 2 \cdot r_5) \cdot \omega_5^6 + (r_4 + r_5) \cdot \omega_H^5 = 0 \end{array} \right. \quad (3.2)$$

where: \vec{r}_i —are positional vectors in respect to the adequate points $k = A, B, \dots, F$.

$r_1 = r_{0A}$, r_2 —pitch radius of wheel 2, $r_4 = r_{0F}$, r_5 —pitch radius of wheel 5, $r_{0D} = r_4 + 2r_5$, $r_{0C} = r_1 + 2r_2$.

We are using here relative rotational velocities.

Each contour generates equations that refer to rotational speeds. Additional rules which are applied:

Table 3.1 Set of data for the analyzed compound gear (gear ratios for $\omega_h = 0$)

No.	Chosen sets of teeth numbers						Gear ratio
	z_1	z_2	z_3	z_4	z_5	z_6	i
1	18	36	-90	22	38	-98	50
2	22	29	-80	18	27	-72	-50
3	26	37	-100	28	35	-98	50
4	28	26	-80	22	23	-68	-50

- refer to changing the order of indicators,
- are related to the reference system,
- refer to geometrical relationships.

The system of Eq. (3.2) is obtained from (3.1) via changing vectors into scalar notions using the geometrical properties e.g. angle 90° between r and ω , which is characteristic for cylindrical wheels. The considered ideas are widely described in [14, 15]. We always obtain the solvable systems of linear equations, if the contours are independent. The most interesting characteristics of the considered problem is: that upon two input velocities, there is a possibility of derivation of two output ones. It is a essential feature of contour graph modelling. The correctness of the methodology originates from its derivation from the mechanism and machine science basics. Furthermore, it was confirmed several times that Hsu's, Marghitu's or Willis' approaches give the same results in case when they can be utilized.

Tables 3.1 and 3.2 show the results of analyses of possible teeth sets. In Table 3.1, six different sets fulfilling the assumed ratio are shown. It is fruitful and reasonable, if an engineer would have sets of realizable variants for further considerations. Such an approach can show more options than one can create based only upon own design experience or via trial and error method. The final set can be chosen performing adequate calculations as well as virtual or laboratory investigations. It is worth to mention, that the every obtained set of teeth assures fully precisely the gear ratio. It is equal exactly to the same assumed absolute value i.e. 50. These aspects were not analyzed in source paper [14].

The results obtained using classical and graph-based methods are the same but the graph methodology of gear analysis has the following advantage: (a) it is algorithmic; (b) equations can be generated via computer program; (c) it is additive i.e. analyzing more complicated gear variants e.g. adding new stages in parallel manner, former (initial) graphs can be utilized and just upgraded.

3.3 Kinematic Analysis of the Coupled and Planetary Gear with Double Bevel Planetary Wheels

This subchapter presents the kinematics of a series-parallel transmission, i.e. a coupled planetary gear (Fig. 3.3) which includes conical teeth. Then the analysis

Table 3.2 Angular velocities ω_H and $\omega_3 = \omega_6$ for arbitrary chosen $\omega_1 = 157$ rad/s and ω_h

ω_h	Number of wheel set							
	1		2		3		4	
	ω_H	$\omega_3 = \omega_6$	ω_H	$\omega_3 = \omega_6$	ω_H	$\omega_3 = \omega_6$	ω_H	$\omega_3 = \omega_6$
157	157	157	157	157	157	157	157	157
100	101.14	88.6	98.86	84.32	52.14	85.18	98.86	80.05
40.70	43.03	17.44	38.38	8.72	43.03	10.47	38.38	0
33.86	36.32	9.23	31.4	0	36.32	1.85	31.40	-9.23
32.40	34.89	7.48	29.90	-1.87	34.89	0	29.90	-11.21
26.17	28.78	0	23.55	-9.81	28.78	-7.85	23.55	-19.62
25	27.64	-1.4	22.36	-11.3	27.64	-9.32	22.36	-21.2
10	12.94	-19.4	7.06	-30.4	12.94	-28.22	7.06	-41.45
0	3.14	-31.4	-3.14	-43.17	3.14	-40.82	-3.14	-54.95
-10	-6.66	43.4	-13.34	-55.92	-6.66	-53.42	-13.34	-68.45
-25	-21.36	-61.4	-28.64	-75.05	-21.36	-72.32	-28.64	-88.7
-50	-45.86	-91.4	-54.14	-106.92	-45.86	-103.82	-54.14	-122.45
-100	-94.9	-151.4	-105.14	-170.67	-94.86	-166.82	-105.14	-189.95
-157	-150.72	-219.8	-163.28	-243.35	-150.72	-238.64	-163.28	-266.9

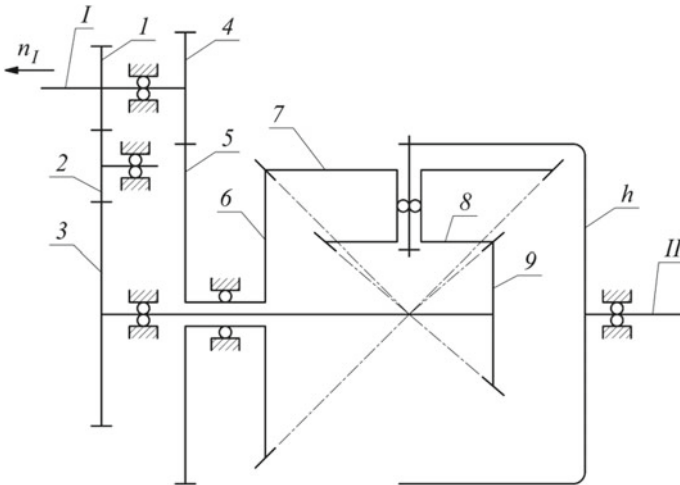


Fig. 3.3 Coupled and planetary gear with double bevel planetary wheels, where 1–4 wheels, 2 wheel, 3–9—central wheel, 5–6–7–8 toothed element; h—carrier

will be carried out using contour graph methods for the planetary gearbox modelling which is very rare but effective. Gear elements are described the same as above.

The degree of mobility of the gear being analyzed is:

$$W = 3 \cdot n - 2 \cdot p_5 - p_4 = 3 \cdot 6 - 2 \cdot 6 - 5 = 1, \quad (3.3)$$

where: $n = 6$ —number of moving elements, $p_5 = 6$ —number of class 5 pairs (rolling bearings), $p_4 = 6$ —number of 4th class pairs (meshing).

The value of the degree of mobility $W = 1$ means that for the kinematics of each transmission element, only the value of the rotational speed of one of the two shafts (input I associated with wheels 1 and 4 or output II associated with arm h) should be known. In this case, the given value of rotational speed n_I of input shaft I was adopted (Fig. 3.3).

3.3.1 Gear Ratio - Using Willis Formula

According to the definition, the kinematic ratio of a helical-bevel gear is:

$$i_{I,II} = \left(\frac{n_I}{n_{II}} \right) = \left(\frac{\omega_I}{\omega_h} \right) \quad (3.4)$$

where: $n_I = n_1 = n_3 = 750$ rpm- given rotational speed of input shaft I with gears 1 and 3;

$$\omega_I = \omega_1 = \omega_3 = \pi \cdot n_I / 30 = 25 \cdot \pi \text{ rad/s}$$

$n_{II} = n_h$ —sought rotational speed of the output shaft with arm h .

The kinematic ratio of the bevel planetary gear (Fig. 3.3) can be determined using the Willis formula. For this purpose, it is given a rotational speed— n_h and then determines the gear base ratio $i_{6,9}^h$:

$$i_{6,9}^h = \left(\frac{\omega_6^h}{\omega_9^h} \right) \quad (3.5)$$

$$i_{6,9}^h = \frac{\omega_6 - \omega_h}{\omega_9 - \omega_h} \quad (3.6)$$

where: $i_{6,9}^h$ —base kinematic ratio of the bevel gear from sun gear 6 to sun gear 9 (through planet gears 7–8), i.e. determined for relative angular velocity of wheels 6 and 9 with respect to the arm h ;

$\omega_j^h = \omega_j - \omega_h$ —relative angular speeds of the bevel planetary gears with the numbers: $j = 6, 7 - 8, 9..$

The kinematics of the planetary gearbox considered for the relative speeds of the wheels 6, 7–8 and 9 relative to the arm h are identical to the kinematics of the fixed axis bevel gear. So the base kinematic ratio is equal to the ratio of the fixed axis gears and can be determined from the known formula:

$$i_{6,9}^h = \left(-\frac{z_7}{z_6}\right) \cdot \left(\frac{z_9}{z_8}\right) = -\frac{z_7 \cdot z_9}{z_6 \cdot z_8} \quad (3.7)$$

where it was assumed that the ratio of the bevel wheels converging at the meshing point is positive (e.g. pair of wheels 8 and 9) and negative for diverging (pair of wheels 6 and 7). Finally received:

$$\frac{\omega_6 - \omega_h}{\omega_9 - \omega_h} = -\frac{z_7 \cdot z_9}{z_6 \cdot z_8} \quad (3.8)$$

then the formula for the angular velocity of the output shaft $\omega_{II} = \omega_h$ is determined as a function of speed $\omega_6 = \omega_5$ and $\omega_9 = \omega_3$:

$$\omega_h = \frac{z_6 \cdot z_8}{z_7 \cdot z_9 + z_6 \cdot z_8} \cdot \left(\omega_5 + \omega_3 \cdot \frac{z_7 \cdot z_9}{z_6 \cdot z_8}\right) \quad (3.9)$$

The missing values of the angular velocities ω_6 and ω_9 of the bevel sun wheels 6 and 9, respectively, can be expressed as a function of the given angular velocity ω_1 , because:

$$i_{1,3} = \frac{\omega_1}{\omega_3} = \frac{\omega_I}{\omega_3} = \left(-\frac{z_2}{z_1}\right) \cdot \left(-\frac{z_3}{z_2}\right) = \frac{z_3}{z_1} \quad (3.10)$$

and

$$i_{4,5} = \frac{\omega_4}{\omega_5} = \frac{\omega_I}{\omega_5} = \left(-\frac{z_5}{z_4}\right) \quad (3.11)$$

so

$$\omega_3 = \omega_9 = \omega_I \cdot \frac{z_1}{z_3} \quad (3.12)$$

$$\omega_5 = \omega_6 = \omega_I \cdot \left(-\frac{z_4}{z_5}\right) \quad (3.13)$$

Finally, the output angular velocity $\omega_h = \omega_{II}$ and the total kinematic gear ratio $i_{I,II}$ can be calculated from the formulas:

$$\omega_h = \omega_I \cdot \frac{z_6 \cdot z_8}{z_7 \cdot z_9 + z_6 \cdot z_8} \cdot \left(-\frac{z_4}{z_5} + \frac{z_1}{z_3} \cdot \frac{z_7 \cdot z_9}{z_6 \cdot z_8}\right) \quad (3.14)$$

$$i_{I,II} = \frac{\omega_I}{\omega_h} = \frac{z_7 \cdot z_9 + z_6 \cdot z_8}{z_6 \cdot z_8 \cdot \left(-\frac{z_4}{z_5} + \frac{z_1}{z_3} \cdot \frac{z_7 \cdot z_9}{z_6 \cdot z_8} \right)} = \frac{\frac{z_7 \cdot z_9}{z_6 \cdot z_8} + 1}{-\frac{z_4}{z_5} + \frac{z_1}{z_3} \cdot \frac{z_7 \cdot z_9}{z_6 \cdot z_8}} \quad (3.15)$$

To conclude the ratio of the gear is calculated.

3.3.2 Use of the Contour Graph Method for the Second Gear

The second analytical method for description of kinematics of a planetary gear will be the method of contour graphs [29]. Figure 3.4 presents the vectors of relative and absolute angular velocities of the elements of the analyzed gearbox and characteristic points A, B, C, D, E, F, G, H and J. The geometric coordinates of these points are described in the rectangular coordinate system 0 – x – y – z.

According to the method of contour graphs, the kinematic relationships of individual elements of a planetary gear (for every contour) are as follows:

$$\sum_i \omega_i^{i-1} = 0, \quad (3.16)$$

$$\sum_i r_{i,i-1} \times \omega_i^{i-1} = 0, \quad (3.17)$$

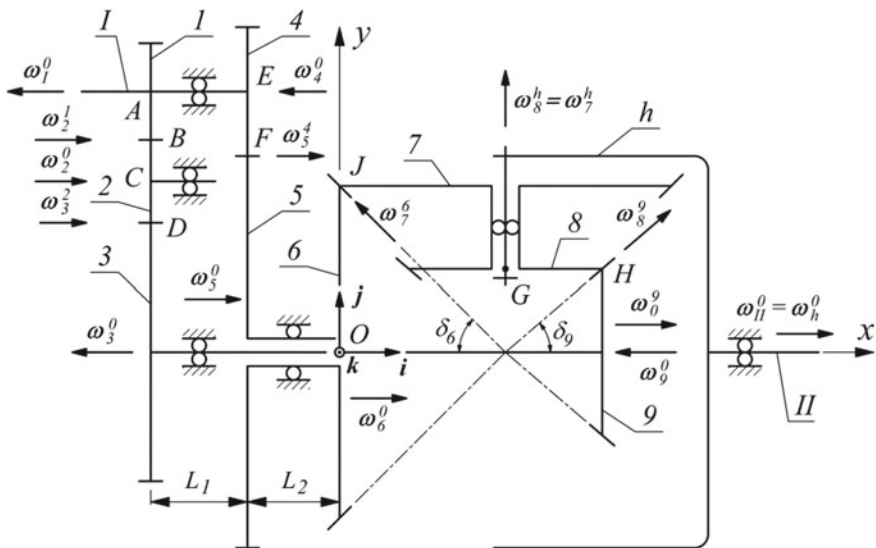
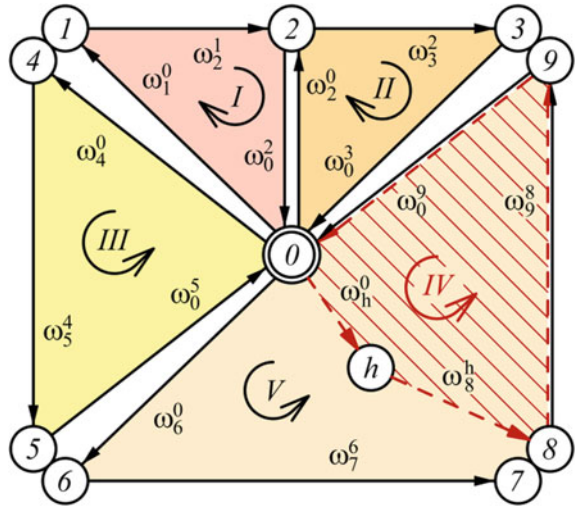


Fig. 3.4 Kinematical diagram of planetary gear with double bevel planetary wheels, meaning of descriptions analogical like in previous figure

Fig. 3.5 Contour graph of the analyzed planetary gear



where: ω_i^{i-1} —the relative angular velocity of the element i with respect to the element $i - 1$, $r_{i,i-1}$ —the position vector of the joint between the i and $i - 1$ elements (for $i = 1, 2, \dots, p + 1$, where p —total number of elements).

For $k = p_4 + p_5 = 11$ joints ($p_5 = 6$ —full joints, $p_4 = 5$ —half joints) and $p = 6$ elements (0, 1–4, 2, 3–9, 5–6, 7–8, h) there are m independent contours of analyzed planetary gear:

$$m = k - p + 1 = 11 - 7 + 1 = 5. \tag{3.18}$$

The graph of the independent contours is presented in Fig. 3.5. The contours are described via numbers in Latin notation i.e.: I, II etc. It should not be confused with description of inputs and outputs of the considered gears which is underlined, once more.

For the first contour consisting of the three elements 0, 1 and 2 arranged in the path 0, 1, 2, 0 in a clockwise direction, the following vector equations can be written:

$$\omega_1^0 + \omega_2^1 + \omega_0^2 = 0, \tag{3.19}$$

$$(\mathbf{r}_{OA} - \mathbf{r}_{OC}) \times \omega_1^0 + (\mathbf{r}_{OB} - \mathbf{r}_{OC}) \times \omega_2^1 = 0, \tag{3.20}$$

where (according to Fig. 3.4):

- $\omega_1^0 = -\omega_1^0 \cdot \mathbf{i} = -\omega_1^0 \cdot \mathbf{i}$ —the known absolute angular velocity of the gear 1 (with respect to the frame 0),
- $\omega_1^0 = \omega_1 = 25 \cdot \pi$ rad/s—given value of the angular velocity of the input shaft I ,
- $\omega_2^1 = \omega_2^1 \cdot \mathbf{i}$ —the unknown relative angular velocity of the gear 2 with respect to the gear 1,

- $\omega_2^1 = \omega_2^0 - \omega_1^0, \omega_0^2 = -\omega_2^0 = -\omega_2^0 \cdot \mathbf{i}$ - the unknown relative angular velocity of the frame 0 with respect to the gear 2, while ω_2^0 is the unknown absolute angular velocity of the gear 2,
- $\mathbf{i}, \mathbf{j}, \mathbf{k}$ —mutually perpendicular unit vectors forming Cartesian reference frame (Fig. 3.3),

Additionally, the relationships related to the geometrical parameters and positions can be written in the following form:

$$\begin{aligned} (\mathbf{r}_{OA} - \mathbf{r}_{OC}) &= (x_A - x_C) \cdot \mathbf{i} + (y_A - y_C) \cdot \mathbf{j} + (z_A - z_C) \cdot \mathbf{k} = (r_1 + r_2) \cdot \mathbf{j}, \\ x_A - x_C &= 0, \quad y_A - y_C = r_1 + r_2, \quad z_A - z_C = 0, \\ (\mathbf{r}_{OB} - \mathbf{r}_{OC}) &= (x_B - x_C) \cdot \mathbf{i} + (y_B - y_C) \cdot \mathbf{j} + (z_B - z_C) \cdot \mathbf{k} = r_2 \cdot \mathbf{j}, \\ x_B - x_C &= 0, \quad y_B - y_C = r_2, \quad z_B - z_C = 0, \end{aligned}$$

After taking into account the above relationships, Eqs. (3.19) and (3.20) take the form:

$$-\omega_1^0 \cdot \mathbf{i} + \omega_2^1 \cdot \mathbf{i} - \omega_2^0 \cdot \mathbf{i} = 0, \quad (3.21)$$

$$[(r_1 + r_2) \cdot \mathbf{j}] \times (-\omega_1^0) \cdot \mathbf{i} + [r_2 \cdot \mathbf{j}] \times \omega_2^1 \cdot \mathbf{i} = 0. \quad (3.22)$$

In order to simplify the Eq. (3.21), i.e. to perform vector multiplication, it is more convenient to write them in the matrix form:

$$\begin{bmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ 0 & r_1 + r_2 & 0 \\ -\omega_1^0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ 0 & r_2 & 0 \\ \omega_2^1 & 0 & 0 \end{bmatrix} = 0. \quad (3.23)$$

After multiplication:

$$(r_1 + r_2) \cdot \omega_1^0 \cdot \mathbf{k} - r_2 \cdot \omega_2^1 \cdot \mathbf{k} = 0. \quad (3.24)$$

Hence, from the equations vector (3.21) and (3.22) we get scalar equations for the first contour (after multiplying by scalars by versions i and k), respectively:

$$-\omega_1^0 + \omega_2^1 - \omega_2^0 = 0, \quad (3.25)$$

$$(r_1 + r_2) \cdot \omega_1^0 - r_2 \cdot \omega_2^1 = 0. \quad (3.26)$$

After further rearrangements, the same gear ratio was also obtained. It would be strongly highlighted that usage of contour graphs for an analysis of planetary gears with conical teeth has not been presented by other authors.

3.4 Application of the Dependency Graph Method for an Assessment of the Number of Gear Teeth

A new approach for the dependency graph can be obtained by using relationships that bind systems with their representation in the form of a dependency graph (in particular in relation to contour graphs).

In general, a graph is an ordered pair $G = (V, E)$, in which V is a finite set of elements called vertices of a graph, and E is a set of pairs $(v_i, v_j) (v_i, v_j \in V)$ called the edges of the graph.

The algorithm for creating a dependency graph and parametric plot for hydraulic systems was presented, among others in author's work [6]. For the contour graph from Fig. 3.5, it is possible to create a dependency graph (Fig. 3.6).

The attached \hat{G} search graph to a regular dependency graph G is defined by replacing all G graph branches with the attached equivalents (Fig. 3.7).

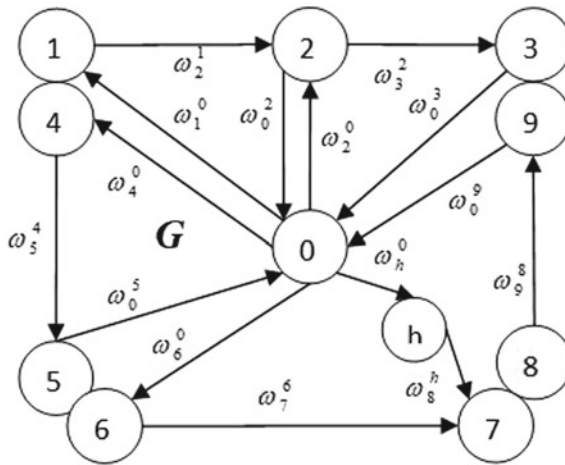


Fig. 3.6 Dependency graph for the considered problem of teeth numbers

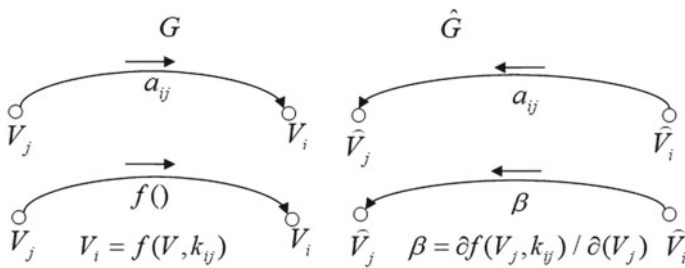


Fig. 3.7 Rules for creating a search graph for dependency graph

A search graph \widehat{G} for linear branches is created by reversing the direction of the arcs. For the non-linear branch described by the function $f(V_j, k_{ij})$ (where: k_{ij} -number coefficient) in addition to reversing the direction of the branches, there is also a change in its description from $f(V_j, k_{ij})$ onto $\partial f(V_j, k_{ij})/\partial(V_j)$, where the derivative is determined at the point V_j corresponding to the solution of the graph G , i.e. searching the entire vertex j . This graph definition, which is the equivalent of the search circuit, allows directly use the search graph method to calculate the sensitivity. The sensitivity of the V_o output value relative to the gain of the linear branch or nonlinear branch stick factor is determined on the basis of the solution of graph G and \widehat{G} in the form:

$$\begin{aligned}\frac{\partial V_o}{\partial a_{ij}} &= \widehat{V}_i V_j \\ \frac{\partial V_o}{\partial k_{ij}} &= \widehat{V}_i \frac{\partial f(V_j, k_{ij})}{\partial k_{ij}}\end{aligned}\quad (3.27)$$

The sensitivity determination method applies to both coupled and unidirectional graphs. By minimally changing only the excitation value (e.g., number of teeth) in the search graph \widehat{G} , the method can be adapted to determine the full gradient vector of the objective function. The objective function will be expressed as the sum of squares of differences between the current values of the output value V_{oi} and the set values d_i :

$$E(W) = \frac{1}{2} \sum_{i=1}^M (V_{oi} - d_i)^2 \quad (3.28)$$

in which

$$W = [W_1, W_2, \dots, W_N]^T \quad (3.29)$$

obtained gradient ∇E has the following form:

$$\nabla E = \begin{vmatrix} \frac{\partial E}{\partial W_1} \\ \frac{\partial E}{\partial W_2} \\ \dots \\ \frac{\partial E}{\partial W_N} \end{vmatrix} \quad (3.30)$$

where:

$$\frac{\partial E}{\partial W_j} = \sum_{i=1}^M (V_{oi} - d_i) \frac{\partial V_{oi}}{\partial W_j} \quad (3.31)$$

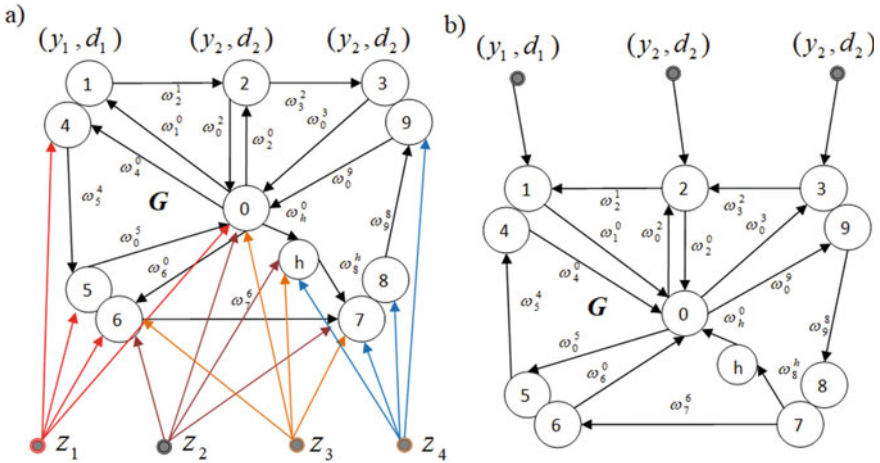


Fig. 3.8 Scheme of the implementation of search graphs

Calculation of the j th vector component of the gradient consists in the calculation of derivative output signals V_{oi} and multiplying them by the appropriate difference: $V_{oi} - d_i$. The back propagation algorithm has been developed for use in unidirectional networks. Its interpretation and generalization in the form of graphs applies to both directed graphs and recursive networks containing feedback (cycles, e.g. a drawing with trees) [6–12].

It should be noted that in the case of unidirectional networks, it is easy to extract successive layers of neurons and search the signal flow between them, while so much in the dependence graphs this task is more complicated due to the existing cycles (feedback and contours). Figure 3.8a shows an example of the implementation of search graph.

The variables (z_1, z_2, z_3) are components of the input vector Z . The output of the graph are neurons marked by their output signals (y_1, y_2, y_3) and their corresponding set signals d_1, d_2, d_3 . When creating an attached \hat{G} (Fig. 3.8b), the output nodes of the normal graph are extracted, which are now available input nodes for the \hat{G} graph. Input signal values can now be expressed as $\varepsilon_1 = y_1 - d_1, \varepsilon_2 = y_2 - d_2, \varepsilon_3 = y_3 - d_3$.

For searching, ranges of values for the number of teeth searched are assumed: $z_1, z_2, z_3, z_4, z_5, z_6, z_7, z_8, z_9$

$$z_1 \in (0, 1, 2, \dots, 100), \quad z_2 \in (0, 1, 2, \dots, 100), \quad z_9 \in (0, 1, 2, \dots, 100).$$

After T steps, the solution $z_1(T), z_2(T), z_3(T) \dots$ is obtained, which can be considered as the solution to the search graph.

In the case of a software solution, a discrete approach is used. This way, based upon the assumed startup conditions (selection of the initial graph vertex), an iterative string is created that leads to the correct final solution. An analytical record of such a

string for each of the initial vertices of the graph (see Fig. 3.6) can be represented by analytical expressions, where G_i is the iterative search string from the initial vertex i . Analytical expressions (3.32)–(3.34) represent iterative sequences for vertices 1, 2, 3 (for the graph depicted in Fig. 3.6).

$$\begin{aligned} G_{z_1}^{++} = & ({}^0z_1({}^1\omega_1z_2({}^2\omega_3z_3({}^3z_3z_9({}^4\omega_0z_0({}^5\omega_h^0h({}^6\omega_8^h z_8({}^7\omega_9^8z_9, z_8z_7)^7)^6, \\ & \omega_6^0z_6({}^6\omega_7^6z_7, z_6z_5({}^7\omega_0^5z_0, z_5z_6)^7)^6, \omega_4^0z_4({}^6\omega_5^4z_5, z_4z_1)^6, \\ & \omega_1^0z_1, \omega_2^0z_2)^5)^4, \omega_0^3z_0)^3, \omega_0^2z_0)^2, g_1z_4)^1)^0 \end{aligned} \quad (3.32)$$

$$\begin{aligned} G_{z_2}^{++} = & ({}^0z_2({}^1\omega_2^2z_3({}^2z_3z_9({}^3\omega_0^9z_0({}^4\omega_h^0h({}^5\omega_8^h z_8({}^6\omega_9^8z_9, z_8z_7)^6)^5, \\ & \omega_6^0z_6({}^5\omega_7^6z_7, z_6z_5({}^6\omega_0^5z_0, z_5z_6)^6)^5, \omega_4^0z_4({}^5\omega_5^4z_5, z_4z_1({}^6\omega_2^1z_2)^6)^5, \\ & \omega_1^0z_1, \omega_2^0z_2)^4, z_9z_3)^3, \omega_0^3z_0)^2, \omega_0^2z_0)^1)^0 \end{aligned} \quad (3.33)$$

$$\begin{aligned} G_{z_3}^{++} = & ({}^0g_3({}^1z_3z_9({}^2\omega_0^9z_0({}^3\omega_h^0h({}^4\omega_8^h z_8({}^5\omega_9^8z_9, z_8z_7({}^6z_7z_8)^6)^5)^4, \omega_6^0z_6({}^4\omega_7^6z_7, z_6z_5 \\ & ({}^5\omega_0^5z_0, z_5z_6)^5)^4, \omega_4^0z_4({}^4\omega_5^4z_5, z_4z_1({}^5\omega_2^1z_2({}^6\omega_3^2z_3, \omega_0^2z_0)^6, z_1z_4)^5)^4, \omega_1^0z_1, \omega_2^0z_2)^3, \\ & z_9z_3)^2, \omega_0^3z_0)^1)^0)^6)^5, \omega_1^0z_1, \omega_2^0z_2)^4, z_9z_3)^3, \omega_0^3z_0)^2, \omega_0^2z_0)^1)^0 \end{aligned} \quad (3.34)$$

If we have a certain n -complex system, then the vertices of the search graph correspond to the teams in E and the set of arcs.

$$Z \subseteq \{z_1, z_2, \dots, z_{n-2}\} \quad (3.35)$$

$$U = \{(e_i, e_j) \in E \times E / (e_i, e_j) \in U \Leftrightarrow u_i \text{ tests } u_j \text{ in } Z\} \quad (3.36)$$

Ultimately, the search graph is a function $\widehat{G} = (E, U, f_d)$, where f_d determines the optimal number of teeth.

Figures 3.9, 3.10 and 3.11 show the examples process of finding the optimal number of teeth z_2 , z_1 and z_3 in the computer software

Own computer program was written. The general ideas of the software are shown in Figs. 3.12 and 3.13.

The rough idea of the algorithm is a special searching through the parametrical game trees as a combination of the classical graph search algorithms. The game structures represent iterative depth-like search. The general, but more detailed, block-scheme of the prepared software is given in Fig. 3.13.

The prepared procedures join effective inspection (scan, walktroug) of the assumed search space—performing Depth First Traversal or Search and simultaneously fast Breadth First Search for the nodes/vertices in the neighborhood of the root, however taking into account cycles i.e. returns (back jumps). Search of the parametric game structures consists in call/triggering of the DFS routines for different levels (depth), starting from the initial value. In every call of DFS routine, there are

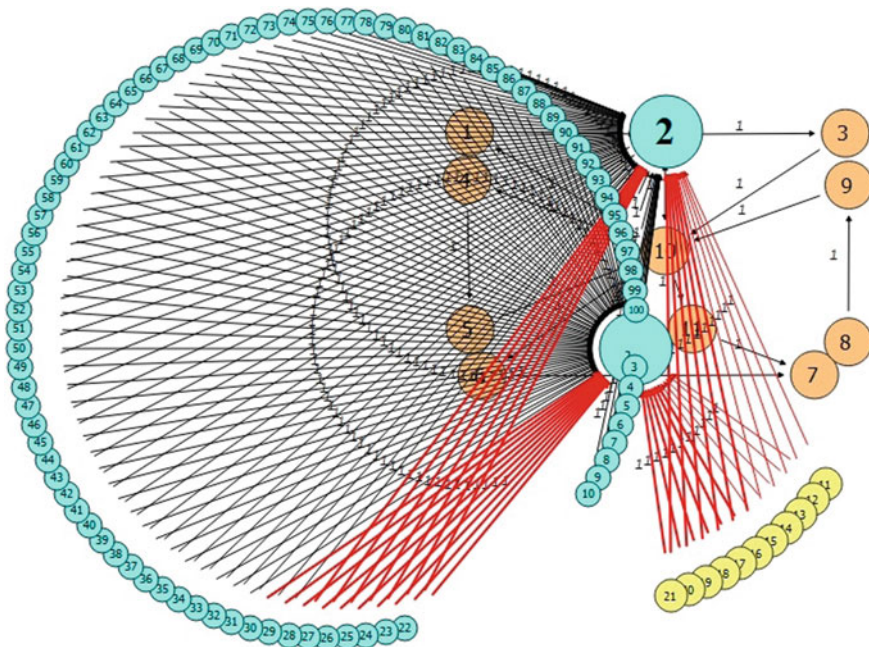


Fig. 3.9 The process of finding the optimal number of teeth for z_2

depths which cannot be overcome/crossed. The algorithm is original. Its usefulness was tested previously for similar problems by the authors [9, 10]. The analogous graph search analyses were made for all considered gear wheels. Finally, the ranges of the number of teeth for 9 gear wheels were generated for the search graphs related to the investigated planetary gear:

$$\begin{aligned}
 z_1 &\in \langle 15, 27 \rangle, z_2 \in \langle 11, 21 \rangle, z_3 \in \langle 46, 59 \rangle, \\
 z_4 &\in \langle 45, 59 \rangle, z_5 \in \langle 76, 88 \rangle, z_6 \in \langle 28, 38 \rangle, \\
 z_7 &\in \langle 27, 39 \rangle, z_8 \in \langle 17, 23 \rangle, z_9 \in \langle 13, 23 \rangle.
 \end{aligned}
 \tag{3.37}$$

For further analysis, the following tooth values z were selected from the given ranges values, with cylindrical wheel modules: $m_1 = \dots = m_5 = 2$ mm and medium bevel wheel modules with straight teeth: $m_{mt6} = m_{mt7} = m_{mt8} = m_{mt9} = 4$ mm, they are given in Table 3.3. Based on engineer experience of the authors particular values of teeth were selected which are within the established intervals.

Taking into account, among others, the formulas (3.4)–(3.15), the values of angular velocity of wheels and gear arm are:

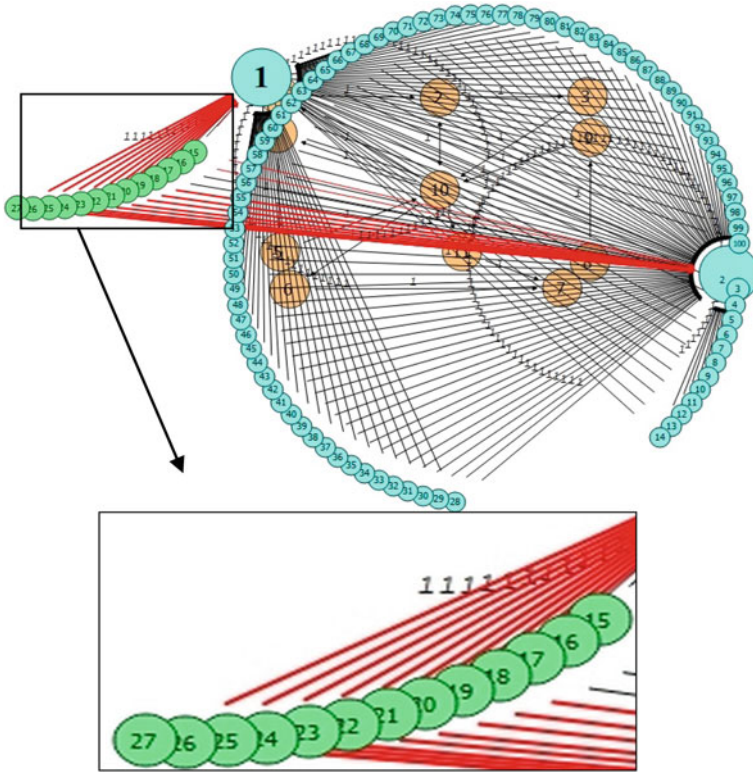


Fig. 3.10 The process of finding the optimal number of teeth for z_1

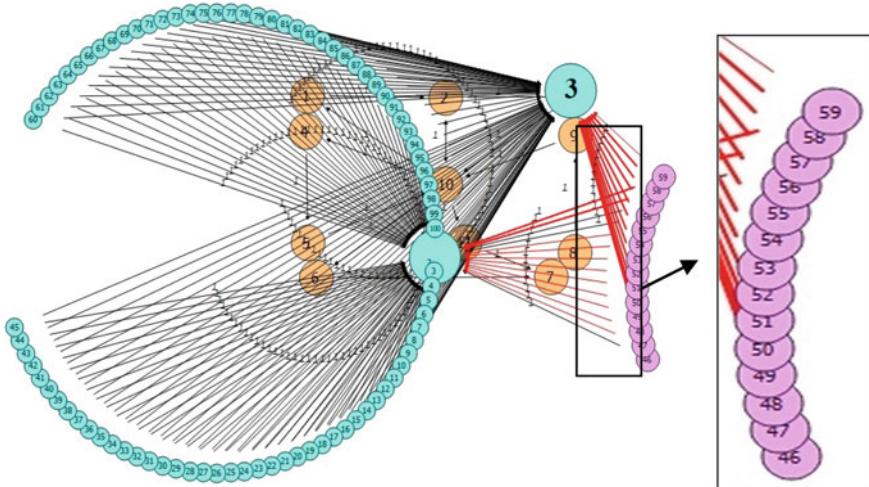


Fig. 3.11 The process of finding the optimal number of teeth for z_3

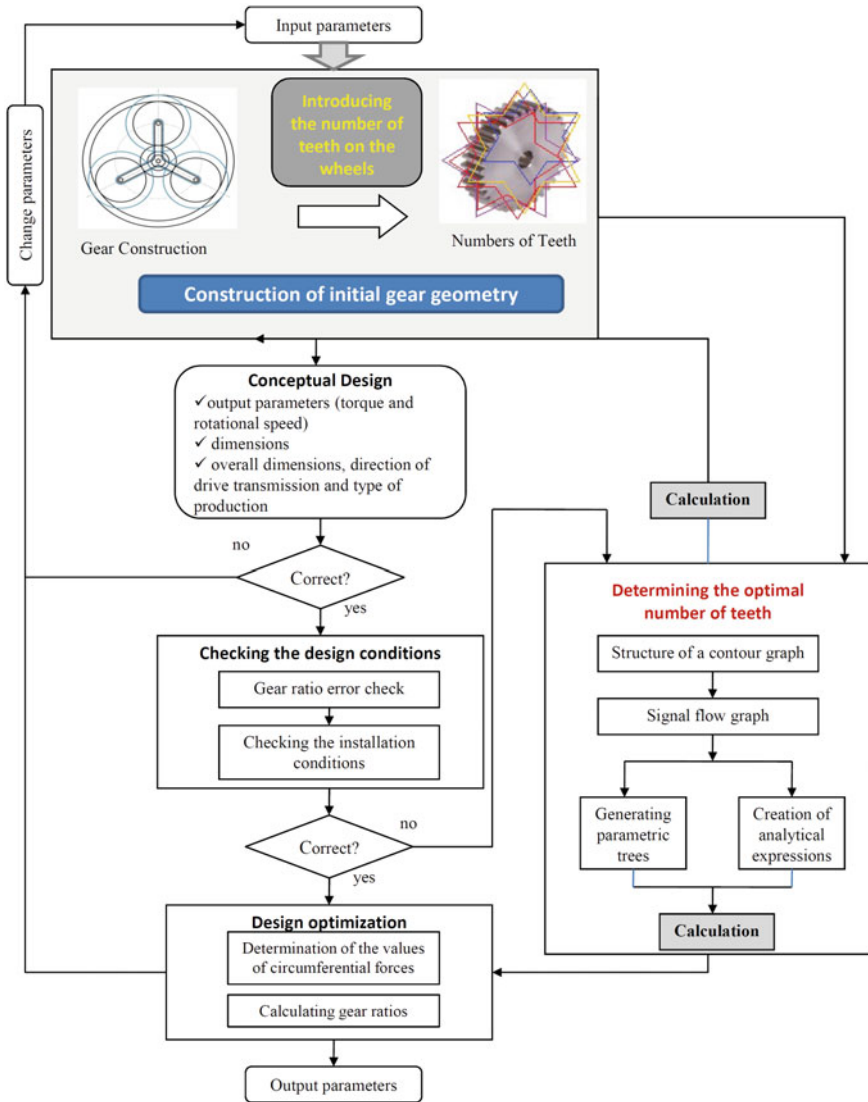


Fig. 3.12 The general bock-scheme of the system/gear analysis using graph-based modelling

- absolute speed of wheel 2:

$$\omega_2 = \omega_1 \cdot \left(-\frac{z_1}{z_2}\right) = 025 \cdot \pi \cdot \left(-\frac{21}{18}\right) = -91.6298 \text{ rad/s} \quad (3.38)$$

- relative speed of wheel 2 relative to 1 wheel:

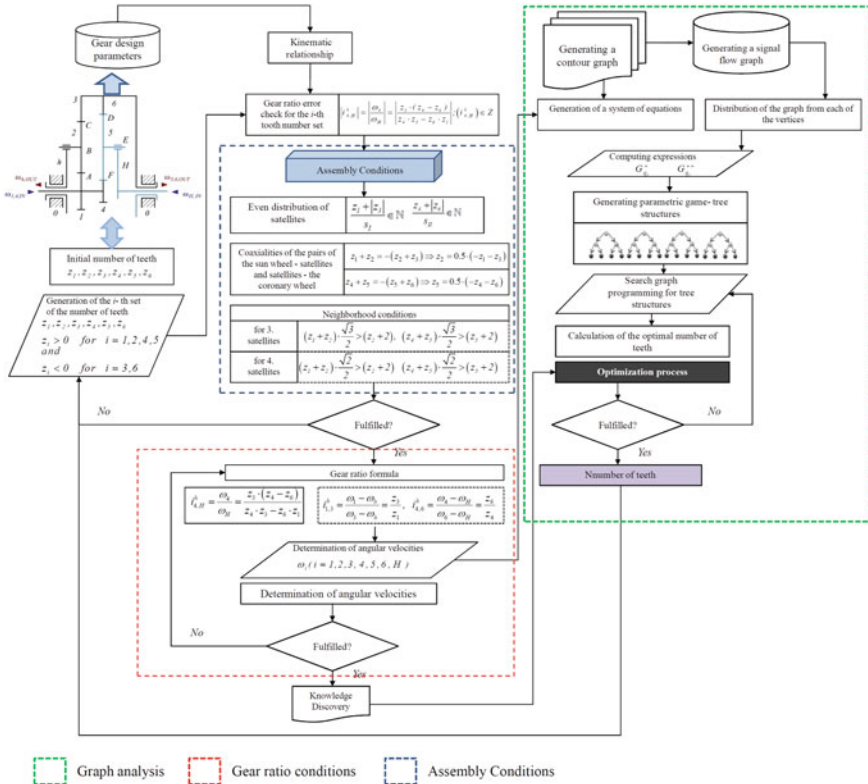


Fig. 3.13 The block-scheme of the software with the distinguished mutually connected subsystems dedicated to ratio calculation and searching for allowable teeth sets fulfilling the assumed ratio

Table 3.3 Set of chosen values for further works

Teeth numbers	Moduli
$z_1 = 21, z_2 = 18, z_3 = 56,$ $z_4 = 28, z_5 = 85$	$m_1 = \dots = m_5 = 2 \text{ mm}$
$z_6 = 36, z_7 = 36, z_8 = 21,$ $z_9 = 18$	$m_{m16} = m_{m17} = m_{m18} =$ $m_{m19} = 4 \text{ mm}$

$$\omega_2^1 = \omega_2^0 - \omega_1^0 = -91.6298 - 78.5398 = -170.1696 \text{ rad/s} \quad (3.39)$$

- absolute speed of wheel 3 (equal to wheel speed 9):

$$\omega_3 = \omega_9 = \omega_1 \cdot \frac{z_1}{z_3} = 25 \cdot \pi \cdot \frac{21}{56} = 29.45243 \text{ rad/s} \quad (3.40)$$

- relative speed of wheel 2 relative to wheel 3:

$$\omega_3^2 = \omega_3^0 - \omega_2^0 = 29.45245 - (-91.6298) = 121.08225 \text{ rad/s} \quad (3.41)$$

- absolute speed of wheel 5 (equal to wheel speed 6):

$$\omega_5 = \omega_6 = \omega_I \cdot \left(-\frac{z_4}{z_5}\right) = 25 \cdot \pi \cdot \left(-\frac{28}{85}\right) = -25.8719 \text{ rad/s} \quad (3.42)$$

- relative speed of wheel 5 relative to wheel 4:

$$\omega_5^4 = \omega_5^0 - \omega_4^0 = -25.8719 - 78.5398 = -104.4117 \text{ rad/s} \quad (3.43)$$

- arm angular speed h (equal to output shaft speed II):

$$\omega_h = \omega_I \cdot \frac{z_6 \cdot z_8}{z_7 \cdot z_9 + z_6 \cdot z_8} \cdot \left(-\frac{z_4}{z_5} + \frac{z_1}{z_3} \cdot \frac{z_7 \cdot z_9}{z_6 \cdot z_8}\right) \quad (3.44)$$

$$\omega_h = 25 \cdot \pi \cdot \frac{36 \cdot 21}{36 \cdot 18 + 36 \cdot 21} \cdot \left(-\frac{28}{85} + \frac{21 \cdot 36 \cdot 18}{56 \cdot 36 \cdot 21}\right) = -0.337614595 \text{ rad/s} \quad (3.45)$$

Finally, the kinematic gear ratio $i_{I,II}$ (Fig. 3.1) and the angular velocity ω_h and rotational n_h of arm h with output shaft II are calculated:

$$n_h = \frac{n_I}{i_{I,II}} = \frac{750}{-231} = -3.247 \text{ rpm} \quad (3.46)$$

$$i_{I,II} = \frac{\omega_I}{\omega_h} = \frac{\frac{z_7 \cdot z_9}{z_6 \cdot z_8} + 1}{-\frac{z_4}{z_5} + \frac{z_1}{z_3} \cdot \frac{z_7 \cdot z_9}{z_6 \cdot z_8}} = \frac{\frac{36 \cdot 18}{36 \cdot 21} + 1}{-\frac{28}{85} + \frac{21}{56} \cdot \frac{36 \cdot 18}{36 \cdot 21}} = -232.63. \quad (3.47)$$

The value of the kinematic ratio $i_{I,II}$, i.e. the analyzed gear is a reduction gear with the direction of rotation of the output shaft opposite to the direction of rotation of the input shaft.

For given values of the number of teeth z_1, z_2, \dots, z_9 and for cylindrical wheel modules as well as medium bevel wheel modules, it is possible to determine these value from the contour graph method but this will be omitted in this work.

3.5 Conclusions and Final Remarks

Design methods and techniques in general can be divided into:

- algorithmic methods, characterized by high formalization of operation (algorithm of operation, rules) and guaranteeing an unambiguous result when substituting the assumed input data,

- heuristic methods that guarantee general instructions for conduct, but do not always guarantee a positive result.

In the present chapter graph-based modelling was utilized for planetary gear analyzes e.g. number of teeth (synthesis problem) or velocities and ratios calculations (analysis tasks).

In case of searching for teeth numbers, graph nodes in the considered structure represent individual information states, and edges are logical transition operations that result in adequate state transformations. Initial nodes are variants of the initial problem description, end nodes are the ending of the inference process.

The graph-based methods of analysis and synthesis of planetary gears provide an alternative approaches for the accomplishing of the tasks in question. Usage of graph-based methods for the analysis of the presented gear special type with a closed internal loop has been also presented which is commonly considered as difficult to analyse. Contour graph method simplifies the task but is rarely used, however it is worth to disseminate and promote.

The methods are relatively uncomplicated, in some aspect—even natural, algorithmic and general. This confirms the usefulness of these methods for verifying the correctness of gear analysis. Graph searching methods describes the space of possible solutions to find the optimum objective function and other graph approaches allow for systematic calculation courses.

References

1. Borutzky, W.: *Bond Graph Modelling of Engineering Systems*. Springer, New York (2011)
2. Buchsbaum, F., Freudenstein, F.: Synthesis of kinematic structure of geared kinematic chains and other mechanisms. *J. Mech.* **5**(3), 357–392 (1970)
3. Chen, Z.S., et al.: Model and algorithm of optimal embedded sensors placement for gearboxes based on signed directed graph of vibration propagation. *J. Aerospace Power* **24**, 2384–2390 (2009)
4. Chen, W., Chen, Y., Xu, W.: Topology synthesis of single-DOF epicyclic gear trains based on graph theory. In: 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO), pp. 1477–1482 (2019)
5. Deo, N.: *Graph Theory with Applications to Engineering and Computer Science*. Prentice-Hall, Englewood Cliffs (1974)
6. Deptuła, A., Partyka, M.: Application of dependence graphs and game trees for decision decomposition for machine systems. *J. Autom. Mob. Robot. Intell. Syst.* **5**, 17–26 (2011)
7. Deptuła, A., Drewniak, J., Partyka, M.A.: Analysis of a planetary gear modelled with a contour graph taking into account the method of parametric play structures. *Mechanik* **7**, 640–642 (2017)
8. Deptuła, A., Drewniak, J., Partyka, M.A.: Analysis of a planetary gear modelled with a contour graph considering the decision making complexity of game-tree structures. In: II International Conference of Computational Methods in Engineering Science, Exploitation and Machine Building, ITM Web Conference, 15, 04002 (2017)
9. Deptuła, A., Drewniak, J., Partyka, M.A.: Selection of the optimal number of teeth for a biplanetary bevel gear based on discrete optimization. *Mach. Dyn. Res.* **42**(3), 55–66 (2018)
10. Deptuła, A., Drewniak, J., Partyka, M.A.: The method of searching trees in determining of the optimal number of wheel teeth for a compound, planetary gear. In: Uhl, T. (eds.) *Advances in*

- Mechanism and Machine Science. IFToMM WC 2019 in Cracow. Mechanisms and Machine Science Series, Springer, vol. 73, pp. 2853–2862 (2019)
11. Deptuła, A., Partyka, M.A.: Zastosowanie graficznych struktur decyzyjnych w metodologii projektowania i zarządzania. Tom 1: Grafy rozgrywające parametrycznie, Studia i Monografie, Zeszyt nr 482, Oficyna Wydawnicza Politechniki Opolskiej, Opole. ISBN: 973-83-66033-10-8 (2018)
 12. Deptuła, A.: Zastosowanie graficznych struktur decyzyjnych w metodologii projektowania i zarządzania Tom 3: Grafy rozgrywające parametrycznie i decyzyjne wielowartościowe drzewa logiczne w analizie przekładni planetarnych. Studia i Monografie, Oficyna Wydawnicza Politechniki Opolskiej (2020)
 13. Dietrych, J., Rugestein, J.: Einführung in die Konstruktionswissenschaft. Silesian Technical University, Gliwice (1982)
 14. Drewniak, J., Zawisłak, S.: Linear-graph and contour-graph-based models of planetary gears. *J. Theor. Appl. Mech.* **48**(2), 415–433 (2010)
 15. Drewniak, J., Zawisłak, S.: Graph methods in kinematical analysis of multi-speed epicyclic gears. *Int. J. Appl. Mech. Eng.* **17**(3), 791–798 (2012)
 16. Drewniak, J., Zawisłak, S.: Synthesis of planetary gears by means of artificial intelligence approach—graph theoretical modeling. *Solid State Phenom.* **164**, 243–248 (2010)
 17. Du, M., Yang, L.: A basis for the computer-aided design of the topological structure of planetary gear trans. *Mech. Mach. Theory* **145**, 103–690 (2020)
 18. Folenta, D.J., Motzer, J., Critelli, F.X.: Design and Development of High Horsepower Marine Planetary Gears. ASME, United Engineering Center, New York (1976)
 19. Freudenstein, F., Yang, A.T.: Kinematics and statics of a coupled epicyclic spur-gear train. *Mech. Mach. Theory* **7**(2), 263–275 (1972)
 20. Gomà Ayatsa, J.R.: Hypergraphs for the analysis of complex mechanisms comprising planetary gear trains and other variable or fixed transmissions. *Mech. Mach. Theory* **51**, 217–229 (2012)
 21. Hsu, C.-H.: A graph notation for the kinematic analysis of differential gear trains. *J. Franklin Inst.* **329**(5), 859–867 (1992)
 22. Kozik, B.: An analysis of criterion for choosing constructional solutions for aeronautical multi power path gear units. *J. KONES Powertrain Transp.* **18**(3), 169–175 (2010)
 23. Lang, S.Y.T.: Graph-theoretic modelling of epicyclic gear systems. *Mech. Mach. Theory* **40**(5), 511–529 (2005)
 24. Laus, L., Simas, H., Martin, D.: Efficiency of gear trains determined using graph and screw theories. *Mech. Mach. Theory* **52**, 296–325 (2012)
 25. Li, X., Wang, A.: A modularization method of dynamic system modeling for multiple planetary gear trains transmission gearbox. *Mech. Mach. Theory* **136**, 162–177 (2019)
 26. Li, X., Schmidt, L.: Grammar-based designer assistance tool for epicyclic gear trains. *ASME J. Mech. Des.* **126**, 895–902 (2004)
 27. Lin, Y.-S., et al.: A method and software tool for automated gearbox synthesis. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, vol. 49026, pp. 111–121 (2009)
 28. Liu, J., Yu, L., Zeng, Q., Li, Q.: Synthesis of multi-row and multi-speed planetary gear mechanism for automatic transmission. *Mech. Mach. Theory* **128**, 616–627 (2018)
 29. Marghitu, D.B.: Kinematic Chains and Machine Components Design. Elsevier, Amsterdam (2005)
 30. Wojnarowski, J.: The graph method of determining the loads in complex gear trains. *Mech. Mach. Theory* **11**(2), 103–121 (1976)
 31. Wojnarowski, J.: Graph representation of mechanical systems. *Mech. Mach. Theory* **30**(7), 1099–1112 (1995)
 32. Wojnarowski, J., Lidwin, A.: The application of signal flow graphs—the kinematic analysis of planetary gear trains. *Mech. Mach. Theory* **10**(1), 17–31 (1975)
 33. Wojnarowski, J., Zawisłak, S.: Modelling of mechanical system by means of matroids. *Mech. Mach. Theory* **36**(6), 717–724 (2001)

34. Wojnarowski, J., Kopec, J., Zawiślak, S.: Gears and graphs. *J. Theor. Appl. Mech.* **44**(1), 139–162 (2006)
35. Xue, H.-L., Liu, G., Yang, X.-H.: A review of graph theory application research in gears. *Proc. Inst. Mech. Eng. C J. Mech. Eng. Sci.* **230**(10), 1697–1714 (2016)
36. Yang, P., Pei, Z., Liao, N., Yang, B.: Isomorphism identification for epicyclic gear mechanism based on mapping property and ant algorithm. *Eng. Computers* **23**(1), 49–54 (2007)
37. Yutao, L., Di, T.: Dynamics modeling of planetary gear set considering meshing stiffness based on bond graph. *Procedia Eng.* **24**, 850–855 (2011)
38. Zawiślak, S.: *The Graph-based Methodology as an Artificial Intelligence Aid for Mechanical Engineering Design*, Bielsko-Biała (2010)
39. Zawiślak, S., Rysiński, J. (eds.): *Graph-Based Modelling in Engineering*, Springer, Cham (2017)

Chapter 4

Modeling Dynamics of Cored Wire in Molten Steel Using Linear Graph Representation



Kyrylo S. Krasnikov

Abstract The manuscript is devoted to graph-based mathematical simulation of a widespread steelmaking process, in which cored wire is injected into treatment ladle with molten steel stirred by argon blowing. Despite the process has better effectiveness than other methods its accuracy can be increased further using mathematical modelling. For example there is melt's depth, where an addition needs to be released after melting of cored wire. Due complexity of wire dynamics many models in literature are oversimplified and don't take important features. A wire has bending strains, remained after its unpacking, which leads to slightly curved shape of the lightweight wire. As consequence when it reaches a weighty melt there is a risk of swirling on the melt's surface instead of diving inside. Also ladle wall and hydrodynamic drag of molten steel can significantly influence wire motion. In this work cored wire is replaced by a mechanical system of rigid rods with initially equal length and rotational springs between them. One of the peculiarities is a variable count of rods in the system during injection. A vector-network graph is used to decompose multirod system to basic mechanical elements (rods, spherical joints, and springs) and their relationships with information about forces and torques. An incidence matrix has repetitive nature and it is created using the graph in a way that simplifies getting of cutset and circuit equations. Using sequential substitutions the equations of motion are obtained in accordance to works of other scientists, which do it for systems with kinematically linked two and three bodies. For a computer realisation the semi-implicit Euler's method is used to numerically solve SLAE. There is a parameters table of conducted numerical experiments and a chart with depth of wire's tip depending on time, which has good correspondence with a real process. Presented computer simulation helps to find an optimal injection speed (3–5 m/s) of considered cored wire in a ladle. Alternatively a wire with higher elasticity can be taken to immerse it deeper, because increasing of the speed is not always effective, especially when it is already high.

K. S. Krasnikov (✉)

Department of Systems Software, Dniprovskiy State Technical University, Kamianske, Ukraine
e-mail: kir_kras@ukr.net

Keywords Cored wire · Graph-theoretical modeling · Mathematical simulation of multibody dynamics · Semi-implicit Euler's method

4.1 Introduction

Metallurgists usually use cored wire to modify and purify chemical consistence of molten steel in a secondary ladle before a casting. When duration of the modification process is important, because temperature is lowered every minute, and high quality steel is demanded, a cored wire gives one of the best results. A problem is to find rational speed and injection place for a cored wire, which costs a lot of money and needs minimization of used length. Mathematical modeling helps in determination of rational technological parameters of the process without spending a lot of resources at a plant or a laboratory. In addition the model gives more opportunities to investigate the process than it can be in the reality, because of high temperature of a melt and its opaqueness.

The aim of this work is to prepare mathematical model of mentioned process for computer realization including a graph-theoretical formulation of mechanical system of rods and its dynamics.

There are many articles about mathematical description of wire dynamics, which present simple models and don't fit considered metallurgical process well. For example, one of them represents a wire as connection of spheres and lacks account for an important bending strain of steel shell. Others replace wire by rigid multi-body system and use stiff equations to predict its motion with assumption that it is inextensible. In our case a cored wire is decreasing its rigidity and elasticity when temperature grows from 300 to 1800 K.

In his chapter [1] O'Reilly uses Kirchhoff's rod theory to solve various problems involving elastic rod, which includes bending and twisting phenomenon. After developing the reduced dynamical system he simulated behavior of a spiral spring using a helix curve.

Taeyoung Lee and others develop an elastic string dynamics with point mass bob in their work [2]. In the beginning they define kinetic energy and potential one of the string in terms of material points on the string. And in the end they get discrete Lagrangian with a geometric numeric integrator, which precisely solves the problem over many timesteps.

Spillmann and Teschner [3] use Cosserat theory of elastic rods to model a dynamically deforming rod taking into account torsion and self-contact processing. In result authors simulate many interesting phenomenon, for instance, looping or pre-shaping of a stiff object. Disadvantage of their model is energy dissipation due renormalization of quaternions on the every timestep.

In work [4] Kharevych and others present computational tool to integrate Lagrangian dynamical system like wire or non-linear elastic toy, which influenced by a significant deformation. The tool takes into account holonomic constraints of

mechanical system. In result authors come to a double performance gain, when compared to an earlier researches.

There are researchers (Andrews and others [5–10]), which use linear graph theory to describe a multibody dynamical problem decomposing system by set of known elements. Beside of clear and detailed description of system, a directed graph is used by authors to get equations of system motion, based on physical values provided by edges of the graph. The advantage of such formulation is straightforward implementation of a mathematical model in a computer program.

Richard and others [6] used a single graph to define both rotational and translational information about the system. Therefore they associate pair of variables and equations for every edge on the graph. Also this vector network graph contains node numbering, which is needed later to synthesize an incidence matrix.

In [11, p. 95] Wittenburg gives the detailed explanation of directed graph construction related to a multibody system with consideration of forces and joints. There are illustrations of problems with open and closed chains of bodies representing them as “tree-structured system”. Presented chain of bodies connected by spherical joints [11, p. 151] (elasticity of the chain can be taken into account) has structural similarity with the one in the current work.

In graph-theoretic formulation of Shi and McPhee [12, 13] nodes correspond to body-fixed reference frames and edges—through variables (forces and torques) and across variables (displacements, velocities, and accelerations). It is similar to vector-network model in [8].

Mariti and others [14] make comparison of eleven solution methods in a numerical simulation of multibody system dynamics using Lagrange multipliers elimination. Authors made analysis of three multibody systems using coordinate partitioning method, QR/SVD decomposition and other techniques. In result these three methods can take significantly more CPU time for such model as the double planar parallelogram or slider-crank mechanism. Authors conclude that effectiveness of each method depends on considered model.

Mouad and Saka [15] present history of application of graph theoretic methods to multibody systems over last decades. They notes about significant growing of publications devoted to the graph theoretic approach.

4.2 Representation of Cored Wire Dynamics Using Directed Graphs

Following assumptions are taken to simulate the dynamics:

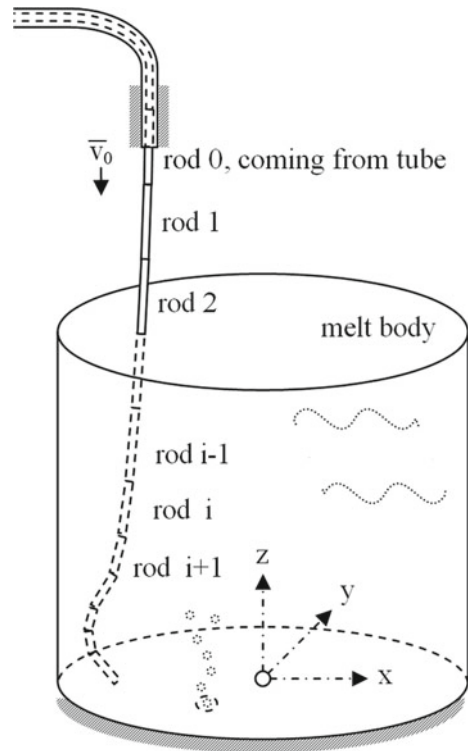
1. Cored wire is continuous and isotropic. It has a constant diameter along its length. Its speed inside guiding tube is constant (about 2 m/s).
2. The wire exhibits an elastic bending deformation modeled by a chain of rigid rods with a virtual rotational spring between adjacent ones. The angle between

adjacent rods is small (less than 30° —this simplifies a formulation of the spring force).

3. Ladle has a cylindrical shape and its wall is modeled by a potential barrier with a high elasticity.
4. Melt has a flat top surface and a constant depth.
5. Velocity field of melt can be calculated separately, because wire is too thin to affect the field and its influence can be neglected. Moreover this will save computer resources.

Cored wire is considered as a discrete system of N extensible rods (Fig. 4.1) with an initial length l . The l needs to be small enough to represent a curved wire. N is incremented after a new rod appears from the tube, and decreased after the last rod is melted up. Elasticity proportionally resists any change in rods length or their relative orientation from an initial value, according to the Hook's law. So rods also act like springs, except rod 0, which in fact is a slider while it is inside the guiding tube.

Fig. 4.1 Schematic view of rods system (wire) in argon-stirred melt body



4.2.1 Graph-Theoretic Formulation of Multirod System

To create a graph of rods chain using approaches in [8] and [13], one initially defines a node for the inertial frame of reference—it is center of ladle bottom (O on the Fig. 4.2). Then all nodes, that correspond to the centers of masses for rods are added (\bar{m}_i on the Fig. 4.2).

Rod 0 (Fig. 4.1) is coming from tube and acting like a slider in mechanical system that's why its motion is defined a priori from the known velocity of wire injection \bar{v}_0 .

Figure 4.2 shows a linear graph considering the rod system from Fig. 4.1. The rods between the first and the last one are omitted. Arrows with dashed lines represent forces.

End of guiding tube and rods are shown by dashed lines for an additional clarity. On the figure these bodies are virtually disconnected according to decomposition of the whole complex system to simple parts, motion of which are easier to define.

In addition McPhee uses the second graph to describe rotations in multibody system to make representation of a complex mechanical system simpler. Other motivation is that one couldn't just add all rotations together like translations. For the multirod system the graph consists of slider joint \bar{s} and spherical joints \bar{h} (Fig. 4.3). The difference from the previous graph is that arrows representing joints go from

Fig. 4.2 Linear graph representation of the multirod system

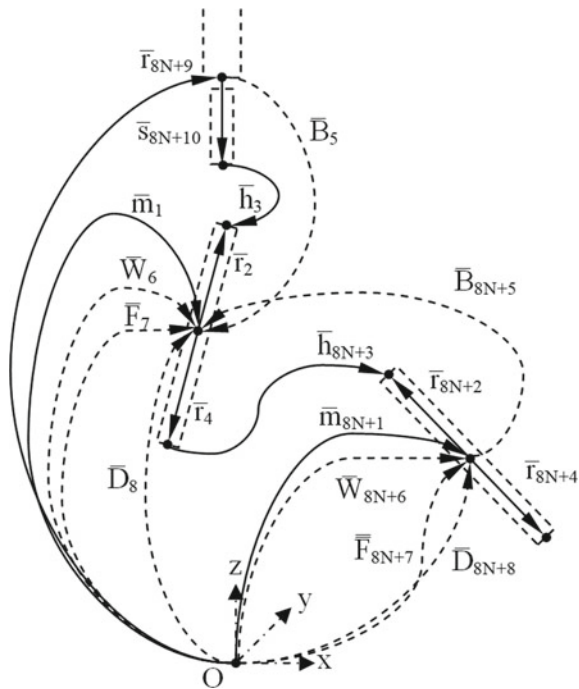
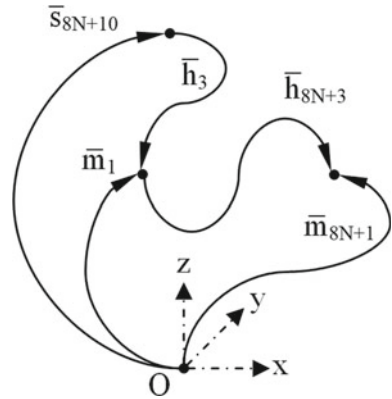


Fig. 4.3 Graph with information about rotations for the multirod system

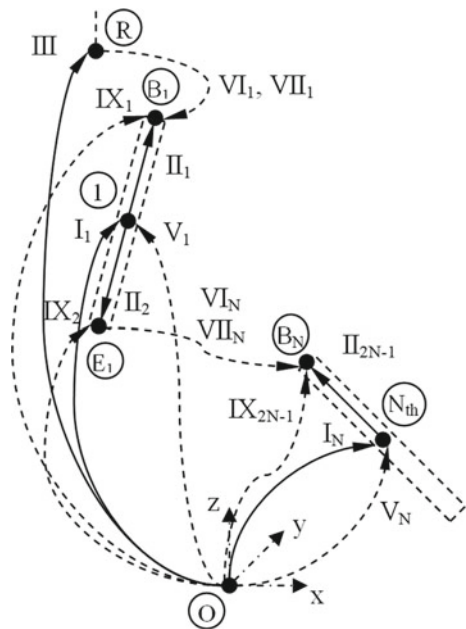


each center of mass to the next one. Relative positions of the mass centers are kept similar to those at the preceding figure.

Figure 4.4 demonstrates vector-network representation of the considered mechanical system.

The first rod (node 1) and the last one (Nth node) has some peculiarities. Nodes and edges for the first rod are repeated for any other except the last one. The node E_{Nth} with related edges is not added on the graph, because corresponding rod has only one connection—to the previous one. Such numbering of graph components

Fig. 4.4 Vector-network graph for multirod system



and their order is specifically selected to get units on the main diagonal of incidence matrix later. This simplifies formulation of cutset and circuit matrices.

It is worth noting that multirod system has free end and as seen the graph has no loops that would require additional efforts to get equations of.

A variable count of rods causes introduction of the following Roman numbering for edge types (Fig. 4.4), instead of those in [6]:

- I—Center of rod with translational and rotational inertia;
- II—Ends of rods, where there are interconnections between neighbors;
- III—Moving reference point with predefined constant velocity;
- IV—Constant direction of moving for point III;
- V—External forces—weight, hydrodynamic drag, potential barrier of ladle wall;
- VI—Torque from bending strain proportional to angle between neighbor rods;
- VII—Constraint force of spherical joint;
- IX—Torque coupling.

A similar graph was obtained by Richard [8, Fig. 4.4] for a simple system of two connected bodies, which don't have rotational spring between them. Also they aren't influenced by external forces, except one force driver, and their count is constant 2. However, his graph contains all types of edges, and is used for verification purposes.

4.2.2 Incidence Matrix as a Basis for Equations of Motion

Nodes and edges of the graph are used to create “incidence matrix” [6], which is basis for generating equations of motion. For multirod system one obtains incidence matrix from vector-network graph (Fig. 4.4). Rows are nodes and columns are vectors on the graph. A matrix for the first rod has the following look:

$$A_1 = \begin{array}{c|cccccccccc} & \text{I}_{1..} & \text{II}_{1..} & \text{II}_{2..} & \text{III} & \text{V}_{1..} & \text{VI}_{1..} & \text{VII}_{1..} & \text{IX}_{1..} & \text{IX}_{2..} \\ \hline (1).. & 1.. & -1.. & -1.. & 0.. & 1.. & 0.. & 0.. & 0.. & 0.. \\ \text{B}_{1..} & 0.. & 1.. & 0.. & 0.. & 0.. & 1.. & 1.. & 1.. & 0.. \\ \text{E}_{1..} & 0.. & 0.. & 1.. & 0.. & 0.. & 0.. & 0.. & 0.. & 1.. \\ \text{R} & 0.. & 0.. & 0.. & 1.. & 0.. & -1.. & -1.. & 0.. & 0.. \\ \text{O} & -1.. & 0.. & 0.. & -1.. & -1.. & 0.. & 0.. & -1.. & -1.. \end{array} \quad (4.1)$$

A matrix for all N rods:

$$A = [1_{i,i}^I] - [1_{i,N+i}^{I\text{odd}}] + [1_{N+i,N+i}^{I\text{odd}}] - [1_{i,2N+i}^{I\text{even}}] + [1_{2N+i,2N+i}^{I\text{even}}] + [1_{3N+1,3N+1}^{III}] \\ + [1_{i,3N+1+i}^V] - [1_{3N+1,4N+1}^{VI}] - [1_{2N+i,4N+2+i}^{VI}] + [1_{3N+1,5N+2+i}^V] \\ + [1_{N+i,4N+1+i}^{VI}] - [1_{3N+1,5N+1}^{VI}] - [1_{2N+i,5N+2+i}^{VI}] + [1_{3N+1,6N+2+i}^V] \\ + [1_{N+i,5N+1+i}^{VII}] + [1_{N+i,6N+1+i}^{IX\text{odd}}] + [1_{2N+i,7N+1+i}^{IX\text{even}}], \quad (4.2)$$

where i —changes from 1 to N . The matrix A has repetitive components for the each rod, so indexing is used for compactness.

Before Andrews [6] got equations of motions he had created three basic sets of equations. To construct “cutset” equations, which represent balance of forces and torques, he rearranged incidence matrix in such order that the first submatrix (with I–IV types of edges) has units only on the main diagonal. Then new matrix is multiplied by column vector of forces and torques:

$$[U_{L,IV,I,IV} A_{L,IV,V,IX}] FT_{L,IX} = 0, \quad (4.3)$$

where U —submatrix with units on the main diagonal; A —incidence submatrix; $FT_{L,IX}$ —column vector of forces and torques, associated with each element I.IX.

According to the mentioned work “circuit” equations can be determined using orthogonality of cutset and circuit matrices.

The third set of equations (“terminal”) defines relations between coordinates (velocities, accelerations) and forces (torques) to complete mathematical model. In his work Andrews presented table of “terminal” equations for nine types of edges. It is worth noting that in [7] authors illustrated common types of joints and terminal equations for them. The multirod system includes seven types of edges and uses spherical joints at both ends of rods. The “terminal” equations for the nine ordinary elements contain few zero forces (F_{IV} , F_{VI} , F_{VIII} , F_{IX}) and torques (T_{II} , T_{III} , T_V , T_{VII}) one can use to simplify cutset and circuit equations and reduce a final set. There are edges (specifically III and V), which need specified expressions. The edge III has simple formulation, because it represents linear motion of wire in tube:

$$\vec{f}_{III}(t) = \vec{f}_{III}^0 + \vec{v}_{III}t, \quad (4.4)$$

where \vec{f}_{III}^0 —a constant position of tube end; \vec{v}_{III} —injection velocity assumed constant; t —time.

The edges V symbolize external forces dependent on position \vec{r} , orientation $\hat{\tau}$, velocity $\dot{\vec{r}}$ of rod and fluid \vec{w} :

$$\vec{f}_V(\vec{r}, \dot{\vec{r}}, \hat{\tau}, \vec{w}) = \vec{W}(\vec{r}) + \vec{P}(\vec{r}) + \vec{D}(\vec{r}, \dot{\vec{r}}, \hat{\tau}, \vec{w}), \quad (4.5)$$

where \vec{W} —weight including into account difference of rod and melt densities; \vec{P} —a potential barrier force (acts on the rod mass center) of ladle wall, which accelerates a rod toward melt body, when the rod is outside; \vec{D} —hydrodynamic drag acts on the rod, which has orientation not parallel to the local velocity of fluid. These forces have following expressions:

$$\vec{W}(\vec{r}) = m (\vec{g}^* \cdot \vec{r}) \hat{z}, \quad (4.6)$$

where m —a rod mass; \vec{g}^* —gravity acceleration multiplied by densities ratio ($1 - \rho_{\text{melt}}/\rho_{\text{wire}}$); \vec{r} —a radius-vector of the rod mass center; \hat{z} —a unit vector in the direction

of z -axis.

$$\vec{P}(\vec{r}) = \kappa_w \left[(R_w - \|\vec{r}_{xy}\|) \frac{\vec{r}_{xy}}{\|\vec{r}_{xy}\|} + (\vec{r} \cdot \hat{z}) \hat{z} \right], \quad \vec{r}_{xy} = \vec{r} - (\vec{r} \cdot \hat{z}) \hat{z}, \quad (4.7)$$

where κ_w —elasticity coefficient for the ladle wall (equals zero if corresponding rod is inside); R_w —radius of the wall; \vec{r}_{xy} —projection of radius-vector \vec{r} on the horizontal plane x - y ; \hat{z} —a unit vector in the z -axis direction—it helps one to project radius-vector \vec{r} on that axis and to distinguish the vertical contact force from the radial one.

$$D(\vec{r}, \dot{\vec{r}}, \hat{\tau}, \vec{w}) = -C_D A \rho_m \frac{\vec{w}^\perp \|\vec{w}^\perp\|}{2}, \quad \vec{w}^\perp = (\dot{\vec{r}} - \vec{w}) - \hat{\tau} (\dot{\vec{r}} - \vec{w}) \cdot \hat{\tau}, \quad (4.8)$$

where C_D —drag coefficient (around 0.5 for rod, however it can depend on the velocity of surrounding melt); A —a cross sectional area (wire diameter multiplied by rod length); ρ_m —melt density; \vec{w} —local velocity of melt; $\hat{\tau}$ —unit vector of rod's orientation.

The edges VI correspond to the rotational springs between rods and depend on their orientations $\hat{\tau}$ (for the orientation of rod 0 the norm of injection velocity is used $\hat{\tau}_0 = \vec{v}_{III} / \|\vec{v}_{III}\|$):

$$\vec{T}_{VI}(\hat{\tau}_{i-1}, \hat{\tau}_i) = \kappa_b(T) (1 - \hat{\tau}_{i-1} \cdot \hat{\tau}_i) (\hat{\tau}_{i-1} \times \hat{\tau}_i), \quad (4.9)$$

where κ_b —elasticity coefficient of rotational spring between two rods.

Coefficients of rigidity are inverse proportional to a local temperature, because cored wire lost its flexibility under high temperature of molten steel and moves like a textile thread in the water before complete dissolution.

To get second-order differential equations one uses above cutset Eq. (4.2) to find needed expressions as demonstrated in [8]. Finally system of equations is obtained in the following form using substitutions:

$$\mathbf{M} \cdot \mathbf{x} = \mathbf{F}, \quad (4.10)$$

where \mathbf{M} —symmetrical mass/moment matrix; \mathbf{x} —column vector of unknown accelerations, constraint forces and torques. That form is famous to solve.

After solving system of linear algebraic equations one integrates accelerations to obtain velocities and coordinates of mass centers for the new time layer. When the moving point III moves to the distance of rod length the new rod appears before the first rod and the point moved to the initial position. One can use a doubly-linked list, instead of a simple array, to store the data of rods in memory as alternative of shifting of all data in the array. This can be more efficient during calculation, because the data is accessed sequentially (no random access).

Length of rod is changed when forces parallel to the rod axis are greater than an elastic deformation limit, which, of course, decreases when temperature of rod

rises. Amount of the change is enough to satisfy a limit of elastic deformation again. Changes are performed until rod is melted or while their values are physically correct.

4.3 Results

Numerical experiments are conducted using parameters obtained from the real process on a metallurgical plant (Table 4.1).

The model can show instability when time step is not sufficiently low so the timestep of 0.001 s is used to simulate the process between 5 and 10 min of model time. Metallurgists usually use such duration to inject cored wire into molten steel.

Figure 4.5 illustrates perspective view from above with obtained chain of rods. Each rod is visualized by a cuboid of two colors—brown and green. Brown ones are above the melt (it is transparent) and green ones are inside of it. A simple directional light gives more illumination to those parts of wire, which are oriented horizontally.

As seen the green part of wire quickly loses its rigidity under melt's surface and looks curvy, which is predictable.

The light pink circle on the figures is the ladle bottom. The black arrows at the bottom shows directions of fluxes in computed melt velocity field on the horizontal plane at the ladle's bottom. Those fluxes can greatly influence wire's motion. For the velocity field the cylindrical coordinate system is used since it fits the geometry of ladle. Ladle's wall is light cyan.

If the injection speed is increased to 7 m/s, zenith angle is set to 2.5 radians and elasticity coefficient of a wire becomes 5000 N/m then it easily reaches ladle's bottom at around the 2nd second of model time (Fig. 4.6). The wall acts like a barrier at the way of the wire. The barrier forces the wire to bend and to slide down to the ladle's bottom, which is another obstacle. At the corner the wire behaves in a predictable way—curls up like a thread because rigidity of steel approaches to zero at temperature around its melting point.

Table 4.1 Physical parameters for numerical experiments

Melt body		Cored wire		End of guiding tube	
Radius	1.1 m	Diameter	10 mm	Height above melt	1 m
Height	2.3 m	Thickness	0.2 mm	x-coordinate	0.2 m
Density	7000 kg/m ³	Density	3200 kg/m ³	y-coordinate	0.02 m
Wall elasticity coefficient	10,000 N/m	Elasticity coefficient	750 N/m	Zenith angle of direction	π rad (down)
		Rod length	0.01 m	Azimuth angle of direction	0 rad

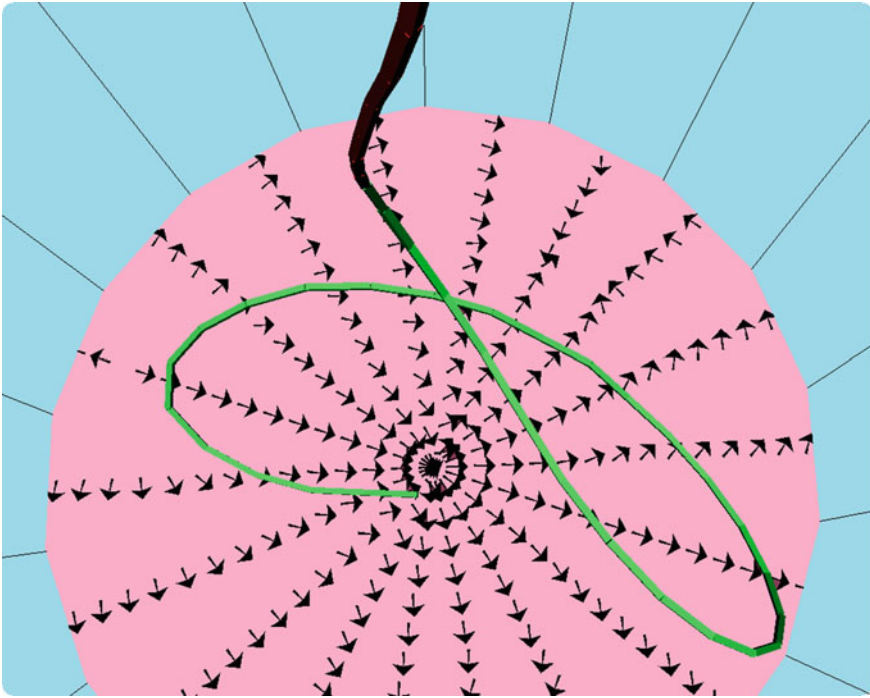


Fig. 4.5 Computed state of multirod system at the 33.1 s (injection speed—5 m/s)

4.3.1 Depth of Wire End Depending on Time

There is requirement to release addition from wire as deep as possible.

Because the addition needs time to react with melt before absorbing by the slag layer above the melt. One of the methods to control the depth of addition release is changing its injection speed (or feed rate). Numerical experiments with mentioned above parameters are conducted with the speed ranging from 1 to 5 m/s (Fig. 4.7). Such speeds are supported by wire-feeding machines for injection. As seen in the beginning of injection the wire with the highest speed reaches a bottom of ladle, but in the following time it moves around half of melt height. The reason is a decreasing of elasticity as well as not sufficient speed of injection, so fluid fluxes lift the rods higher.

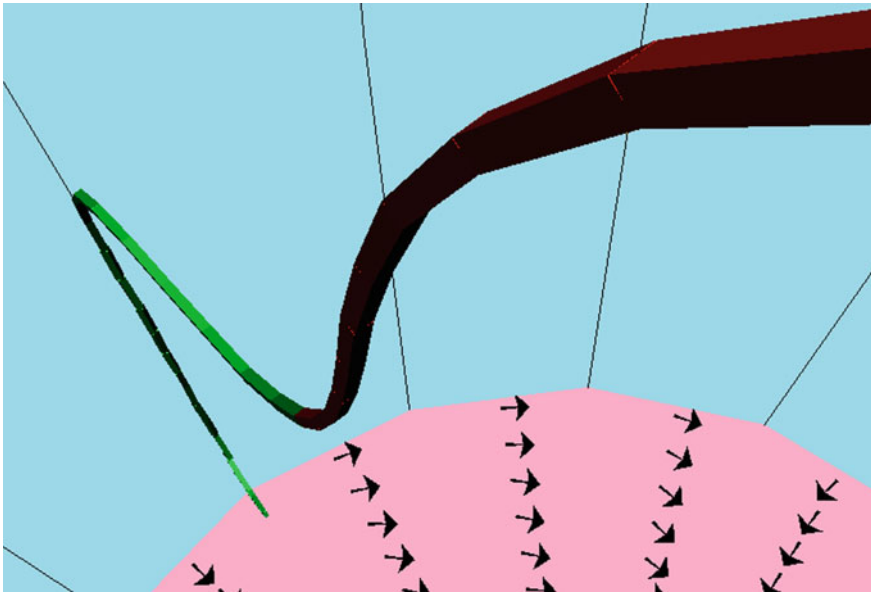


Fig. 4.6 A long contact of wire with ladle’s walls at around 2 s after start of injection (speed of injection is 7 m/s, initial elasticity of rotational springs is 5000 N/m and zenith angle of guiding tube is 2.5 rad)

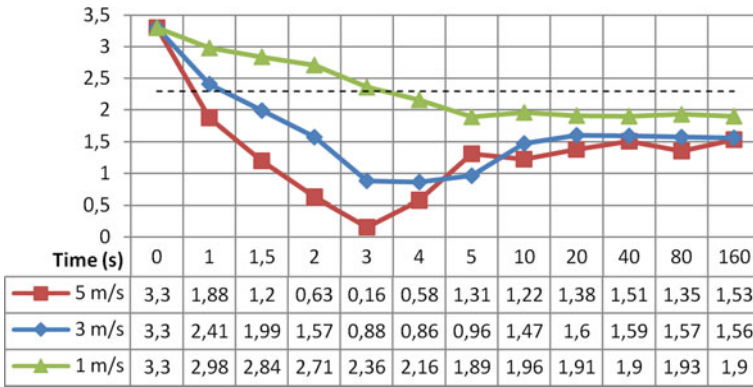


Fig. 4.7 Numerically predicted z-coordinate (in meters, lower are better) of wire’s tip for three different injection speeds. Dashed line is depth of melt

4.4 Conclusions

The manuscript propose a mathematical model of cored wire dynamics, which can predict a complex three dimensional movement taking into account bending elasticity, hydrodynamic drag of melt, contact with ladle wall. Presented results of numerical experiments show that the deepest release is achieved by using speed 5 m/s. Also in this case temperature losses of the melt will be the smallest, because duration of injection is minimal—only 3 min needed to insert 50 kg of addition (for 3 m/s—the duration is 5.1 min and for speed 1 m/s—15.5 min).

Therefore this injection speed is recommended to use with the considered sizes of melt's body and cored wire. However, not all wire-feeding machines can reach 5 m/s, so metallurgist can choose a variant with the speed of 3 or 4 m/s. It is worth noting that the spike at the 5th second (Fig. 4.7) can be explained by impact of wire with ladle's bottom—high speed brings some surprises.

In addition the depth of addition release can be improved by enlarging elasticity of wire. For a test, the speed is set to 3 m/s and the initial elasticity coefficients for rotational springs are 5000 N/m. As result, after a 15th second, addition is constantly releasing at about 1.2 m above the ladle's bottom, which is even better than combination of 5 m/s and 750 N/m correspondingly (around 1.4 m above the bottom). So increasing wire's rigidity to 5000 N/m (without changing the speed at 3 m/s) lowers realizing point by 0.3–0.4 m. This is another way to improve effectiveness of the considered process.

The further research can be devoted to the injection of two or four wires simultaneously, because they are often used for big metallurgical ladles. Therefore a multi-threaded implementation of SLAE solution is needed to speed up computation of system with hundreds of rods. Furthermore the configuration with multiple wires can lead to occasion impacts of the bodies influencing motion of each other. An extra possible situation is wire breaking into parts, one of which will be in a free motion until melted.

Acknowledgements Author thanks to the scientists of Institute of ferrous metallurgy of NASU for their consultations devoted to the considered metallurgical process and the structure of cored wire. Also there is gratitude to the all reviewers for their comments and suggestions.

References

1. O'Reilly, O.M.: Kirchhoff's rod theory. In: Modeling Nonlinear Problems in the Mechanics of Strings and Rods. Interaction of Mechanics and Mathematics. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-50598-5_5
2. Lee, T., Leok, M., McClamroch, N.H.: Dynamics of a 3D elastic string pendulum. In: Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference. Shanghai, P.R. China, Dec 16–18, pp. 3347–3352 (2009)

3. Spillmann, J., Teschner, M., Corder: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects. In: Eurographics ACM SIGGRAPH Symposium on Computer Animation 2007, San Diego, CA, Aug 03–04, pp. 63–72 (2007)
4. Kharevych, L., et al.: Geometric, variational integrators for computer animation. In: Eurographics/ACM SIGGRAPH Symposium on Computer Animation (2006)
5. Andrews, G.C., Kesavan H.K.: Simulation of multibody systems using the vector-network model. In: Magnus, K. (ed.) Dynamics of MULTIBODY SYSTEMS. Springer, Berlin (1978)
6. Richard, M.J., Anderson, R., Andrews, G.C.: Generalized vector-network formulation for the dynamic simulation of multibody systems. *J. Dyn. Syst. Meas. Contr.* **108**, 322–329 (1986)
7. Andrews, G.C., Richard, M.J., Anderson, R.J.: A general vector-network formulation dynamic systems with kinematic constraints. *Mech. Mach. Theory* **23**(3), 243–256 (1988)
8. Richard, M.J., Anderson, R.J., Andrews, G.C.: The vector-network method for the modelling of mechanical systems. *Math. Comput. Simul.* **31**, 565–581 (1990)
9. McPhee, J.J.: On the use of linear graph theory in multibody system dynamics. *Nonlinear Dyn.* **9**, 73–90 (1996)
10. Banerjee, J.: Graph-theoretic sensitivity analysis of dynamic systems. PhD thesis. University of Waterloo, 191 pp (2013)
11. Wittenburg, J.: Dynamics of Multibody Systems, 2nd edn. Springer, Berlin (2008). <https://doi.org/10.1007/978-3-540-73914-2>
12. Shi, P., McPhee, J.: Dynamics of flexible multibody systems using virtual work and linear graph theory. In: Multibody System Dynamics, vol. 4, pp. 355–381 (2000)
13. McPhee, J.J.: Dynamics of Multibody Systems: Conventional and Graph-Theoretic Approaches. University of Waterloo, Canada, Department of Systems Design Engineering (2004)
14. Mariti, L., Belfiore, N.P., Pennestri, E., Valentini, P.P.: Comparison of solution strategies for multibody dynamics equations. *Int. J. Numer. Meth. Eng.* **68**, 637–656 (2011). <https://doi.org/10.1002/nme.3190>
15. Mouad, G., Saka, A.: An overview of graph theoretic problems: state of the art and main idea. In: International Conference on Integrated Design and Production (2016)

Chapter 5

Methodology of Solving Selected Routing Problems



Bogna Mrówczyńska

Abstract The article presents the methodology of solving selected routing problems, which include the traveling salesman problem (TSP) and the arc routing problem (ARP). Graphs are used to model problems, which have become a natural language useful for describing the created models. Basic theorems of graph theory are used to solve routing tasks. TSP is reduced to the task of determining the Hamilton cycle in the complete graph, and ARP to the task of determining the Euler cycle in the Euler graph. TSP is NP—hard problem. ARP tasks may become such in complex cases. Artificial immune systems are used as a tool supporting solving the formulated problems. The proposed tools are very effective also for large tasks. The concepts and theorems of graph theory are used here to reduce a given problem to a form that is most convenient to be solved by the adopted method. TSP and ARP solving methods have a wide range of applications, including in transport logistics and in the organization of production. In addition, TSP can be used in the control of CNC machines and even in DNA sequencing.

Keywords Traveling salesman problem · Arc routing problem · Artificial immune system · Clonal selection · Graphs

5.1 Introduction

The aim of this article is to present an original method of solving complex routing tasks using elementary models and basic theorems of graph theory and artificial immune systems. The article presents two elementary problems of graph theory: the first is the task of determining the Euler cycle in a connected graph, and the second—the determination of the Hamilton cycle. The first task initiated the emergence and development of graph theory. In 1736, Leonhard Euler, a well-known mathematician and physicist staying in Königsberg, formulated and solved the problem known to this day as the problem of Königsberg bridges. It relates to the following story: the

B. Mrówczyńska (✉)
Faculty of Transport, Silesian University of Technology, Gliwice, Poland
e-mail: Bogna.Mrowczynska@polsl.pl

© Springer Nature Switzerland AG 2022
S. Zawiślak and J. Rysiński (eds.), *Graph-Based Modelling in Science, Technology and Art*, Mechanisms and Machine Science 107,
https://doi.org/10.1007/978-3-030-76787-7_5

park situated on two islands on the Pregola River was a favourite meeting place and walks for Euler and his scholars colleagues. The islands were connected with each other and with the river banks by seven bridges (Fig. 5.1a). Euler considered if it was possible to cross all bridges without crossing the same bridge twice. He illustrated the problem with a multigraph (Fig. 5.1b, c), whose edges represent bridges, and nodes represent islands or river banks. Then he formulated and proved the theorem

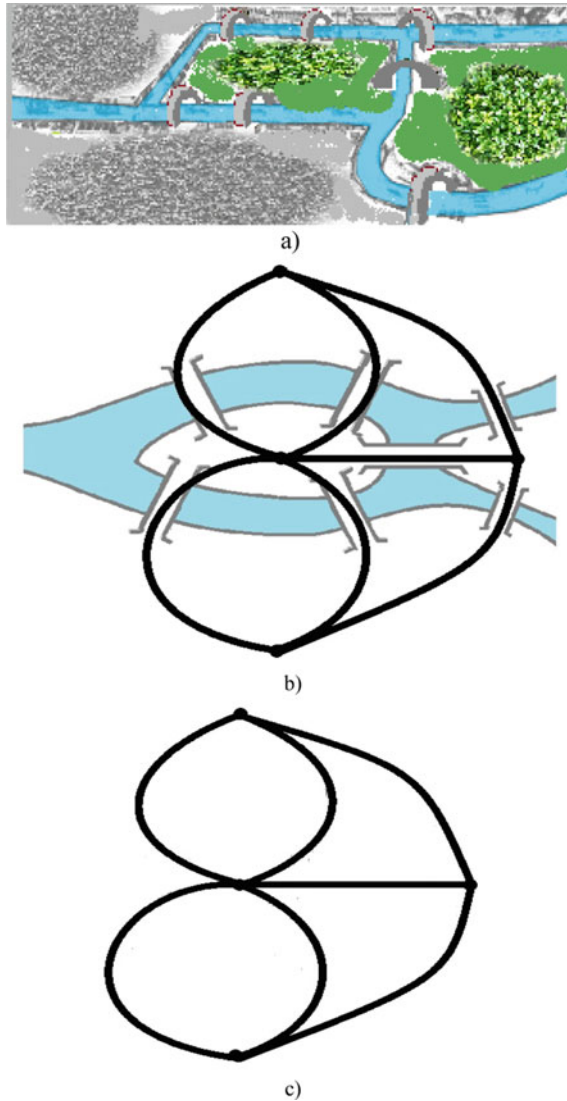


Fig. 5.1 a The bridges on the Pregola River, b the routes across the bridges on the Pregola River, c the multigraph representing the problem

that in a connected graph it is possible to determine the cycle passing through all the edges exactly once, when the graph does not have odd-degree vertices or has exactly two of them. Such a cycle is called the Eulerian cycle. Later in the article, the problem of determining the Euler cycle will be called the Euler problem (EP). Finding of Euler cycles is used here to solve problems formulated more generally as arc routing problems (ARP).

The Hamilton cycle is a cycle that passes through each vertex of the graph exactly once. The task of determining the minimum Hamilton cycle on a connected graph, is known as the traveling salesman problem (TSP). The first formulation is considered to be a practical description of the traveling salesman route planning problem in a textbook published in 1832: “*Der Handlungsreisende – wie er sein soll und was er zu thun hat, um Aufträge zu erhalten und eines glücklichen Erfolgs in seinen Geschäften gewiß zu sein – von einem alten Commis-Voyageur*”.

The traveling salesman task was formulated as a math problem by the Austrian mathematician Karl Menger in 1930. Since the 1950s, algebraic methods of solving the traveling salesman problem have been developed, such as integer linear programming, dynamic programming [4, 13], the branch—and—bound method [14], branch—and—cut [2, 38] and others. In 1972 RM Karp in the article [13] proved that the traveling salesman task is NP-hard, and in 1981 Lenstra, and Rinnooy Kan, they proved in the article [17] that in general the routing tasks, which include the problems of traveling salesman and Euler, are NP—hard and have shown that accurate algorithms are only effective for small problems.

For larger tasks, the strict methods of solving turned out to be insufficiently effective, because, with the increase of the number of vertices or edges, the number of variables increases rapidly. Unfortunately, the real problems can be much bigger and there are various conditions and restrictions. In this case, heuristics and meta-heuristics, which include methods of artificial intelligence, are often more suitable for practical applications. These methods often do not provide optimal solutions in a strict, mathematical sense, but in a reasonable computational time, acceptable solutions are obtained, sufficiently close to the optimal ones.

The most commonly used methods are: integer linear programming methods [42], dynamic programming [4, 13], division and constraint method [16], branch—and—cut method [2, 38]. In recent years, solving tasks modelled with the methods described above has been supported by artificial intelligence methods. The most frequently used are: genetic and evolutionary algorithms [22], immunological algorithms, simulated annealing [6], taboo search [15, 18, 45], the nearest neighbor method [14, 41], ants algorithms [11, 45], swarm algorithms [19, 44], neural networks [3, 20], authors’ own heuristics and others. The methods listed are among the most popular, but there are still more and more new methods.

In the calculation methodology presented in the article, artificial immune systems were used. It is one of the newer methods of artificial intelligence that was created at the turn of the twentieth and twenty-first centuries. Artificial immune systems algorithms mimic the natural immune system of a mammal. The following patterns are most often used: clonal selection, negative selection, immune networks, dendritic cell mode [9, 10]. The tools described in the article use the clonal selection paradigm.

Their use in solving the traveling salesman problem was first described in [10]. Artificial immune systems are also used for clustering [12]. Comparisons of artificial immune systems with other methods show that they are efficient and fast.

The main driving force behind the development of methods of solving the traveling salesman problem is the operation of vehicles focused on minimizing the costs of this operation, and recently also on minimizing the negative impact on the environment [34]. TSP models is used in scheduling the work of machines in a production or assembly hall [25]. The result will be the smooth use of all machines, without unnecessary downtime, and the consequence may be an increase in production. TSP can also be used when placing goods on warehouse shelves [26, 30]. The traveling salesman problem can also be used to optimize the loading of heterogeneous packages [27, 28, 37]. The optimal distribution of the load in the trailer of a large delivery vehicle or in the hold of an aircraft is not only economically significant due to the possibility of loading more packages, but also affects the safety of transport.

A new field in which the TSP model has been applied is numerical control of machine tools [11, 15]. Shortening the path of a single drill, and additionally determining the sequence in which other drills are to be used, e.g. when holes have different diameters, significantly shortens the machining time of a single element, which in the case of mass production, with the same size of products, significantly reduces working time of machine tools, i.e. reducing their wear, working time of operators, which together reduces production costs. The traveling salesman problem has also been applied in biology for DNA sequencing [5, 40].

ARP is widely used in various tasks related to the operation of motor vehicles, such as planning the work of vehicles to keep streets clean, snow plows [8], vehicles transporting waste from residential estates to landfills and waste processing plants [8], school buses transporting students to schools [39, 43] and to restore the availability of the road network damaged during natural disasters [1].

5.2 Methodology

Reviewing the various TSP and ARP tasks, it can be concluded that artificial intelligence methods are used as tools to solve the previously formulated integer linear programming problem, i.e. the system of linear equations and inequalities, which were obtained in this way. The presented methodology uses other approach. The solved problem is represented by a graph which, depending on the type of problem, is extended to a complete or Euler graph. Such an extension is performed by methods of graph theory. Thus, graphs are obtained in which, respectively: in a complete graph, each Hamiltonian cycle is an acceptable solution, and in an Euler graph, each Euler cycle is an acceptable solution. Solving the problem boils down to finding the optimum among these cycles because of the accepted criteria and meet the given criteria and constraints.

In the proposed methodology, artificial immune systems (AIS) were used for many calculations. AIS is an artificial intelligence method that mimics the behaviour

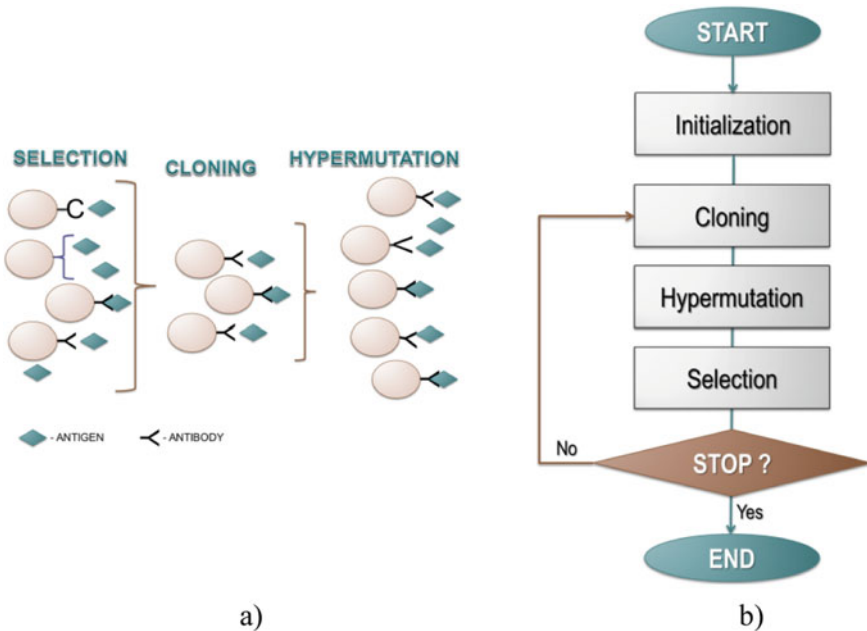


Fig. 5.2 Clonal selection: **a** natural, **b** artificial

of the natural human immune system. The algorithms used in the methodology use a clonal selection pattern. In short, it looks like this: when antigens attack the human body, they encounter mechanical (skin), chemical (tears, mucus) or physical (increased temperature) barriers. If these fail, the antigens end up in the body. Then the antibodies come to defense. Their goal is to kill antigens. Those that do it more effectively are cloned. Then mutated. Among the mutations, the best are re-cloned and mutated, etc., until antibodies are obtained that effectively inactivate the antigens in question. The mechanism of operation of natural clonal selection and the algorithm of artificial clonal selection are presented in Fig. 5.2a, b.

Although the tasks of traveling salesman and Euler are different, similar phases can be noticed in solving them. The basic scheme of proceeding in solving the traveling salesman and Euler problem is presented in Fig. 5.3, and the characteristics of individual elements of the methodology are presented below.

5.2.1 Topology

Regardless of whether the problem to be solved requires finding the Hamilton cycle or the Euler cycle in the graph representing it, first the topology of the problem should be prepared. An example of topological representation of streets in a housing estate is shown in Fig. 5.4.

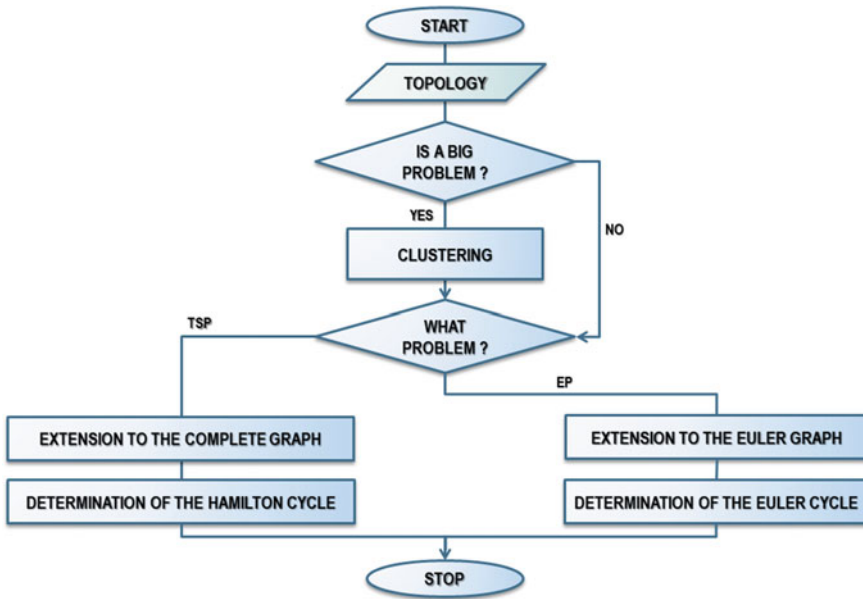


Fig. 5.3 Scheme of the multigraphs extension procedure

A mixed graph G is a topological representation of the problem:

$$G = (V, E, A) \tag{5.1}$$

where:

- V non-empty set of vertices,
- E set of edges, that is, two-element subsets of the set V ,
- A a set of elements of a Cartesian product:

$$V \times V = \{(x, y) : x \in V, y \in V\} \tag{5.2}$$

Elements (x, y) form ordered pairs called arcs, where x is the starting point, and y is the final one.

The problem of a traveling salesman for a mixed graph G is solved by determining the minimum Hamilton cycle for this graph. It is a Hamilton cycle, for which the sum of weights of edges and arches is minimum.

The Hamilton cycle will be represented by a sequence of vertices:

$$(v_0, v_1, \dots, v_i, \dots, v_n) \tag{5.3}$$

where $\forall i \in N \cup \{0\}: v_i \in V$.

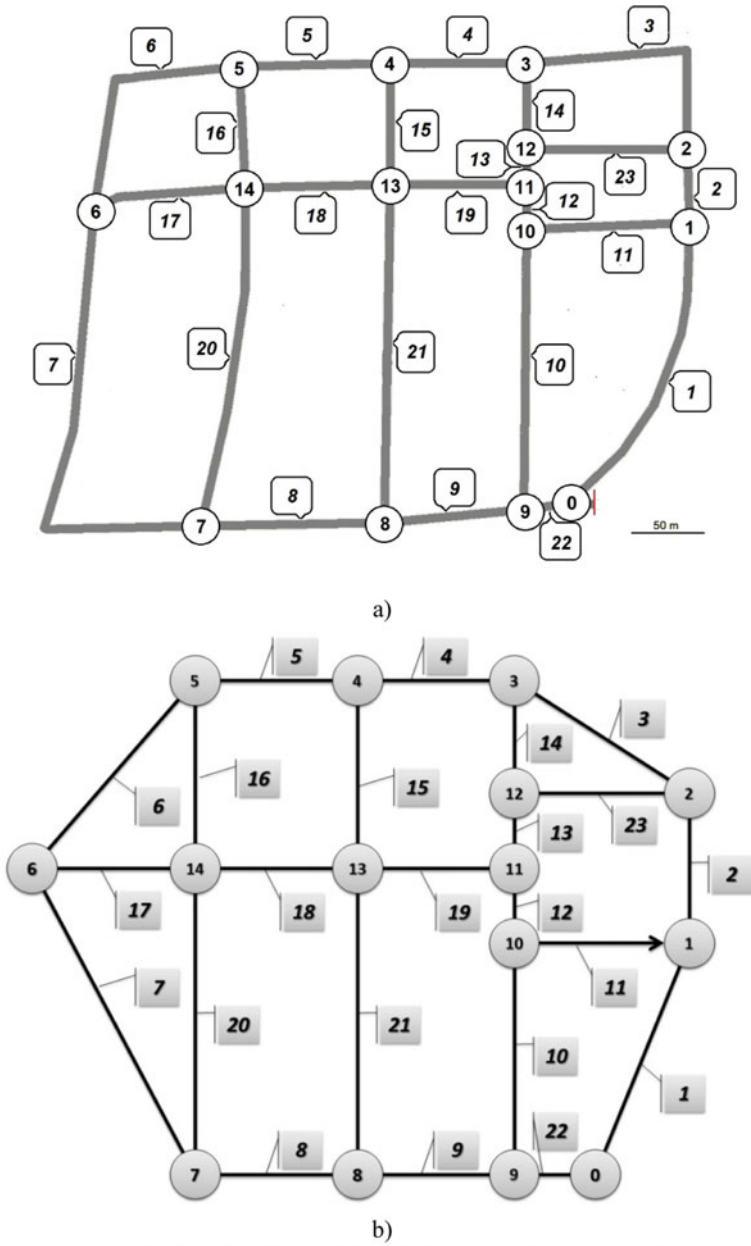


Fig. 5.4 a) Numbering of vertices and edges of the graph on the estate plan, b) graph representing the estate

The cycle begins and ends in v_0 . The sequence (5.3) is a permutation of a sequence $(0, 1, 2, \dots, n)$.

The Euler Problem in a mixed graph is the task of the determination of the Euler's cycle. Euler's cycle will be represented by the sequence:

$$(v_0, e_0, v_1, e_1 \dots, v_i, e_i \dots e_{n-1}, v_n) \quad (5.4)$$

where

$$\{\forall i \in \mathbb{N} \setminus \{0\} : e_i = \{v_i, v_{i+1}\} \in E \vee e_i = (v_i, v_{i+1}) \in V \times V\} \quad (5.5)$$

The sum of weights of edges and arches is the same for each designated Euler cycle.

The graphs representing the problem can be mixed graphs, where the edges represent two-way streets and the arcs represent one-way streets. Figure 5.3a shows a fragment of the map of the estate, where the graph's vertices are marked—in this case they are road intersections. The edges mark two-way streets between intersections. In the presented fragment of the estate, there is only one one-way road number 11, which leads from vertex 10 to 1. In Fig. 5.3b, which shows the graph of the estate, one-way street is represented by an arc, which is beginning at vertex 10 and ending at vertex 1.

5.2.2 Graph Extension

The next step is to expand the graph. Depending on the cycles sought, a complete graph or an Euler graph is determined. The complete graph is determined by building the matrix of minimum paths between each two vertices of the base graph. Table 5.1 presents such a matrix for our example housing estate. To obtain it, the Dijkstra algorithm was used. The obtained complete graph is presented in Fig. 5.5.

In the process of extending the primary graph to the Euler graph, the orientation of the arcs is taken into account. In Fig. 5.6 there is an Euler graph which was obtained by extending the base graph representing the estate. The extension was obtained by adding the edges drawn here with the coloured line.

Table 5.2 describes the edges of the obtained Euler graph. the first 23 edges belong to the base graph and the remaining eight to the extension of the graph. The edges extending this graph are selected in such a way that the obtained Euler cycle satisfies the condition for a minimum sum of the weights of the edges and arcs of the graph extension. Since the Euler cycle passes through all edges and arcs of the graph then their sum of weights is constant, and the final value of the objective function depends on the sum of the weights of edges and arcs added to the graph to build the Euler graph. An artificial immune system was used to determine the edges and arcs of the graph extension.

Table 5.1 Array of distances [km]

Vertices	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
1	0	220	273	547	452	552	600	260	130	31	231	466	606	370	500
2	220	0	53	222	365	465	482	622	517	418	218	187	163	283	382
3	273	53	0	180	725	825	873	533	403	304	165	134	110	643	773
4	444	224	169	0	95	195	365	516	413	314	114	83	59	179	276
5	548	328	275	95	0	100	270	452	322	579	209	178	202	82	181
6	637	417	364	195	100	0	170	321	451	507	307	276	254	180	81
7	702	482	429	365	270	170	0	340	439	733	326	295	319	199	100
8	260	480	533	516	421	321	340	0	130	229	429	667	643	370	240
9	130	350	403	417	322	422	439	130	0	99	299	537	513	240	339
10	31	251	304	473	616	550	569	229	99	0	200	438	414	534	469
11	330	110	165	114	209	309	326	429	299	200	0	31	55	127	226
12	407	187	134	83	178	278	295	435	537	438	31	0	24	96	195
13	286	165	110	59	154	254	319	459	360	255	55	24	0	120	219
14	457	237	290	177	82	180	199	370	240	327	127	96	236	0	99
15	500	382	773	276	181	81	100	240	339	469	226	195	219	99	0

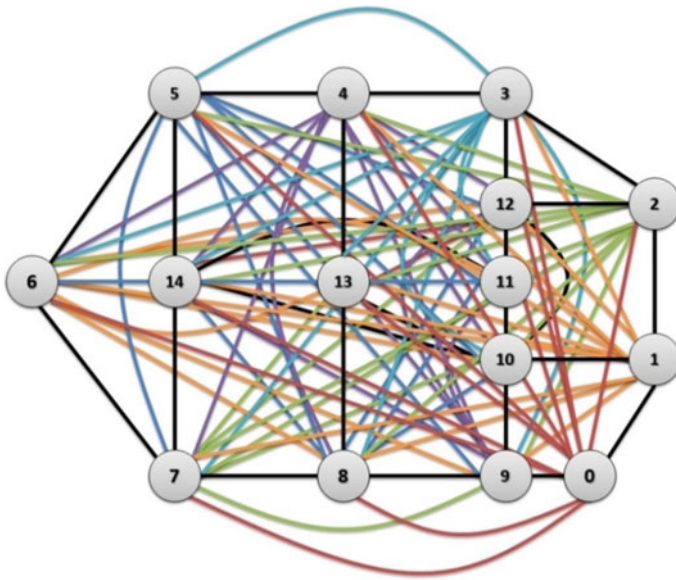


Fig. 5.5 A graph extension to a complete graph

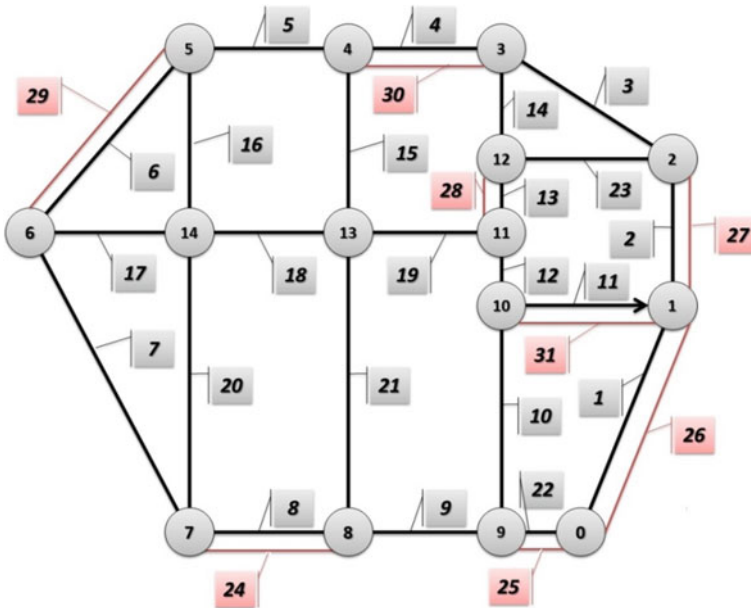


Fig. 5.6 Graph extension to the Euler graph

5.2.3 Determining the Cycles

Graph extensions enable the determination of appropriate cycles. In a complete graph, each cycle passing through all the vertices of the graph is a Hamiltonian cycle. Thus, each such cycle may be an acceptable solution in the problem of finding the Hamilton cycle with the minimum sum of the weights of edges or arcs. In an Euler graph, each cycle traversing all edges is an Eulerian cycle. In this case, for each Euler cycle, the sum of the edge and arc weights is minimal, which was taken care of while extending the problem graph to the Euler graph.

In Fig. 5.7, the numbered arrows show the sequence of passing along the edges and arcs of one of the Euler cycles. As you can see, the direction of the passing of the arc follows the orientation of the arc. The Hierholzer method is sufficient to determine such a cycle.

When ARP has various limitations, the task becomes more complicated. In the described methodology, both in the simple task of determining the Euler cycle, and in the more complicated case, artificial immune systems are used.

Table 5.2 A description of the edge of the Euler graph

<i>Base graph</i>			
No of edge	Initial vertex	Final vertex	Weight of edge
1	0	1	220
2	1	2	53
3	2	3	180
4	3	4	95
5	4	5	100
6	5	6	170
7	6	7	460
8	7	8	130
9	8	9	99
10	9	10	200
11	10	1	110
12	10	11	31
13	11	12	24
14	12	3	59
15	13	4	82
16	14	5	81
17	14	6	100
18	14	13	99
19	13	11	96
20	14	7	240
21	13	8	240
22	9	0	31
23	12	2	110
<i>Graph extension</i>			
24	7	8	130
25	9	0	31
26	0	1	220
27	1	2	53
28	11	12	24
29	5	6	170
30	4	3	95
31	10	1	110

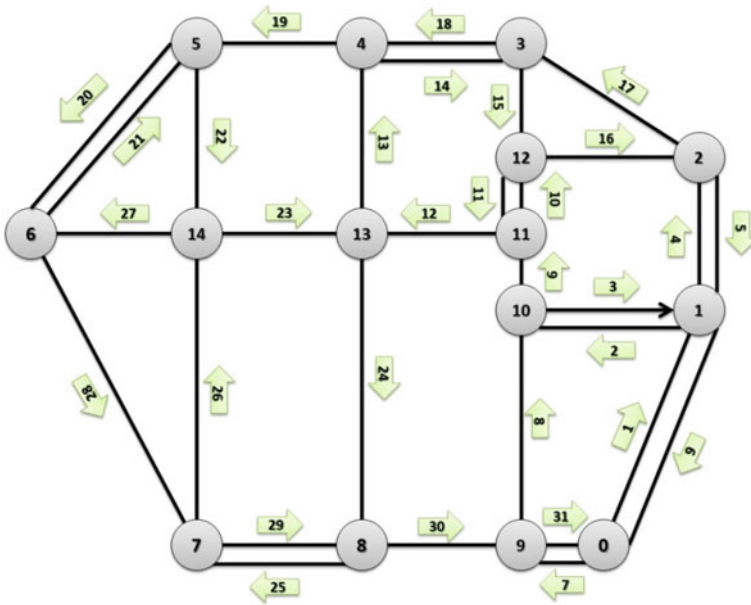


Fig. 5.7 Euler's cycle

5.2.4 Clustering

As the size of the tasks increases, the calculation slows down. Figure 5.8 shows the average time to obtain the best solution as a function of the number of vertices of the graph nodes for the case described in the article [35]. The clonal selection algorithm implemented for the TSP problems of the discussed methodology was used for the calculations. As you can see from the chart, calculations for a task represented by a

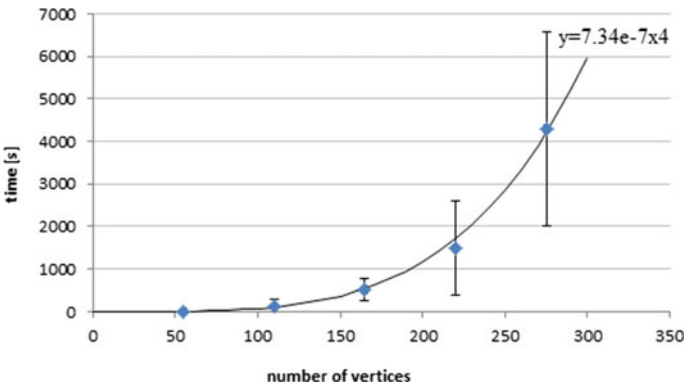


Fig. 5.8 Average time to obtain the best solution as a function of the number of vertices [35]

graph with 200 vertices take almost half an hour, and for a graph with 250 vertices—almost an hour. The computation time increases rapidly with the fourth power of the number of vertices in the graph.

Various methods are used to speed up TSP calculations. The efficiency of parallel computing in order to solve the traveling salesman problem for standard test problems was assessed in the article [3]. In large TSPs, to speed up the computation, the task is often split into smaller ones through clustering. The latter method was incorporated into the described methodology. After the graph is prepared, it should be assessed whether the task is too large, i.e. whether the graph has too many nodes. If it is, you should perform clustering, which will properly separate the graph into smaller ones. In accordance with the presented methodology, the clonal selection algorithm is used. The efficiency of clustering in the problems solved in accordance with the presented methodology is presented in articles [24, 34].

The article [24] presents the problem of deliveries from warehouses to customers with the use of a heterogeneous fleet of delivery vehicles, which differ, among others, in load capacity and fuel consumption. The calculations were carried out with the clonal selection method with clustering. In the first stage, customers were divided into groups due to the minimum road distances between them, the fuel consumption and with the limitation of the mass of goods due to the load capacity of the delivery truck. With such a limitation, a cluster is created that brings together only as many customers as the vehicle assigned to the cluster can handle. The adoption of the criterion of minimum road distances and minimum fuel consumption will result in the creation of clusters of customers located close to each other as a result of the calculations. In the second stage, route and time minimization with the Pareto optimality criterion was carried out.

The results of the calculations were compared with the optimization without initial clustering and with the two-step method, where the clustering was done with the Forgy algorithm and the solution of the salesman's task with the branch and bound method. The results of the calculations are presented in Table 5.3. The best results were obtained in one step using the clonal selection algorithm. But the results of the two-step method, in which clonal selection was applied, both, at the clustering stage and at the determination of the optimal path, are not much worse than in the first method.

In the second case described in [34], the results of the proposed two-step method were compared with optimization without initial clustering. The method of planning routes of delivery vehicles to the network of small self-service stores providing their customers with everyday products was presented. Each store could have a different order. The time for handling each point also can be different.

Calculations were carried out for 6 vehicles. The number of pallets delivered with the goods, the time of unloading the pallet and other activities related to delivery were done. For each of these vehicles, a route was determined and its length and travel time was calculated. All calculations were carried out several times. The table contains the best of the obtained results. The time to obtain results did not exceed a few minutes.

Table 5.3 Results

The method	Criteria for optimization	No of vehicle	Mass of goods	Length of road	Fuel consumption
			[kg]	[km]	[l]
Branch and bound method	Minimizing the length of road while taking into account vehicle capacity	5	2833	100	14
		4	4265	122	21
		1	6231	114	27
		1	6833	87	21
				423	422
Clustering with clonal selection	Minimizing the length of road and fuel consumption while taking into account vehicle capacity	3	3980	71	16
		1	7196	90	21
		4	4699	84	14
		2	4287	94	16
				339	338
Clonal selection	Minimizing the length of road while taking into account vehicle capacity	1	7363	103	25
		2	5849	124	27
		3	5390	74	13
		5	1560	28	4
				329	330

The division into clusters was carried out due to the constraints for the capacity of the car allocated to the cluster and due to the minimum distances between each two points in clusters, where the minimum distances have been calculated as the lengths of the shortest routes. The travel times between the points have not been included. The results are presented in Table 5.4. Table 5.5 contains the results of calculations using the direct method, that is without division into clusters and with clustering of collection points.

In the case of a one-step solution, delivery vans covered the 343.33 km route in the total travel and the service time corresponded to 20 h and 43 min. The route of an individual vans ranged from 53.85 km to 63 km, while the travel and the service time was in range from 1 h 14 min up to 3 h 57 min. Table 5.5 contains the results of calculations using the direct method, that is without division into clusters.

Calculations were carried out for 6 vehicles. For each of these vehicles, a route was determined and its length was calculated. The number of pallets delivered with the goods, the time of travel, unloading of the pallet and other activities related to delivery were done. All calculations were carried out several times. The table contains the best of the obtained results. The time to obtain results did not exceed a few minutes.

Table 5.4 Results of minimizing the length of the road of the vehicles in the clusters

No centroids	Delivery size (number of pallets)	The minimum distances of points in clusters	Service time	Results of minimizing the length of the road and the working time of the vehicle in the clusters			
				Solution A		Solution B	
				The length of the road	Travel time	The length of the road	Travel time
		[km]	[min]	[km]	[min]	[km]	[min]
4	11	11.9	79	50.9	66	50.9	66
13	20	18.9	157	62.4	101	62.9	100
17	20	6.0	143	51.1	76	51.2	73
18	19	11.0	125	64.6	76	64.6	76
20	18	3.9	121	53.8	68	53.8	68
29	20	16.0	157	54.8	88	54.9	86
$\Sigma =$	108	67.7	782	337.5	475	338.2	469

Table 5.5 Results: minimization of the length of the road and the working time of the vehicle using the one-step method

The vehicle number	Delivery size (number of pallets)	Travel and service time	The length of the road
		[min]	[km]
1	20	237	55.64
2	20	233	61.39
3	20	211	63.00
4	19	207	54.65
5	20	221	53.85
6	9	134	54.80
Σ	108	1243	343.33

In the case of calculations with clustering, the same solutions were obtained for both calculation criteria. And so, the delivery cars travelled a total distance of 337.5 km during and the travel and service time was equal to 20 h and 57 min. The route of individual cars ranged from 50.09 km to 64.6 km, and the journey time and service fluctuated from 2 h 25 min to 4 h 18 min. As it can be seen the results of the calculations are very similar. Perhaps this is due to the relatively small number of serviced points. However, the method with clustering gives hope for an effective application to solve really big problems of routing. In turn, the Pareto optimality method provides the entire set of optimal solutions and allows the final selection to be guided by other, unmeasurable criteria.

The effectiveness of the methodology and the computational algorithms implementing it have been tested on many examples and compared with other methods, which include both strict and heuristic methods as well as artificial intelligence

Table 5.6 Criteria, conditions and restrictions used in the methodology

Description of the problem	Criteria	Restrictions
TSP for a heterogeneous vans fleet [31, 32]	Minimum distance and travel time with taking into account variable traffic volumes	Vehicle load capacity, limitations on drivers' working time
TSP for a heterogeneous vans fleet [7, 33]	Minimum of the sum of roads of all fleet cars and the sum of travel times and the sum of customer service times, minimization of the number of vans	Limited time for customer service by every van of a fleet of vans
TSP for a heterogeneous vans fleet [34]	Minimum distance and minimum fuel consumption	Vehicle load capacity
TSP for arranging goods on shelves in a warehouse [26, 30]	Minimum distance from the picking area and minimum time to move goods from the shelves to the picking zone	
TSP in loading heterogeneous cuboidal parcels into cuboidal space - on a pallet, on a truck [27, 28]	Maximum loaded packages and minimum distance of the center of gravity of the loaded packages from the floor	
ARP in planning routes for garbage collection vehicles along the streets of housing estates [23, 29, 36]	Minimum distance	Vehicle load capacity

[21]. All tests favored artificial immune systems. In the optimization problems to be solved, the results were obtained which approximated the optimal solutions with high accuracy, sufficient in these problems.

Table 5.6 presents a list of publications with examples of applications of the methodology supported by clonal selection algorithms. The applicable criteria and limitations are listed.

5.3 Conclusions

The article presents the theoretical basis of the methodology of solving TSP and EP. Although the methods of solving both problems differ quite significantly, the procedure is similar. The original mathematical model of solving problems is part of this methodology. As is clear from the literature review, other authors in most cases they use to solve the TSP or ARP algebraic methods, eg. integer linear programming (ILP). Artificial intelligence methods are in such cases used to solve the obtained system of linear equations and inequalities. The approach presented here is more flexible because it is not limited in advance by a rigid algebraic model.

Graphs are used in a natural way to describe the created models. Basic theorems and methods of graph theory are used to transform graphs representing a problem to be solved into a complete graph in TSP or an Euler graph in EP, respectively. In the complete graph, each Hamiltonian cycle is an acceptable solution, and in the Euler graph it is each Euler cycle (in EP each Euler cycle has the same length and the optimization problem usually arises only when additional conditions and constraints are taken into account). The clonal selection of the artificial immune system is used to find a cycle for which the value of the adopted criterion is optimal and all limiting conditions are met.

With a different mathematical model, graphs became a natural language useful for describing the created models.

As part of the methodology, the problem of clustering was also addressed, which supports and solves large tasks. Articles [24, 34] show that the results obtained with the aid of clustering are not worse than those obtained without clustering. This topic will be continued by the author.

References

1. Akbari, V., Salman, F.S.: Multi-vehicle synchronized arc routing problem to restore post-disaster network connectivity. *Eur. J. Oper. Res.* **257**, 2 (2017), 625–640 (2017). <https://doi.org/10.1016/j.ejor.2016.07.043>
2. Ávila, T., Corberán, Á., Plana, I., Sanchis, J.M.: A branch-and-cut algorithm for the profitable windy rural postman problem. *Eur. J. Oper. Res.* **249**(3), 1092–1101 (2016). <https://doi.org/10.1016/j.ejor.2015.10.016>
3. Avcı, B., Aliabadi, D.E.: Parallelized neural network system for solving Euclidean traveling salesman problem. *Appl. Soft Comput.* **34**, 862–873 (2015). <https://doi.org/10.1016/j.asoc.2015.06.011>
4. Bellman, R.: Dynamic programming treatment of the travelling salesman problem. *J. Assoc. Comput. Mach.* **9**(1962), 61–63 (1962)
5. Caserta, M., Voß, S.: A hybrid algorithm for the DNA sequencing problem. *Discrete Appl. Math.* **163**, 87–99 (2014). <https://doi.org/10.1016/j.dam.2012.08.025>
6. Cerny, V.: A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm. *J. Optim. Theory Appl.* **45**(1985), 41–51 (1985)
7. Cieślak, M., Mrówczyńska, B.: Problem of medicines distribution on the example of pharmaceutical wholesale. In: Zawiański, S., Rysiński, J. (eds.) *Graph-Based Modelling in Engineering, Mechanisms and Machine Science*, vol. 42, pp. 51–65. Springer, Cham (2017)
8. Dussault, B., Golden, B., Groër, C., Wasil, E.: Plowing with precedence: a variant of the windy postman problem. *Computers Oper. Res.* **40**(4), 1047–1059 (2013). <https://doi.org/10.1016/j.cor.2012.10.013>
9. Dasgupta, D., Yu, S., Nino, F.: Recent advances in artificial immune systems: models and applications. *Appl. Soft Comput.* **11**(2), 1574–1587 (2011). <https://doi.org/10.1016/j.asoc.2010.08.024>
10. De Castro, L.N., Timmis, J.: *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer, Berlin (2002)
11. Eldos, T., Kanan, A., Aljumah, A.: Solving the printed circuit board drilling problem by ant colony optimization algorithm. In: Ao, S.I., Douglas, C., Grundfest, W.S., Burgstone, J. (eds.) *World Congress on Engineering and Computer Science (WCECS 2013)*, vol. I, pp. 584–588. International Association of Engineers (IAENG) (2013)

12. Graaff, A.J., Engelbrecht, A.P.: Clustering data in an uncertain environment using an artificial immune system. *Pattern Recogn. Lett.* **32**(2), 342–351 (2011). <https://doi.org/10.1016/j.patrec.2010.09.013>
13. Held, M., Karp, R.M.: A dynamic programming approach to sequencing problems. *J. Soc. Ind. Appl. Math.* **1**(10), 196–210 (1962)
14. Kirschstein, T., Bierwirth, C.: The selective traveling salesman problem with emission allocation rules. *OR Spectrum* **40**(1), 97–124 (2018). <https://doi.org/10.1007/s00291-017-0493-z>
15. Kolahan, F., Liang, M.: A tabu search approach to optimization of drilling operations. *Computers Ind. Eng.* **31**(1), 371–374 (1996). [https://doi.org/10.1016/0360-8352\(96\)00154-4](https://doi.org/10.1016/0360-8352(96)00154-4)
16. Laporte, G., Martello, S.: The selective travelling salesman problem. *Discrete Appl. Math.* **26**(2), 193–207 (1990). [https://doi.org/10.1016/0166-218X\(90\)90100-Q](https://doi.org/10.1016/0166-218X(90)90100-Q)
17. Lenstra, J.K., Kan, A.H.G.R.: Complexity of vehicle routing and scheduling problems. *Networks* **11**(2), 221–227 (1981). <https://doi.org/10.1002/net.3230110211>
18. Li, X., Leung, S.C.H., Tian, P.: A multistart adaptive memory-based tabu search algorithm for the heterogeneous fixed fleet open vehicle routing problem. *Expert Syst. Appl.* **39**(1), 365–374 (2012). <https://doi.org/10.1016/j.eswa.2011.07.025>
19. Mahi, M., Baykan, Ö.K., Kodaz, H.: A new hybrid method based on particle swarm optimization. In: *Ant colony optimization and 3-opt algorithms for traveling salesman problem*. *Appl. Soft Comput.* **30**, 484–490 (2015). <https://doi.org/10.1016/j.asoc.2015.01.068>
20. Masutti, T.A.S., de Castro, L.N.: Neuro-immune approach to solve routing problems. *Neurocomputing* **72**(10), 2189–2197. <https://doi.org/10.1016/j.neucom.2008.07.015>
21. Mestria, M.: New hybrid heuristic algorithm for the clustered traveling salesman problem. *Computers Ind. Eng.* **116**, 1–12 (2018). <https://doi.org/10.1016/j.cie.2017.12.018>
22. Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolutionary Programs*. Springer, Berlin (1996)
23. Mrówczyńska, B.: Route planning of separate waste collection on a small settlement. *Transp. Problems* **1**(9), 61–68 (2014)
24. Mrówczyńska, B.: Zastosowanie sztucznego systemu immunologicznego do rozwiązania wielokryterialnego problemu dystrybucji dostaw. *Problemy transportu w inżynierii logistyki – część 1.*, *Prace Naukowe Politechniki Warszawskiej*. Transport z.117. ISSN 1230, pp. 219–229 (2017)
25. Mrówczyńska, B.: Optimal distribution of sub-assemblies in stores of factory by evolutionary Algorithms. *Diagnostyka* **4**(44), 73–76 (2007)
26. Mrówczyńska, B.: Optimal goods distribution in supermarket's store by evolutionary algorithms. In: *Burczyński, T., Cholewa, W., Moczulski, W. (eds.) Recent Developments in Artificial Intelligence Methods. AI-METH 2007*, Gliwice, Poland, 7–9 November 2007, Silesian University of Technology. Department for Strength of Materials and Computational Mechanics. Department of Fundamentals of Machinery Design, Polish Association for Computational Mechanics. Gliwice, pp. 147–154 (2007)
27. Mrówczyńska, B.: An application of evolutionary and immune algorithms for the optimisation of packing a diversified set of packets on a pallet. *Problemy Eksploatacji* **4**(137–145), 2008 (2008)
28. Mrówczyńska, B.: A clonal selection algorithm for pallet loading problem. In: *Burczyński, T., Périaux, J. (eds.) Evolutionary and Deterministic Methods For Design, Optimization and Control. Applications to Industrial and Societal Problems*, © CIMNE, Barcelona, Spain, pp. 129–135 (2011)
29. Mrówczyńska, B.: Optimal routes scheduling for municipal waste disposal garbage trucks using evolutionary algorithm and artificial immune system. *Transp. Problems* **6**(4), 5–12 (2011)
30. Mrówczyńska, B., Sładkowski, A.: Rozmieszczenie zapasów w magazynie z uwzględnieniem czasu transportu. *Studia Ekonomiczne. Zeszyty Naukowe Wydziałowe nr 143. Uniwersytet Ekonomiczny w Katowicach*. Katowice, pp. 301–311 (2013)
31. Mrówczyńska, B.: Multicriteria vehicle routing problem solved by artificial immune system. *Transp. Problems* **10**(3), 141–152 (2015)

32. Mrówczyńska, B.: Comparison of Pareto efficiency and weighted objectives method to solve the multi-criteria vehicle routing problem using the artificial immune system. *Appl. Computer Sci.* **12**(4), 78–87 (2016)
33. Mrówczyńska, B., Cieśla, M.: Planning routes of vans in a catering company. In: ICLEEE 2017 International Conference of Logistic, Economics and Environmental Engineering. Maribor, Slovenia, pp. 66–70 (2017). ISBN 978-961-6672-11-5. http://www.vpsmb.net/images/ICLEE/Zbornik_ICLEEE_2017_ver_16_5_2017.pdf
34. Mrówczyńska, B.: Application of artificial immune systems for planning of beverage's delivery to network of retail shops. In: ZIRP 2017, International Conference on Traffic Development, Logistics & Sustainable Transport, Croatia, Opatija, 1–2 June 2017, pp. 223–229 (2017). ISBN 978-953-243-090-5. <http://www.fpz.unizg.hr/zirp-1st/assets/files/ZIRP-2017-conference-proceedings.pdf>
35. Mrówczyńska, B., Król, A., Czech, P.: Artificial immune system in planning deliveries in a short time. *Bull. Polish Acad. Sci. Tech. Sci.* **67**(5), 969–980 (2019). 10.244 25/bpas.2019.126630s.
36. Nowakowski, P., Mrówczyńska, B.: Towards sustainable WEEE collection and transportation methods in circular economy—comparative study for rural and urban settlements. *Resour. Conserv. Recycling* **135**, 93–107 (2018). <https://doi.org/10.1016/j.resconrec.2017.12.016>
37. Nowakowski, P., Król, A., Mrówczyńska, B.: Supporting mobile WEEE collection on demand: a method for multi-criteria vehicle routing, loading and cost optimisation. *Waste Manage.* **69**, 377–392 (2010). <https://doi.org/10.1016/j.wasman.2017.07.045>
38. Padberg, M., Rinaldi, G.: A Branch-and-Cut Algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Rev.* **33**(1), 60–100 (1991). <https://doi.org/10.1137/1033004>
39. Park, J., Tae, H., Kim, B.-I.: A post-improvement procedure for the mixed load school bus routing problem. *Eur. J. Oper. Res.* **217**(1), 204–213 (2012). <https://doi.org/10.1016/j.ejor.2011.08.022>
40. Pevzner, P.A., Lipshutz, R.J.: Towards DNA sequencing chips. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 841 LNCS, pp. 143–158
41. Roberti, R., Wen, M.: The electric traveling salesman problem with time windows. *Transport. Res. Part E: Logistics Transport. Rev.* **89**, 32–52 (2016). <https://doi.org/10.1016/j.tre.2016.01.010>
42. Salari, M., Toth, P., Tramontani, A.: An ILP improvement procedure for the open vehicle routing problem. *Computers Oper. Res.* **37**(12), 2106–2120 (2010). <https://doi.org/10.1016/j.cor.2010.02.010>
43. Schittekat, P., Kinable, J., Sörensen, K., Sevaux, M., Spiessma, F., Springael, J.: A metaheuristic for the school bus routing problem with bus stop selection. *Eur. J. Oper. Res.* **229**(2), 518–528 (2013). <https://doi.org/10.1016/j.ejor.2013.02.025>
44. Shao, L., Bai, Y., Qiu, Y., Du, Z.: Particle Swarm Optimization algorithm based on semantic relations and its engineering applications. *Syst. Eng. Procedia* **5**(2012), 222–227 (2012). <https://doi.org/10.1016/j.sepro.2012.04.035>
45. Zhang, T., Ke, L., Li, J., Li, J., Huang, J., Li, Z.: Metaheuristics for the tabu clustered traveling salesman problem. *Comput. Oper. Res.* **89**, 1–12 (2018). <https://doi.org/10.1016/j.cor.2017.07.008>

Chapter 6

Using Graphs for Modeling and Solving Cyclic Flow Shop with Waiting Time Constraints



Czesław Smutnicki  and Wojciech Bożejko 

Abstract The chapter deals with the permutation flow-shop scheduling problem with time couplings in the form of an upper/lower bound on waiting-time between operations in the job and the minimal cycle time criterion. We propose certain decomposition of the problem leading to two particular sub-problems, namely “to find the minimal cycle time for a fixed job processing order” and “to find the best job processing order”. A variety of graphs have been used to solve the former sub-case in the sequential and parallel computing environments. For the latter sub-case we recommend a meta-heuristic with some special properties derived from these graphs.

Keywords Scheduling · Cyclic flow-shop · Waiting times · Graphs · Algorithm · Time couplings

6.1 Introduction

A deterministic scheduling theory is frequently used for modeling and optimization of workflow in production systems with robots, see for example review in Dolgui et al. [9]. These models perform well for human-less systems, since robots usually works in almost deterministic way. The existing taxonomy of problems in the scheduling field is set to cover, as this is possible, practical instances and takes into account, among others, architecture of production/service units, classes of the schedules, additional constraints and optimization criteria. The goal of this classification is to fit for each practical case to the appropriate model and to recommend a powerful solution method, see for example Bocewicz et al. [1, 2]. Among the schedule

C. Smutnicki (✉)

Faculty of Electronics, Department of Computer Engineering, Wrocław University of Science and Technology, Wybrzeże Wyspińskiego 27, 50-370 Wrocław, Poland
e-mail: czeslaw.smutnicki@pwr.edu.pl

W. Bożejko

Faculty of Electronics, Department of Control Systems and Mechatronics, Wrocław University of Science and Technology, Wrocław, Poland

© Springer Nature Switzerland AG 2022

S. Zawiślak and J. Rysiński (eds.), *Graph-Based Modelling in Science, Technology and Art*, Mechanisms and Machine Science 107,
https://doi.org/10.1007/978-3-030-76787-7_6

105

classes appeared in the scheduling theory, we use only two of them in this chapter: (1) non-delay schedule, which means the single execution of the job set, see e.g. definition in Pinedo [21] and (2) cyclic schedule, which means repeatable execution of the job set, see e.g. Brucker and Kampmeyer [5, 6]. The former one is treated as auxiliary for the latter. The permutation flow-shop scheduling problem appears in the literature quite frequently, see for example the survey in Ruiz and Maroto [23]. A few real implementations were presented till now in the literature, e.g. in the chemical industry, Rajendran [22], food industry, Hall and Sriskandarajah [15] and construction automation, Grabowski and Pempera [12].

We consider the cyclic flow-shop scheduling problem in which an assortment of products (a certain fixed mixture of products) is delivered to the output by repeating identical, but mixed in the composition production series, each of which called a Minimal Part Set (MPS). The problem has also included “limited waiting time” constraints between successive operations of a job, which is a generalization of the case known in the literature as “no-wait flow-shop”. Both mentioned technologies are used to eliminate or reduce the queue before a stage. Gilmore and Gomory [11] analysed a non-delay schedule for the flow-shop problem with “no-wait” constraints for two machines. This result can be easily extended to cyclic schedule in case of two machines. For more than two machines, problem has a relation to the Traveling Salesman Problem (TSP), thus algorithms for asymmetric TSP can be recommended. Other variants of the flow-shop with “no-wait” constraints have been considered in Grabowski and Pempera [14]. We also refer to algorithms dedicated to flow-shop [16] and hybrid flow-shop [8, 24, 28].

The remainder of the chapter is as follows. Section 6.2 gives a mathematical model with reference to a linear programming task to find the minimum cycle time for a given job processing order. Alternative solutions algorithms with the use of graphs are presented in the Sect. 6.3. Section 6.4 describes methods of parallel determination of the minimum cycle time (Table 6.1).

6.2 Mathematical Model

The conventional flow-shop scheduling problem refers to the machine park consisting of m service stages defined by the set $M = \{1, \dots, m\}$, which are used to perform n jobs from the set $N = \{1, 2, \dots, n\}$. Each job flows through machines $1, 2, \dots, m$ in the identical technological order, however the order of loading the jobs into the manufacturing system is changeable. In the paper we consider two cases: (a) n jobs are loaded to the system once, in some order, providing so called non-delay schedule considered next in the context of regular scheduling criteria; (b) n jobs are performed in the repetitive way, duplicating the loading processing order repeatedly, providing so called cyclic schedule, which is actually an irregular scheduling criteria. Processing time of a job $j \in N$ on machine $i \in M$ is called an operation (i, j) and has the duration $p_{i,j} > 0$. Time event constraint means that the time of waiting between processing of successive operation of a job has pre-defined lower $(a_{i,j})$ and upper $(b_{i,j})$

Table 6.1 List of symbols

data	meaning
$M = \{1, 2, \dots, m\}$	set of machines
$N = \{1, 2, \dots, n\}$	set of jobs
$O = M \times N$	set of operations
p_{ij}	processing time of operation (i, j)
a_{ij}	lower bound on waiting time $(i, j) \rightarrow (i + 1, j)$
b_{ij}	upper bound on waiting time $(i, j) \rightarrow (i + 1, j)$
variables	meaning
$\pi = (\pi(1), \dots, \pi(n))$	processing order; permutation on N
$S = [S_{ij}]_{m \times n}$	schedule (matrix)
T	cycle time
criteria	meaning
$C_{\max}(\pi)$	makespan for processing order π
$T(\pi)$	minimal cycle time for processing order π
auxiliary	meaning
$x = 1, 2, \dots$	index of cycles (MPSes)
i, j	machine index; job index
$(i, j) \in O$	operation i of job j
$o = m \cdot n$	number of operations
$G(\pi)$	planar graph for π
$H(\pi)$	wrapped-around-cylinder graph for π
$G^*(\pi)$	chain of $m + 1$ graphs $G(\pi)$
i^x	operation (pair) in cycle x in $G^*(\pi)$
$U_{i^x, j^y} = (u_0, \dots, u_v)$	path $i^x \rightarrow j^y$
$L(i^x, j^y)$	path length
$B_{a:b}$	block
$B_{a+1:b-1}$	internal block

bounds. Processing jobs on the machines cannot be suspended. Each machine can process at most one job at a time, and each job can be processed on only one machine at a time.

Jobs enter the system in the order established by a permutation $\pi = (\pi(1), \dots, \pi(n))$ on the set N , which will be called next the “loading sequence” on the system entry. Each π implies the schedule $S = [S_{i,j}]_{m \times n}$ (defined as starting times of all operations). Considering case (a) we refer to the most popular criterion called the makespan written as

$$C_{\max}(\pi) = \max_{1 \leq i \leq m} \max_{1 \leq j \leq n} (S_{i,j} + p_{i,j}). \quad (6.1)$$

Note that values of other regular criteria can be calculated from a non-delay schedule S . For the case (b) we refer to the minimal cycle time $T(\pi)$, which is calculated in more complex manner, described in the sequel. The overall aim is to find permutation π^* with minimal criterion value

$$K(\pi^*) = \min_{\pi \in \Pi} K(\pi), \quad (6.2)$$

where Π is the set of permutations and $K(\pi)$ is either $C_{\max}(\pi)$ or $T(\pi)$. Let us consider the problem of finding S in cases (a) and (b) for the given π in detail.

The manufacturing system repeats processing of jobs from N in the form of successive MPSes indexed by $x = 1, 2, \dots$. Case (a) corresponds to $x = 1$, case (b) to infinite sequence $x = 1, 2, \dots$. We assume in case (b) that the job processing order π in each cycle remains the same. The schedule in successive cycles is periodical so can be obtained by shifting on the time axis the schedule S by a certain number of periods T . For the given π , the minimal period of time, for which technological and sequential constraints inside the cycle and between cycles have been satisfied is $T(\pi)$. We are looking in case (b) for the synchronous periodical schedule with the property

$$S_{i,j}^x = S_{i,j} + (x - 1)T, \quad i = 1, \dots, m, j = 1, \dots, n, x = 1, \dots \quad (6.3)$$

where $S_{i,j}^x$ is the start time of job $j \in N$ on machine i in x th cycle and $T \geq T(\pi)$. From (6.3) it is enough to set single start time S , since all the remain S^x for $x = 1, 2, 3, \dots$ follow immediately from this S . Hence, we define cyclic schedule S as this satisfying the following constraints

$$S_{i,j} \geq 0, \quad i = 1, \dots, m, j = 1, \dots, n, \quad (6.4)$$

$$S_{i,\pi(j)} \geq S_{i,\pi(j-1)} + p_{i,\pi(j-1)}, \quad i = 1, \dots, m, j = 2, \dots, n, \quad (6.5)$$

$$a_{i,j} \leq S_{i+1,j} - (S_{i,j} + p_{i,j}) \leq b_{i,j}, \quad i = 2, \dots, m, j = 1, \dots, n, \quad (6.6)$$

$$S_{i,\pi(n)} + p_{i,\pi(n)} \leq S_{i,\pi(1)} + T \quad i = 1, \dots, m \quad (6.7)$$

Constraint (6.4) forces the schedule to be on the positive part of the time axis. Constraint (6.5) means that starting of successive operation on a machine is possible upon completion of the preceding operation on this machine. The last constraint fixes bounds on the waiting time expected between successive technological operations. Notice, assuming $a_{i,j} = 0 = b_{i,j}$ we get the well-known flow-shop scheduling problem with “no-wait” constraints. Value $a_{i,j}$ can be also perceived as the transport time between successive operations of a job. Minimal cycle time $T(\pi)$ for fixed job processing order π can be found using a few methods. We discuss them in detail. The

first approach follows immediately from the linear programming (LP) task, namely

$$T(\pi) = \min_s T \quad (6.8)$$

under constraints (6.4) – (6.7). The task has $mn + 1$ non-negative variables and $2mn - n$ constraints. Although this task is a polynomial-time method, it seems to be numerically too expensive, since usually it is called repeatedly. Among other approaches, which can be adopted to find the cyclic schedule, we have found: mixed integer linear programming, [6]; network flows, [18]; cycles in graphs, [17]. Each of these technologies provides a non-polynomial algorithm to find $T(\pi)$, has various running time and requires specific modelling fashion. In this chapter we provide a few polynomial-time methods to find $T(\pi)$, based on specific graphs defined for the problem, see [20] for other implementations of this idea.

6.3 Graph Models

In this section we will introduce three various graphs (networks of operations) applied further to the formulation of special properties of the problem. These graphs refer to a few common notions defined in the sequel. Each graph is defined as the operational network presented in the AoN (Activity-on-Node) style. Node $(i, j) \in O = M \times N$ in the graph corresponds to the operation of job $j \in N$ performed on machine $i \in M$. The node (i, j) has weight $p_{i,j}$ and has assigned label $S_{i,j}$ which means the event “start time of the operation (i, j) ”. Any constraint between time events $S_{i,j}$ and $S_{k,l}$, for some (i, j) and (k, l) , written generally as

$$S_{i,j} + p_{i,j} + c_{i,j,k,l} \leq S_{k,l} \quad (6.9)$$

is represented by the arc $(i, j) \rightarrow (k, l)$ with the weight $c_{i,j,k,l}$.

In order to build suitable graphs, we need to assign inequalities (6.4)–(6.7) to particular arcs in the network. Actually, inequality (6.5) re-written in the form of

$$S_{i,\pi(j-1)} + p_{i,\pi(j-1)} \leq S_{i,\pi(j)}, \quad i = 1, \dots, m, \quad j = 2, \dots, n \quad (6.10)$$

can be expressed by the arc

$$(i, \pi(j-1)) \rightarrow (i, \pi(j)) \quad (6.11)$$

with the weight $c_{i,\pi(j-1),i,\pi(j)} = 0$. Equation (6.6) can be transformed into two independent inequalities (6.12) and (6.13), namely

$$S_{i+1,j} - S_{i,j} - p_{i,j} \geq a_{i,j}, \quad i = 2, \dots, m, \quad j = 1, \dots, n, \quad (6.12)$$

$$S_{i+1,j} - S_{i,j} - p_{i,j} \leq b_{i,j}, \quad i = 2, \dots, m, \quad j = 1, \dots, n, \quad (6.13)$$

To identify arcs following from (6.12) and (6.13) we have to convert them to the form of (6.9). After a few simple mathematical transformations of (6.12) we have

$$S_{i,j} + p_{i,j} + a_{i,j} \leq S_{i+1,j}, \quad i = 2, \dots, m, \quad j = 1, \dots, n, \quad (6.14)$$

which corresponds to the arc

$$(i, j) \rightarrow (i + 1, j) \quad (6.15)$$

with the weight $c_{i,j,i+1,j} = a_{i,j}$. Next, from (6.13) we obtain

$$S_{i+1,j} + p_{i+1,j} + (-p_{i,j} - p_{i+1,j} - b_{i,j}) \leq S_{i,j}, \quad i = 2, \dots, m, \quad j = 1, \dots, n, \quad (6.16)$$

which corresponds to the arc

$$(i + 1, j) \rightarrow (i, j) \quad (6.17)$$

with the weight $c_{i+1,j,i,j} = -p_{i,j} - p_{i+1,j} - b_{i,j}$. Inequality (6.7) converted to

$$S_{i,\pi(n)} + p_{i,\pi(n)} - T \leq S_{i,\pi(1)}, \quad i = 1, \dots, m. \quad (6.18)$$

generates the arc

$$(i, \pi(n)) \rightarrow (i, \pi(1)) \quad (6.19)$$

with the weight $c_{i,\pi(n),i,\pi(1)} = -T$.

6.3.1 Non-delay Schedule

This planar graph is recommended for processing single job set N , see Fig. 6.5 (left) for the given job processing order π . It allows us to find a non-delay schedule S satisfying constraints (6.4)–(6.6), but not (6.7). It is suitable for finding the makespan $C_{\max}(\pi)$ or other regular criterion value for the given processing order π . The graph has the form

$$G(\pi) = (O, E(\pi)), \quad (6.20)$$

where $O = M \times N$ is the set of nodes and $E(\pi) \subset O \times O$ is the set of arcs

$$E(\pi) = P \cup R \cup Q(\pi). \quad (6.21)$$

Arcs from P refer to inequality (6.14), have the form of (6.15), represent "technological" constraints associated with lower bounds on "waiting-time"

$$P = \bigcup_{i=1}^{m-1} \bigcup_{j=1}^n \{(i, j), (i+1, j)\}. \quad (6.22)$$

The arc $((i, j), (i+1, j)) \in P$ has weight $a_{i,j}$. Arcs from R refer to inequality (6.16), have the form of (6.17), can be also considered as “technological” and express upper bounds on “waiting-time”

$$R = \bigcup_{i=1}^{m-1} \bigcup_{j=1}^n \{(i+1, j), (i, j)\}. \quad (6.23)$$

The arc $((i+1, j), (i, j)) \in R$ has weight $-p_{i,j} - p_{i+1,j} - b_{i,j}$. Arcs from $Q(\pi)$ refer to inequality (6.10), have the form of (6.11), express “processing order” constraints and have the weight zero,

$$Q(\pi) = \bigcup_{i=1}^m \bigcup_{j=1}^{n-1} \{(i, \pi(j-1)), (i, \pi(j))\}. \quad (6.24)$$

We call schedule S feasible for this π if labels $S_{i,j}$ assigned to nodes $(i, j) \in O$ satisfy appropriate constraints for all $((i, j), (k, l)) \in E(\pi)$. Note that although $G(\pi)$ has graph cycles however it does not contain cycles of the length greater than zero. This means that the feasible S always exists. To find S we use the labeling algorithm, called next LA, applied previously in the paper [25] for a directed graph with graph cycles, see Fig. 6.1. LA checks for each arc $((i, j), (k, l)) \in E(\pi)$ whether the inequality $S_{k,l} \geq S_{i,j} + p_{i,j} + c_{i,j,k,l}$ is satisfied.

If the answer is “not satisfied”, LA adjusts $S_{k,l}$ to fulfil the constraint. Variable tag is used to detect the early end of the adjustment process. LA can be perceived as

```

Parameter:  $\pi$ 
for  $(i, j) \in O$  do  $S_{i,j} = 0$ ;
for  $v = 1$  to  $m \cdot n - 1$  do
   $tag = 0$ ;
  for  $((i, j), (k, l)) \in E(\pi)$  do
    if  $S_{k,l} < S_{i,j} + p_{i,j} + c_{i,j,k,l}$  then
      {  $S_{k,l} = S_{i,j} + p_{i,j} + c_{i,j,k,l}$ ;  $tag = 1$ ; }
    end if;
  end for  $((i, j), (k, l))$ ;
  if not  $tag$  then return;
end for  $v$ ;

```

Fig. 6.1 Algorithm LA

```

Parameter:  $\pi$ 
 $k = \pi(1); S_{1,k} = 0;$ 
for  $i = 1, \dots, m-1$  do  $S_{i+1,k} = S_{i,k} + p_{i,k} + a_{i,k};$ 
for  $j = 2, \dots, n$  do
 $k = \pi(j); l = \pi(j-1); S_{1,k} = S_{1,l} + p_{1,l};$ 
for  $i = 2, \dots, m$  do
 $S_{i,k} = \max\{S_{i,l} + p_{i,l}, S_{i-1,k} + p_{i-1,k} + a_{i-1,k}\}$ 
end for  $i;$ 
for  $i = m, \dots, 2$  do
 $S_{i-1,k} = \max\{S_{i-1,k}, S_{i,k} - p_{i-1,k} - b_{i-1,k}\}$ 
end for  $i;$ 
end for  $j;$ 

```

Fig. 6.2 Algorithm LA⁺

a far analogy to the well-known Bellman-Ford algorithm dedicated for the shortest paths from the single source.

Property 1 *For the given π algorithm LA provides non-delay schedule S in the time $\Omega(m \cdot n)$ and $O(m^2 \cdot n^2)$.*

Apart LA, we propose more advanced labelling algorithm LA⁺ with computational complexity considerably smaller than LA. Observe that $G(\pi)$ can be decomposed into a sequence of “strips” unscrambled along job processing order π , see Fig. 6.5 (left). Strip $\pi(i)$ has sequential predecessors in strips $\pi(1), \pi(2), \dots, \pi(i-1)$ and technological predecessors only inside strip $\pi(i)$. This allows us to calculate S strip-by-strip, down-and-up, according to the constraints (6.10), (6.14), (6.16), see the pseudo-code of LA⁺ in Fig. 6.2. Indeed, the first operation of the job $k = \pi(j)$, has no technological predecessors, so the substitution $S_{1,k} = S_{1,l} + p_{1,l}$ is clear. Next, formula $S_{i,k} = \max\{S_{i,l} + p_{i,l}, S_{i-1,k} + p_{i-1,k} + a_{i-1,k}\}$ in the former loop for $i = 2, \dots, m$ ensures (6.10), (6.14) but not necessarily (6.16). Checking (6.16) and adjusting $S_{i-1,k}$ is performed in the later loop for $i = m, \dots, 2$. It can be verified that LA⁺ ensures the feasibility of (6.10), (6.16), (6.14).

Property 2 *For the given π algorithm LA⁺ provides non-delay schedule S in the time $\Theta(m \cdot n)$.*

Besides the theoretical analysis of the computational complexity, one expects also experimental comparisons between LA and LA⁺. Using 80 flow-shop instances with $m \cdot n \leq 1,000$ from the benchmark set of [27] we find that the ratio of the running time of LA to the running time of LA⁺ is on average $0.6 \dots 0.7 \cdot m \cdot n$. It means, among others, that already for medium size instances $n = 50, m = 10$, LA⁺ is approximately 300 times faster than LA. It also means that the complexity of LA is closer to $O(m^2 \cdot n^2)$ than to $\Omega(m \cdot n)$.

Properties 1 and 2 allow us to find efficiently the goal function value for π for the majority of regular scheduling criteria, for example the makespan, mean flow time. Prospective benefits for the minimal cycle time criterion derive from the following property, the proof is obvious.

Property 3 $T(\pi) \leq C_{\max}(\pi)$, where $C_{\max}(\pi)$ is the makespan found for $G(\pi)$.

6.3.2 Cyclic Schedule

We introduce graph $H(\pi)$ to find the cyclic schedule S (for the given π and T) satisfying constraints (6.4)–(6.7) converted to the equivalent form (6.10), (6.14), (6.16), (6.18), if only such schedule exists. It vicariously provides an alternative method of finding $T(\pi)$, presented at the end of this subsection. The graph has the form of

$$H(\pi) = (O, E(\pi) \cup F(\pi)), \quad (6.25)$$

where O and $E(\pi)$ have been already defined for $G(\pi)$ (graph $G(\pi)$ fulfills constraints (6.10), (6.14), (6.16)), whereas set of arcs

$$F(\pi) = \bigcup_{i=1}^m \{(i, \pi(n)), (i, \pi(1))\}, \quad (6.26)$$

represents constraint (6.18). Each arc has the form (6.19) and the negative weight $-T$. $H(\pi)$ is a non-planar graph, can be drawn as an extension of $G(\pi)$ wrapped around a cylinder, see Fig. 6.5 (right), and can be perceived as a graph with unknown value of the parameter T . The best way is to set $T = T(\pi)$, but actually we do not know $T(\pi)$. Let us assign labels $S_{i,j}$ to nodes in $H(\pi)$ interpreted as the start time of an operation $(i, j) \in O$, and discuss the relation between T and S .

From (6.8) it follows that $T(\pi)$ is the minimal value of T for which the feasible schedule S under constraints (6.4) – (6.7) still exists. It means that: (a) if $T \geq T(\pi)$ then the feasible S can be found, (b) if $T < T(\pi)$ no feasible S exists. Concerning the graph in case (b) we conclude that $H(\pi)$ contains a graph cycle with the length greater than zero because of too small value of T .

Considering the broader context of using graphs, $H(\pi)$ has at least two possible applications. The former is obvious and consists in finding the schedule S for $T = T(\pi)$ with known in advance $T(\pi)$. This needs a single run of, for example LA, applied to $H(\pi)$ and this T . The latter is to find $T(\pi)$ through the identification of the border value between regions of feasible and unfeasible S . This needs multiple runs of a modified LA for variable values of T . We present hereafter more details about the latter idea, including an extension of LA.

At first, we have equipped LA with graph cycle detector used in case (b), see extended version of LA+E in Fig. 6.3. After the “end for v” loop LA+E checks additionally whether labels S have been set in the feasible manner. If not, ERR indicates the lack of feasibility. Based on LA+E we have designed the following bi-partition method. Let us assume that the minimal cycle time $T(\pi)$ is located in an interval, it means $T(\pi) \in [T_*, T^*]$, where T_* is a lower bound, T^* is an upper bound of the interval. One expects that S found for T^* is feasible, but for T_* is unfeasible. Next we run LA+E for trial value $T = (T_* + T^*)/2$. If for this T algorithm LA+E returns ERR then $T(\pi) \in [T, T^*]$; otherwise, if returns OK then $T(\pi) \in [T_*, T]$. The partition is repeated until a stopping criteria will be met, for example $T^* - T_* \leq \epsilon$, for certain small value ϵ . From the formulation of LP task it follows that the num-


```

Parameter:  $\pi$ 
  for  $(i, j) \in O$  do  $S_{i,j} = 0$ ;
  for  $v = 1$  to  $m \cdot n - 1$  do
     $tag = 0$ ;
    for  $((i, j), (k, l)) \in E(\pi) \cup F(\pi)$  do
      if  $S_{k,l} < S_{i,j} + p_{i,j} + c_{i,j,k,l}$  then
        {  $S_{k,l} = S_{i,j} + p_{i,j} + c_{i,j,k,l}$ ;  $tag = 1$ ; }
      end if;
    end for  $((i, j), (k, l))$ ;
    if not  $tag$  then return OK;
  end for  $v$ ;
  for  $((i, j), (k, l)) \in E(\pi) \cup F(\pi)$  do
    if  $S_{k,l} < S_{i,j} + p_{i,j} + c_{i,j,k,l}$  then return ERR;
  return OK;

```

Fig. 6.3 Algorithm LA+E

ber of LA+E calls in the bi-partition method until stop is unknown since $T(\pi)$ does not need to be integer. Notice, if ERR occurs, then the computational complexity of single run of LA+E is $\Theta(m^2 \cdot n^2)$, otherwise is $O(m^2 \cdot n^2)$ and $\Omega(m \cdot n)$. Therefore, finding $T(\pi)$ by a bi-partition method has the computational complexity $O(n^2 \cdot m^2 \cdot s)$, where s is the number of LA+E calls. Therefore, the bi-partition method with LA+E provides the algorithm of finding $T(\pi)$ with the pseudo-polynomial computational complexity. The method although theoretically interesting, might not be computationally attractive comparing to those presented in the next subsection.

One can ask whether algorithm LA⁺ significantly faster than LA in the non-delay scheduling case has the similar dominance also for $H(\pi)$. The improvement from LA+E to LA⁺+E remains true, however the potential profits are lesser comparing to LA+E because of the structure of paths in $H(\pi)$. Indeed, a graph cycle in $H(\pi)$ can go around the cylinder at most m times, which means that analyse of strips has to be repeated m times. The appropriate pseudo-code of LA⁺+E is shown in Fig. 6.4. To check feasibility in LA⁺+E we use the symbol $V \leftarrow E$, where V is a variable and E is an expression: if $V = E$ than do nothing, otherwise if $V < E$ than set $V = E$ and $tag = 1$.

The computational complexity of single run of LA⁺+E is $\Omega(m \cdot n)$ and $O(m^2 \cdot n)$. To complete considerations of this subsection we provide the method of setting initial values T_* and T^* for the bi-partition method. From Property 3 we obtain immediately $T^* = C_{\max}(\pi)$. On the other hand, from (6.7) we have for $i = 1, \dots, m$

$$T \geq S_{i,\pi(n)} + p_{i,\pi(n)} - S_{i,\pi(1)} \geq d_{i,\pi(1),i,\pi(n)} \quad (6.27)$$

```

Parameter:  $\pi$ 
  for  $i = 1 \dots m$  do for  $j = 1 \dots n$  do  $S_{i,j} = 0$ ;
  for  $v = 1, \dots, m$  do
     $tag = 0$ ;
     $k = \pi(1)$ ;  $l = \pi(n)$ ;  $S_{1,k} \leftarrow \max\{S_{1,k}, S_{1,l} + p_{1,l} - T\}$ ;
    for  $i = 2, \dots, m$  do
       $S_{i,k} \leftarrow \max\{S_{i,k}, S_{i-1,k} + p_{i-1,k} + a_{i-1,k}, S_{i,l} + p_{i,l} - T\}$ 
    end for  $i$ ;
    for  $i = m, \dots, 2$  do
       $S_{i-1,k} \leftarrow \max\{S_{i-1,k}, S_{i,k} - p_{i-1,k} - b_{i-1,k}\}$ 
    end for  $i$ ;
    for  $j = 2, \dots, n$  do
       $k = \pi(j)$ ;  $l = \pi(j-1)$ ;  $S_{1,k} \leftarrow \max\{S_{1,k}, S_{1,l} + p_{1,l}\}$ ;
      for  $i = 2, \dots, m$  do
         $S_{i,k} \leftarrow \max\{S_{i,k}, S_{i,l} + p_{i,l}, S_{i-1,k} + p_{i-1,k} + a_{i-1,k}\}$ 
      end for  $i$ ;
      for  $i = m, \dots, 2$  do
         $S_{i-1,k} \leftarrow \max\{S_{i-1,k}, S_{i,k} - p_{i-1,k} - b_{i-1,k}\}$ 
      end for  $i$ ;
    end for  $j$ ;
    if not  $tag$  then return OK;
  end for  $v$ ;
  if  $tag$  then return ERR;
return OK;

```

Fig. 6.4 Algorithm $LA^+ + E$

where $d_{i,\pi(1),i,\pi(n)}$ is the length of the longest path between start of node $(i, \pi(1))$ and end of node $(i, \pi(n))$ in $H(\pi)$. This finding allow us to set

$$T_* = \max_{1 \leq i \leq m} d_{i,\pi(1),i,\pi(n)}. \quad (6.28)$$

Appropriate algorithm of finding $d_{i,\pi(1),i,\pi(n)}$, $i = 1, \dots, m$, can be obtained by a slight modification of LA to run m times from the graph sources $(i, \pi(1))$, $i = 1, 2, \dots, m$. It has the computational complexity $O(m^2 \cdot n)$. Notice, in a general case (6.28) is not a tight bound (Fig. 6.5).

6.3.3 Chain of Graphs

$T(\pi)$ can be calculated precisely by using so-called chain graph $G^*(\pi)$. Briefly speaking, $G^*(\pi)$ is the concatenation of $m + 1$ identical graphs $G(\pi)$, see Fig. 6.6, to represent the sequence of $m + 1$ successive cycles indexed by $x = 1, 2, \dots, m + 1$. The concatenation has been made with the help of additional arcs going from each last operation on a machine in some cycle to the first operation on this machine

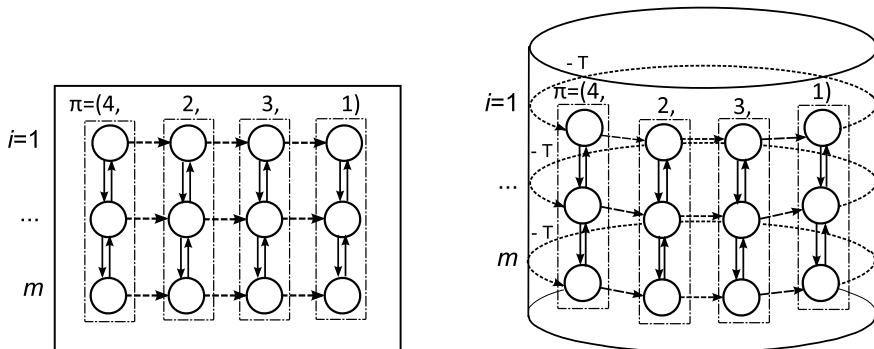


Fig. 6.5 Illustrative graph $G(\pi)$ for LA and strips used in LA* (left). Wrap-around-cylinder graph $H(\pi)$ for this instance (right)

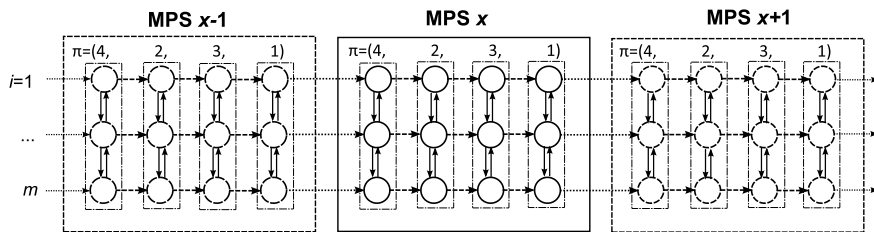


Fig. 6.6 Chain graph for the instance

in the next cycle. We will provide hereafter a more precise and formal description. In order to avoid too excessive mathematics, in this subsection we denote by single index j the operation $j \in O$ (actually, j is the pair). The graph has the form

$$G^*(\pi) = (O^*, E^*(\pi) \cup W^*(\pi)), \tag{6.29}$$

with a set of nodes

$$O^* = \bigcup_{x=1}^{m+1} O^x, \quad O^x = \{j^x : j \in O\}, \tag{6.30}$$

where O^x is the x th copy of the set O ; we denote by j^x the operation j in the x th cycle. Likewise, we define the extended set of arcs

$$E^*(\pi) = \bigcup_{x=1}^{m+1} E^x(\pi), \quad E^x(\pi) = \{(i^x, j^x) : (i, j) \in E(\pi)\}, \tag{6.31}$$

where $E^x(\pi)$ is the x th copy of $E(\pi)$. To link copies of graphs we define additional arcs with no weights

$$W^*(\pi) = \bigcup_{x=1}^m W^x(\pi), \quad W^x(\pi) = \{((l, \pi(n))^x, (l, \pi(1))^{x+1}) : l \in M\}. \quad (6.32)$$

$G^*(\pi)$ can be perceived as an extension of $G(\pi)$, so we will introduce also the extended schedule vector $S_j, j \in O^*$.

Any path from i^x to $j^y, x \leq y$, in $G^*(\pi), y \leq m + 1$ can be described by a sequence of nodes from O^* occurring in this path

$$U_{i^x j^y} = (u_0, u_1, \dots, u_v), \quad (6.33)$$

where $u_0 = i^x, u_v = j^y$. Since there may exist many alternative paths from i^x to j^y , then we denote by $L(i^x, j^y)$ the length of the longest one among all paths from i^x to j^y , formally

$$L(i^x, j^y) = \max_U \sum_{i=0}^{v-1} (p_{u_i} + c_{u_i, u_{i+1}}), \quad (6.34)$$

where $c_{u_i, u_{i+1}}$ is the weight of the arc $u_i \rightarrow u_{i+1}$ and maximization runs over all paths U defined by (6.33). We supplement this definition setting $L(i^x, j^y) = \infty$ if path U does not exist. $L(i^x, j^y)$ depends on π , however for the convenience sake of notation we do not express this fact directly. Values $L(i^x, j^y), i^x, j^y \in O^*$ can be found either by LA or LA^* with respect to $G^*(\pi)$.

Property 4 *The calculation of $L(i^x, j^y)$ for any $i^x, j^y \in O^*$ in $G^*(\pi)$, can be done by LA in the time $O(m^6 n^3)$ and by LA^+ in the time $O(m^4 n^2)$.*

Graph $G^*(\pi)$ has $\Theta(m^2 n)$ nodes and $\Theta(m^2 n)$ of arcs, see (6.30)–(6.32). For each source node $i^x \in O^*$ we can run either LA or LA^+ to obtain paths from this i^x to all $j^y \in O^*$. Whence, the final computational complexity for LA is $O(m^2 n) \cdot O((m^2)^2 n^2) = O(m^6 n^3)$, whereas for LA^+ is $O(m^2 n) \cdot O(m^2 n) = O(m^4 n^2)$.

Considering relations between time events in the extended schedule S found on the base on $G(\pi)$, we conclude that

$$S_{j^y} - S_{i^x} \geq L(i^x, j^y). \quad (6.35)$$

Inequality (6.35) has an immediate application as follows.

Property 5 *We have*

$$T(\pi) = \max_{k=2, \dots, m+1} \max_{l=1, \dots, m} \frac{L(a_l^1, a_l^k)}{k-1}, \quad (6.36)$$

where $a_l^k = ((l, \pi(1))^k)$.

Now, we are ready to formulate the algorithm of finding $T(\pi)$.

Property 6 $T(\pi)$ can be calculated from (6.36) by using graph $G^*(\pi)$ and LA in the time $O(m^5n^2)$ and by using LA^+ in the time $O(m^3n)$.

The properties imply various optimization strategies: (A-1) use $C_{\max}(\pi)$ instead of $T(\pi)$ in order to evaluate the upper bound on a time window for cyclical schedule S (see Property 3), (A-2) find π^A by using an approximate method A with $C_{\max}(\pi)$ criterion, then calculate $T(\pi^A)$ once (see Property 3), (A-3) find π^A by using an approximate method A with $T(\pi)$ criterion (see Property 6).

It is clear, comparing Property 6 with Property 3, that variants (A-1) and (A-2) are faster than (A-3). The question “whether these algorithms remain competitive in terms of quality” will be verified next in computer tests.

6.3.4 Block Properties

The path in any graph we describe by a sequence of nodes, see (6.33). Path length is defined as the sum of node weights plus the arcs weights, see (6.34). The path $U^\pi = (w_0, \dots, w_v)$ for which right-hand-side of (6.36) reaches the maximal value we call the *extended critical path* (ECP). U^π can be naturally decomposed into sub-sequences of operations having specific features: “pass through machine” and “pass through the job”. Operations passing through the machine can be intermixed (more precisely jobs having these operations can be altered), while operations passing through the job cannot be intermixed due to belonging to the same job. Then, only the former type of sub-sequence can be considered in the context of possible improving $T(\pi)$. The maximal sub-sequence of the critical path containing nodes corresponding to successive operations processed on the same machine is called the *block* and is denoted by $B_{a:b}$, $a \leq b$, $a, b \in \{0, \dots, v\}$. In each block we distinguish the first a and the last b operation of the block; the remaining operations $B_{a+1:b-1}$ constitute the *internal block*. Although notions block and internal blocks have appeared in many our earlier papers, see for example [13, 19], for the considered problem these notions are unusual. They have completely different meaning since ECP corresponds to several laps around the cylinder, but still behaves analogous elimination properties.

Property 7 Let σ be the new processing order of jobs obtained from π by altering jobs corresponding to operations processed on certain machine. If $T(\sigma) < T(\pi)$, then at least one operation from at least one internal machine block is processed before the first or processed after the last operation of this block. Processing of jobs are adjusted accordingly.

Property 7 is a nontrivial extension of the so-called block property from our earlier papers, [13, 19], which has become the flagship approach of our research team.

6.4 Parallel Methods

Two methods of parallel determination of the minimum cycle time for the determined order of operations π will be presented below, differing in their operation time and the required number of processors.

6.4.1 Parallel Determination of the Longest Paths in the Graph

Later in the work it will be required to quickly determine the longest paths in the graph between all pairs of vertices. Typically, a variant of the Warshall-Floyd algorithm with the longest paths (the shortest in the original being determined) is used sequentially, see Cormen et al. [7] for detail. However, when running the algorithms in a multiprocessor environment, it is preferable to parallelize the less efficient algorithm sequentially, called the longest path algorithm based on the Cormen et al. [7] matrix multiplication idea. This algorithm has a sequential complexity of $O(\frac{o^3}{\log o})$, where o is the number of vertices in the graph.

In the further part of the work, a parallel version of this algorithm will be proposed, with the computational complexity of $O(\log^2 o)$ with the use of $O(\frac{o^3}{\log o})$ processors. Algorithm starts from an idea of Gibbons and Rytter [10] for the shortest paths. It has been developed in the paper of Steinhöfel et al. [26] for the longest path and requires $O(o^3)$ processors, and then in the work [3] in the version for $O(\frac{o^3}{\log o})$ processors. As the algorithm requires simultaneous reading and there is no parallel writing to memory cells, the CREW PRAM (Concurrent Read Exclusive Write Parallel Random Access Machine) is an adequate model of parallel computing.

Let $A = [a_{ij}]_{o \times o}$ be the distance matrix of a certain graph, which is the input of the algorithm. As the output algorithm will return the matrix $A' = [a'_{ij}]_{o \times o}$ defined as follows:

$$a'_{ij} = \begin{cases} 0 & \text{for } i = j, \\ \max_{i=i_0, i_1, \dots, i_k=j} \{a_{i_0 i_1} + a_{i_1 i_2} + \dots + a_{i_{k-1} i_k}\} & \text{for } i \neq j, \end{cases} \quad (6.37)$$

where the maximum is determined for all possible strings of indices $i_0, i_1, i_2, \dots, i_k$, so that $i = i_0$ and $j = i_k$. The value of a'_{ij} is the length of the longest path from the vertex i to j in the graph with the distance matrix A . The algorithm will additionally use auxiliary matrices: $M = [m_{ij}]_{o \times o}$ (two-dimensional) and $Q = [q_{ijk}]_{o \times o \times o}$ (three-dimensional).

Steps 1 and 3 can be performed in $O(1)$ constant time on $O(o^2)$ processors. In step two, the maximum can be calculated over time $O(\log o)$ with $O(\frac{o}{\log o})$ processors (see Božejko [4]). By using o^2 processors, each of which has a group of $O(\frac{o}{\log o})$ processors to determine the maximum in parallel, the entire Step 2 can be performed

```

Input : A matrix;
Output: A' matrix;
{Step 1}
parfor all (i, j), i = 1, 2, ..., o, j = 1, 2, ..., o do
  |  $m_{ij} \leftarrow a_{ij}$ ;
end
{Step 2}
for iter = 1, 2, ...,  $\lceil \log_2 o \rceil$  do
  | parfor all (i, j, k), i = 1, 2, ..., o, j = 1, 2, ..., o, k = 1, 2, ..., o do
  | |  $q_{ijk} \leftarrow m_{ij} + m_{jk}$ ;
  | end
  | parfor all (i, j), i = 1, 2, ..., o, j = 1, 2, ..., o do
  | |  $m_{ij} \leftarrow \max\{m_{ij}, q_{i1j}, q_{i2j}, \dots, q_{ioj}\}$ ;
  | end
  | parfor all (i, j), i = 1, 2, ..., o, j = 1, 2, ..., o do
  | |  $m_{ij} \leftarrow \max\{m_{ij}, q_{i1j}, q_{i2j}, \dots, q_{ioj}\}$ ;
  | end
end
{Step 3}
parfor all (i, j), i = 1, 2, ..., o, j = 1, 2, ..., o do
  | if i  $\neq$  j then
  | |  $a'_{ij} \leftarrow m_{ij}$ ;
  | else
  | |  $a'_{ij} \leftarrow 0$ ;
  | end
end

```

Algorithm 1: ParLongestPaths (A, A')

in $O(\log^2 o)$ using $O(\frac{o^3}{\log o})$ processors, which determines the complexity and the number of processors required for the entire method.

6.4.2 Parallel Determination of the Minimum Cycle Time

Property 8 *The value of the minimum cycle time in the problem under consideration can be determined in time $O(\log^2 o)$ on $O(\frac{o^3}{\log o})$ -processors CREW PRAM.*

Proof The longest paths in a chain graph $G^*(\pi)$ can be determined by the ParLongestPaths algorithm in the time $O(\log^2 m)$ on $O(\frac{m^3}{\log m})$ -processors CREW PRAM (substituting the number of machines m in place of o). As an result we will obtain a matrix of m^2 lower bounds of the minimal cycle time, from which at least one—the greatest one—is an exact value of the minimal cycle time $T(\pi)$. The minimum of m^2 values can be determined over time $O(\log m^2) = O(2 \log m) = O(\log m)$ on a CREW PRAM with $O(\frac{m^2}{\log m^2}) = O(\frac{m^2}{\log m}) = O(\frac{m^3}{\log m})$ processors. Additionally, the time required to determine the weights in the $G^*(\pi)$ graph should be taken into account, which (using the already mentioned ParLongestPaths algorithm applied to the $G(\pi)$ graph) is $O(\log^2 o)$ on $O(\frac{o^3}{\log o})$ -processors the CREW PRAM and

determines the complexity of $O(\max\{\log^2 o, \log^2 m\}) = O(\log^2 o)$ and the number of processors $O(\max\{\frac{o^3}{\log o}, \frac{m^3}{\log m}\}) = O(\frac{o^3}{\log o})$ of the entire method. The last inequality is due to the fact that $o > m$.

Property 9 *The value of the minimum cycle time in the problem under consideration can be determined in time $O(o + \log^2 m)$ using $O(\frac{m^2}{\log m})$ -processors the CREW PRAM.*

Proof The procedure presented in the proof of the previous theorem should be used, with the difference that the longest paths in chain graph $G^*(\pi)$ should be determined using the method based on a vertex review in topological order, analogically to the LA^+ algorithm. This method has a sequential complexity of $O(mo) = O(m^2n)$ (the number of operations $o = m \cdot n$) due to the fact that m times it must perform the vertex review procedure in topological order, starting each time from the next source—a vertex representing the execution of the first operation on the machine. This routine can be easily paralleled by using m processors and getting $O(o)$ parallel run-time.

The remaining elements remain the same as in the proof of Property 8, i.e., the longest paths in time should be found in $O(\log^2 m)$ with using $O(\frac{m^3}{\log m})$ processors, and then find the maximum from m^2 calculated values, in time $O(\log m)$ with $O(\frac{m^2}{\log m}) = O(\frac{m^3}{\log m})$ processors. The total time will then be $O(\max\{o, \log^2 m\}) = O(o + \log^2 m)$ and $O(\max\{m, \frac{m^3}{\log m}\}) = O(\frac{m^3}{\log m})$ processors.

Despite the worse computation time of the method related to the Property 9, it may be in practice more beneficial from a practical point of view, because it requires a much smaller number of processors.

6.5 Application

A firm cooperating with our University would like to optimize the process of producing building blocks made of the cellular concrete. The factory provides blocks in combination of various sizes and various physical features fitted to the building construction purposes. The technological process consists of eighth stages performed consecutively, see Grabowski and Pempera [12] for details:

1. mixing the concrete ingredients (depending on the concrete sort),
2. filling the chosen size mould by a liquefied concrete,
3. primal establishing concrete in a mould to get a soft texture,
4. snipping blocks from soft concrete,
5. product quality control,
6. hardening of blocks in specific conditions (pressure, temperature),
7. transportation of blocks to the warehouse,
8. transport of blocks to clients.

A fixed portion of blocks (their number, geometry, physical properties) ordered by a client is called the production task. There is a pre-defined set of various tasks to be performed in the factory. Because of the technology requirements, there exist bounds on waiting times between operations 1 and 2, 2 and 3, 3 and 4. Indeed, filling the form can be performed immediately after the mixing components step, but not too late because of the fast hardening process of liquefied concrete in the mixing unit. After the filling the mould in step 2, the primal hardening of the concrete in the mould (step 3) should be in some range of waiting time given by appropriate lower/upper bounds. Snipping operation in step 4 cannot be performed too late after operation 3 because of possible hardening of the concrete in the mould. The optimization process can be considered in two contexts: (a) having the given the set of tasks, we would like to find the schedule minimizing the makespan, (b) having the set of tasks performed in the repetitive way, we would like to find the cyclic schedule minimizing the cycle time.

6.6 Conclusions

To our best knowledge, the proposed modelling fashion of “limited waiting-time” constraints as well as cyclical schedule are unusual. Various graphs play the significant role in the modeling as well as the solution technology. The problem has been decomposed into two sub-problems: (1) to find the optimal processing order, and (2) to find the minimal cycle time for the given processing order. For the sub-problem (2) we have proposed a few methods based on graph properties as well as parallel computations. For the sub-problem (1) we have introduced through graphs some elimination properties for local neighbourhood search metaheuristics.

Further research are needed to verify experimentally theoretical findings and to compare proposed approaches. The whole approach can be extended on other, more complex cyclic scheduling problems (as an example job-shop), problems with buffers, and so on. Each of proposed algorithm can be embedded in any meta-heuristic algorithm seeking the best job processing order π . In turn, the proposed methodology of parallel determination of the minimum cycle time can be used both on the multi-core CPU platform and GPU co-processor platform, which has recently been growing in popularity.

Acknowledgements The paper is partially supported by the National Science Centre, grant OPUS no. DEC 2017/25/B/ST7/02181.

References

1. Bocewicz, G., Muszynski, W., Banaszak, Z.: Models of multimodal networks and transport processes. *Bulletin of the Polish Academy of Sciences: Technical Sciences* **63**(3), 635–650 (2015)
2. Bocewicz, G., Nielsen, I., Banaszak, Z., Majdzik, P.: A cyclic scheduling approach to maintaining production flow robustness. *Adv. Mech. Eng.* **10**(1) (2018)
3. Bożejko, W.: Solving the flow shop problem by parallel programming. *J. Parallel Distrib. Comput.* **69**, 470–481 (2009)
4. Bożejko, W.: A new class of parallel scheduling algorithms. *Oficyna Wydawnicza Politechniki Wrocławskiej* textit1 (2010)
5. Brucker, P., Kampmeyer, T.: Cyclic job shop scheduling problems with blocking. *Annals of Operations Research* **159**(1), 161–181 (2008a)
6. Brucker, P., Kampmeyer, T.: A general model for cyclic machine scheduling problems. *Discrete Appl. Math.* **156**(13), 2561–2572 (2008)
7. Cormen, T.H., Leiserson, C.E. Rivest, R.L.: *Introduction to algorithms* (revised ed), 2nd edn. MIT Press, Cambridge (2001)
8. Dios, V., Fernandez-Viagas Framinan, M.J.: Efficient heuristics for the hybrid flow shop scheduling problem with missing operations. *Comput. Ind. Eng.* **115**, 88–99 (2018)
9. Dolgui, A., Ivanov, D., Sethi, S.P. Sokolov, B.: *Scheduling in production, supply chain and industry 4.0 systems by optimal control: fundamentals, state-of-the-art and applications*. *Int. J. Prod. Res.* **5**(2), 411–432 (2019)
10. Gibbons, A., Rytter, W.: *Efficient Parallel Algorithms*. Cambridge University Press, Cambridge (1988)
11. Gilmore, P., Gomory, R.: Sequencing a one state-variable machine: A solvable case of the traveling salesman problem. *Operations Research* **12**(5), 655–679 (1964)
12. Grabowski, J., Pempera, J.: Sequencing of jobs in some production system. *European Journal of Operational Research* **125**(3), 535–550 (2000)
13. Grabowski, J., Pempera, J.: New block properties for the permutation flow shop problem with application in tabu search. *Journal of Operational Research Society* **52**(2), 210–220 (2001)
14. Grabowski, J., Pempera, J.: Some local search algorithms for no-wait flow-shop problem with makespan criterion. *Computers & Operations Research* **32**(8), 2197–2212 (2005)
15. Hall, N.G., Sriskandarajah, C.: A survey of machine scheduling problems with blocking and no-wait in process. *Operations Research* **44**(3), 510–525 (1996)
16. Henneberg, M., Neufeld, J.: A constructive algorithm and a simulated annealing approach for solving flowshop problems with missing operations. *International Journal of Production Research* **54**(12), 3534–3550 (2016)
17. Howard, R.: *Dynamic programming and markov processes*. Technology Press and Wiley, New York, NY (1960)
18. McCormick, S., Pinedo, M., Shenker, S., Wolf, B.: Sequencing in an assembly line with blocking to minimize cycle time. *Operations Research* **37**(6), 925–935 (1989)
19. Nowicki, E., Smutnicki, C.: A fast tabu search algorithm for the permutation flow-shop problem. *European Journal of Operational Research* **91**(1), 160–175 (1996)
20. Pempera, J., Smutnicki, C.: Open shop cyclic scheduling. *European Journal of Operational Research* **269**(2), 773–781 (2018)
21. Pinedo, M.: *Scheduling*, vol. 5. Springer, Berlin (2012)
22. Rajendran, C.: A no-wait flowshop scheduling heuristic to minimize makespan. *Journal of the Operational Research Society* **45**(4), 472–478 (1994)
23. Ruiz, R., Maroto, C.: A comprehensive review and evaluation of permutation flowshop heuristics. *European Journal of Operational Research* **165**(2), 479–494 (2005)
24. Saravanan, M., Sridhar, S., Harikannan, N.: Minimization of mean tardiness in hybrid flow shop with missing operations using genetic algorithm. *Journal of Advanced Manufacturing Systems* **15**(02), 43–55 (2016)

25. Smutnicki, C.: A new approach for cyclic manufacturing. In: International Conference on Intelligent Engineering Systems, pp. 275-280 (2018)
26. Steinhöfel, K., Albrecht, A., Wong, C.K.: Fast parallel heuristics for the job shop scheduling problem. *Computers & Operations Research* **29**, 151–169 (2002)
27. Taillard, E.: Benchmarks for basic scheduling problems. *European Journal of Operational Research* **64**(2), 278–285 (1993)
28. Tseng, C., Liao, C., Liao, T.: A note on two-stage hybrid flowshop scheduling with missing operations. *Computers & Industrial Engineering* **54**(3), 695–704 (2008)

Part II
Graph-Based Modelling in Science
Especially in Medicine and Chemistry

Chapter 7

Using Graphs in Processing of Light Microscope Medical Images



M. Ždímalová, A. Chatterjee, M. Kopáni, and H. Svobodová

Abstract We consider graph theoretical approach to image processing, especially to segmentation of the biological images. At first, we provide a brief overview of methods from graph theory used for to image segmentation. Next, we focus on graph cuts method and its application in image processing. We show how we transform an image to a graph and corresponding segmentation network. Then, we show how we deal with segmentation of the real biological data. We define a completely new method for pre-processing data to get realistic results. We bring new pre-processing algorithm for preparing data for segmentation and also postprocessing counting of results and verifying the hypotheses. The chapter fullfils expectations of medical and biological researchers to get mathematical methods and tools for analysing the results and verifying the hypotheses. It refers to user-defined requirements of data processing, numerical evaluation, counting of cells or minerals, percentual performing, counting of minerals in the tissue. Our own contribution is the “software”, is created on special properties, requirements and requests on biological and medical researchers, because free and open software commonly available on Internet does not give them satisfying results. We want to show that graphs and its applications bring enough good tools for processing iron in medical and biological data. We used biological data in the connection with Alzheimer disease to achieve our goal.

Keywords Graph cuts · Clustering · Segmentation · Computer image analyses · Alzheimer disease

M. Ždímalová (✉)

Slovak University of Technology in Bratislava, Bratislava, Slovakia
e-mail: zdimalova@math.sk

A. Chatterjee

Indian Institute of Technology Kharagpur, Kharagpur, India
e-mail: anuprava.livetowin@gmail.com; anuprava.livetowin@iitkgp.ac.in

M. Kopáni · H. Svobodová

Comenius University in Bratislava, Bratislava, Slovakia
e-mail: martin.kopani@fmed.uniba.sk

H. Svobodová

e-mail: helena.svobodova@fmed.uniba.sk

7.1 Introduction

In computer vision, image segmentation is the process of dividing a digital image into multiple segments (sets of pixels, also called as super pixels). The objective of segmentation is to simplify or modify the representation of an image into something that is more significant and easier to analyze [1, 2]. Image segmentation is normally used to trace objects and boundaries (lines, dots, curves, etc.) that appear in images. Image segmentation is the process of allocating a label that shares some pictorial characteristics. The outcome of image segmentation is a set of segments or regions that together represent the entire object. Every pixel in a region is similar to each other with respect to certain characteristics briefly computed property, such as color, texture, intensity etc. Neighboring areas of regions are meaningfully different with respect to the same characteristics [1]. It means that areas getting by segmentation have different properties and these differences divides the picture into region and objects. The structure of the graphs is always a grid and its extensions into a network. Numerous practical applications of image segmentation, are for example, medical images—finding tumors and other pathological tissues, computer—guided surgery, treatment planning, optical character recognition (OCR), the study of anatomical structures, i.e. locating objects in satellite images (roads, rivers, forests, etc.), face recognition, traffic control systems, fingerprint recognition, brake light detection, agricultural imaging-crop disease detection, machine vision. Some general-purpose algorithm and techniques have been developed for image segmentation. Since there is no common solution to the image segmentation problem, these techniques have to be combined with sphere knowledge towards effective solution of an image segmentation problem for a problem domain. The methods can be classified in three main categories: traditional methods, graph theoretical methods, combinations of both the traditional and graph theoretical methods.

In this chapter we focus on graph theory approach to image processing. We show the application of graph cuts method to segmentation and its application for proceeding and analyzing real bio-medical. We present a new method for preprocessing data, proceeded real bio-medical data in connection with Alzheimer disease. Alzheimer disease, is a neurodegenerative brain damage. Amyloid plaques and tau proteins damage neurons and destroy their function [3, 4]. Patients have memory impairment and cognitive problems [5–6]. Prevalence of the Alzheimer disease increases with increasing age of population [7]. Problem is that we do not know restore to health these patients and also, diagnostic is possible mostly in late stage of this disease after cognitive and memory problems started, [8]. Therefore, early diagnostic is vital for Alzheimer patients [9, 10]. Many experiments from last years showed changes in iron metabolism in brain—mostly in cortex and in hippocampus, which is associated with memory [4, 7, 11–12]. Hence, we aim to iron in cortex area. Knowledge about iron changes could help early diagnostic and could prolong patient's life [13, 14]. This contribution is unique in the sense, that we created a new program with implemented a completely new preprocessing algorithm with

combination of graph cuts algorithms. We need to highlight that the new own software was created on special requirements for medical and biological data. Biologists very often needs for their processing of data mathematical tools which will give them adequate results. They need mathematical proofs for give finding elements, minerals in samples. The free software or open source programs will often not give enough good and satisfying answers for their analyses. Even open software like Image J [15], Cell profiler [16], Meta Morph [17, 18], Ilastic [19] and others did not give them enough good results. On Faculty of Medicine, Comenius University in Bratislava, researchers usually use just basic threshold methods, which will not give good segmentation and definitely not any kind of numerical evaluation of medical data. We even cannot discuss give this way number of elements, minerals or percentage presentations of segmented elements in these samples. Our contribution arises in direct call from practice, for the special analyses of biologists. We created special software with implemented our new algorithm for preprocessing of data, and also postprocessing for evaluating results. Others known methods were not suitable for these data. This contribution satisfied special requirements of real medical analyses. It happens very often that for some kind of special biological research biologist needs to answer special kind of questions. Special programs of special properties need to be created on the calls from real biological research. Biologists were not able to obtain their analyses by the basic methods. But our own program gave them good results and evaluations. The paper discusses graph theory approach to image segmentation. We use the method of graph cuts for analyzing bio-medical data. In Sect. 7.2 we discuss used biological material and samples and data and methods. In the third section we give an overview of graph theory methods used in image processing and some others examples. In the Sect. 7.4 we define new preprocessing method handling biological data. In Sects. 7.5 and 7.6 we summarize results and give quantitative evaluation of samples—detection iron in the samples.

7.2 Material and Methods

Our research was performed on Alzheimer mouse model APP/PS1, which is good explored and these animals have mutations lead to Alzheimer disease, [20, 21]. We used wild-type littermates (wt) as controls. Animals were in standard laboratory-controlled conditions with 12:12 light-dark cycle. They had food and water ad libitum. Experimental animals were sacrificed in different ages—2.5 month, 8 months, 13 months, 24 months. Brains were perfused and collected to 4% formaldehyde. After 4 h, the brains were dehydrated in 30% saccharose overnight. We cut the brains into 35 μm sections with microtome (Leica SM 2000F, Wetzlar, Germany). Sections were stained with Perl's blue for iron with DAB (3,3'-Diaminobenzidine) [22]. After staining, we could observe iron as dark brain dots. For visualization of amyloid plaques, we used congo red staining in the same section [23]. The stained samples were observed with microscope (Zeiss, Gottingen, Germany) and photos were taken with attached AxioCam MRC 5 camera (Zeiss, Gottingen, Germany). We

took photos from cortex area of each section. We continue with data analysis and graph theory approach for segmentation of data.

7.3 Segmentation and Pre-processing Method

We created the software based on graph cutting algorithms as well as new pre-processing method. The program was created in C++. The process of preprocessing data, image segmentation and postprocessing data consists of the following steps:

1. Data collection from a microscope.
2. Pre-processing of the data: thresholding combined with new algorithm.
3. Software initialization and specific input image loading. (the graph cut algorithm).
4. Processing of the data-segmentation using graph cuts. (the graph cut algorithm).
5. Counting of the segmented objects.
6. Output saving: image and numerical data.

The process of segmentation using graph cut will be described in details in the next chapter. Now we present a new approach to data pre-processing. We need to note that if we used directly origin images from microscope and use graph cut, we did not get results with weak real interpretation. Therefore, on the request of the specialist of the images we developed a new algorithm for data pre-processing for getting real quantity of iron in the data.

The usual process of grey thresholding and other traditional thresholding methods involve defining a range of brightness value in the original image, then selecting the pixels within the range as belonging to the foreground and rejecting all of the other pixels to the background. The problem with this type of algorithm is that it does not segregate between two pixels of the same brightness but different colour schemes. Hence, detection of small brown dots in a biological image using simple thresholding leads to a lot of false positive counts. For example, while attempting to identify the iron samples (small brown dots represented by brown arrows) in Fig. 7.1, the algorithm also wrongfully identifies the artefacts (black dots), amyloid (red spots shown by black arrows), and the occasional dark spots in the Brain cortex, just because all of them have the same brightness.

Hence, we came up with the RGB based multispectral thresholding method. In any given color image, every pixel is characterized by three RGB values. We extract the RGB values of every pixel and apply hard thresholds on each of the 3 channels, i.e. the red, blue and green channels separately. The hard thresholds are decided by examining the training set of images and listing the RGB values of all the pixels which are known to be our target. The minimum and maximum values in that list of all the three colors are then accepted as boundaries. An error of 10% is allowed on either side of the boundaries for experimental purposes. These multiple criteria are then combined with a logical AND operation, where we reject all of those pixels which do not meet the one or more of the three thresholds to the background and accept all

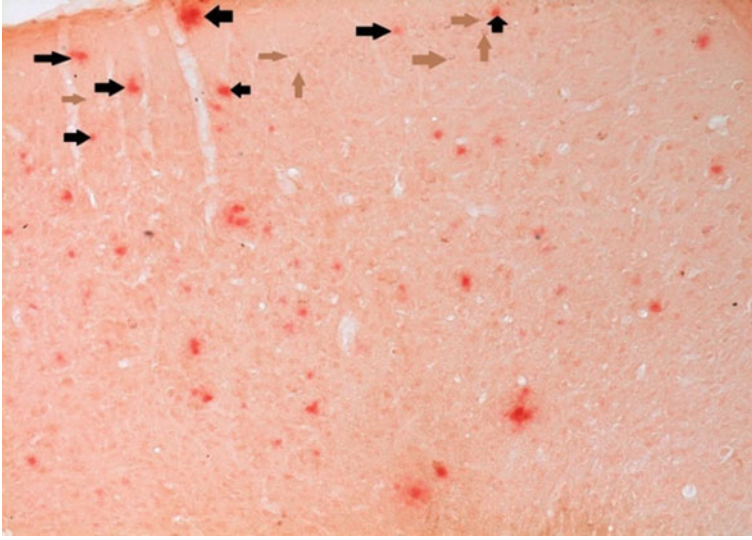


Fig. 7.1 Brain cortex sample for detecting presence of iron

the other pixels as belonging to the foreground. The mathematical equivalent of the technique is depicted from Eq. (7.1)–(7.4)

$$g1(x, y) = \begin{cases} 1, & L_r < r(x, y) < H_r, \\ 0, & \text{otherwise.} \end{cases} \quad (7.1)$$

$$g2(x, y) = \begin{cases} 1, & L_g < g(x, y) < H_g, \\ 0, & \text{otherwise.} \end{cases} \quad (7.2)$$

$$g3(x, y) = \begin{cases} 1, & L_b < b(x, y) < H_b, \\ 0, & \text{otherwise.} \end{cases} \quad (7.3)$$

$$h(x, y) = g1(x, y) \text{ AND } g2(x, y) \text{ AND } g3(x, y) \quad (7.4)$$

Here, L_r , L_g , and L_b depict the target lower limits of the red, green and blue channels respectively, while H_r , H_g , and H_b signify the upper limits. $r(x, y)$, $b(x, y)$, and $g(x, y)$ signify the red, blue and green values on the RGB scale of the pixel with x and y as coordinates. $h(x, y)$ gives the coordinates of the pixels which match the target RGB thresholding and are potential iron samples present in the Brain cortex. But this doesn't eliminate the occasional dark spots in the brain cortex which might be mistaken for iron. Hence, from the training set of images, the coordinates signifying the iron samples are curated, and the maximum size of a cluster formed by adjacently placed pixels from that list is determined. Let this value be represented by S . Every (x, y) with $g(x, y) = 1$ in the test images is associated with a cluster. The condition for considering a given pixel $P(x, y)$ as a part of a cluster is

if $g(x, y + 1)$, $g(x + 1, y + 1)$, or $g(x + 1, y) = 1$. If yes, then P is considered a part of the cluster, and 1 is added to the cluster size. Iterating this for all the identified points, a list of possible clusters is curated. The size of a cluster is equal to the number of pixels associated with that cluster. If the total size is greater than S , then that cluster and all those points are discarded. This is based on the notion that real iron dots would be detected inside the cell cytoplasm and hence large associations of dark dots don't signify iron. Determination of whether the thresholding method is successful or not, relies solely on human intervention. Therefore, the threshold value needs to be varied until acceptable results are achieved, based on human observation. That is why, in carrying this method, it may be necessary to do a few levels of thresholding in order to get the best results.

Others steps of the process are provided by graph cuts methods. We describe graph approach in the next chapters. We describe in details how the graph cuts will be provided as well how we bring the overview of many segmentation methods.

7.4 Graph Theoretical Approach to Image Segmentation

Many methods of image segmentation lead to the graph theory. We need to define at first certain concepts [24–25].

Let $G = (V, E)$ be a graph, where $V = \{1, 2, \dots, n\}$, is a set of vertices that correspond to the image elements, which might be represented pixels or regions in Euclidean space. Let $E, E \subseteq 2^V$ be the set of edges that connects certain pairs of neighboring vertices.

- Each edge $\{i, j\} \in E, i, j \in V$ is measured by the weight $w_{ij}, i, j \in V$, which represents a certain quantity based on the property between the two vertices connected by that edge.
- The image can be divided into separate mutually exclusive components, being accepted that each component A is a connected graph $G = (V', E')$, where $V' \subseteq V, E' \subseteq E$ and E' contains only the edges (arcs) created from the vertices in V' .
- For the components it should be valid that the properties such as brightness value, color, and texture are similar throughout the whole component.
- Then, the degree of dissimilarity of two components can be calculated as the cut of the graph. A graph is related to a set of edges by which the graph G is partitioned into two disjoint sets A and B .
- As a consequence, the segmentation of an image can be interpreted in form of a graph cut. The graph cut value is defined as in Eq. 7.5.

$$\text{cut}(A, B) = \sum_{i \in A, j \in B} w_{i,j}, \quad (7.5)$$

where i and j are vertices of the different components. Afterwards, the issue of image segmentation can be treated as an optimization problem in which we try to minimize according certain criterion.

- In this case it shall be optimal to divide the graph into two segments, which minimize the graph cut.
- It is difficult to obtain an accurate image segmentation solution. Therefore, it is more appropriate to solve this problem with optimization method.

The optimization-based approach formulates the problem as a minimization according to some established criterion, whereas one can find an exact or approximate solution to the original uncertain visual problem. The optimal bi-partitioning of a graph can be taken as the one which minimizes the cut value in Eq. (7.1). Image segmentation can be formulated also as a labelling problem, where a set of labels L is assigned to a set of sites in S . In two class segmentation, for example the problem can be described as assigning a label f_i from the set $L = \{\text{object, background}\}$ to site $i \in S$, where the element in S are the images or regions. Labelling can be performed separately from image partitioning. They achieve the same effort on image segmentation. Many methods perform both partitioning and labelling simultaneously [1, 26, 27]. In the following text, we describe the basic partitioning methods using the knowledge from the graph theory [28, 29]. Methods in image segmentation can be categorized into automatic methods and interactive methods. Automatic segmentation is desirable in many cases for its convenience and generality. However, in many applications such as medical or biological imaging, objects of interest are often ill-defined so that even sophisticated automatic segmentation algorithms would fail. Interactive methods can improve the accuracy by incorporating prior knowledge from the user however, in some practical applications where a large number of images are needed to be handled, they can be of high cost and much time consuming.

Minimal cuts methods. Using the method of graph cutting for image segmentation was firstly proposed by Wu and Leahy [1, 2]. Graph cut based methods possess a distinctive property against previous methods in that a general framework of optimally partitioning the graph globally is presented. This brings the advantages that for different applications different cost functions can be designed with a clear definition of segmented objects. Graph cut provides a graph partition: minimizing this cut makes vertices in different sets dissimilar. It also requires that the vertices in the same set be similar. These two requirements are studied by existing graph cut methods which try to meet one or two of these requirements [1, 26].

Normalized graph cuts: In addition to graph cut defined in the Eq. (7.6), there are alternative definitions of cuts for which we want to minimize the value. From the previous statement, the advantage of these methods is apparent, including the ability to use different types of cuts for various applications. The main disadvantage of the cut defined as before Eq. (7.2) is a preference for finding small components. This problem can be solved by using a normalized cut. The graph cut is measured by the weights of $vol(\cdot)$, which is the total connection from vertices in a set (e.g. A) to all the vertices in the graph [2, 30, 31]. Formally we have depicted in Eq. 7.6

$$\text{vol}(A) = w_{i,j}, \quad (7.6)$$

where the weight w_{ij} , $i, j \in \{1, \dots, n\}$, measures a certain image quantity (e.g. intensity, color, etc.) between two vertices connected by that edge. Then N cut cost function is defined in Eq. 7.7 as

$$\text{Ncut}(A, B) = \frac{\text{cut}(A, B)}{\text{vol}(A)} + \frac{\text{cut}(A, B)}{\text{vol}(B)}. \quad (7.7)$$

Graph cut with shape prior. Incorporating the shape prior in graph cut has been proven to be very useful for image segmentation. This visual clue can be added in the regional term or the boundary terms to force these segmented objects to follow a certain pre-defined shape. The idea of using a signed map function to represent some shape was proposed by Kolmogorov [32]. The gradient flow evolution of a surface was computed by L_2 distance from the drift from its current position. It guarantees that the shape is not very far from the involving process. Freedman et al. used similar ideas in level-set methods to specify the template as a distance function whose zero-level set corresponds to the template, Das et al. and Veksler [1, 2, 26, 30] studied more general shape prior for image segmentation. These shapes are defined on the relative positions of neighboring pairs of pixels. The neighborhood system for incorporating the shape constraints is the same as the boundary constraints. In two labelling cases, minimizing the shape-based energy functions can be accomplished exactly with a graph cut if all the pairwise terms are submodular.

Interactive graph cut methods. The interactive property of graph cut allows an efficient editing of segmentation results. The lazy snapping and Grabcut [26] provides quick object marking schemes for better user experience. Users are allowed to lose position seed points to indicate which parts of the image are objects and modify the segmentation results by editing the boundary with some soft constraints [2, 26, 30, 33].

Efficient graph-based Image Segmentation. This method divides an image into regions or segments [1, 26, 30]. A predicate is defined to measure the indication for a boundary between two regions of an image, using a graph-based representation. Afterwards, an efficient segmentation algorithm is developed based on this predicate. It shows that although the algorithm makes avaricious decisions it produces image segmentation that fulfils global properties. The algorithm applied to image segmentation uses of native neighborhoods in creating the graph and determines the results with both actual and artificial images. The algorithm runs in time, approximately linear in the number of graph edges and also in quick practice.

Iterated Graph Cuts for Image Segmentation. This technique begins from a sub-graph of a given graph representing an image. This includes the user labelled foreground or background regions. This technique works iteratively to label the neighboring nonsegmented regions or image segments. In order to get better efficiency and robustness of image segmentation, the mean shift method is used to split the image

into homogenous regions, and then the iterated graph cuts algorithm is implemented by compelling each region, rather than every pixel, as the graph node for image segmentation [2, 26, 30, 33].

Segmentation using minimal spanning trees. Methods are based on minimal spanning tree [1, 26]. Frame T is a tree of a graph $G = (V, E)$, V is the set of vertices of G , E is the set of edges of G , such that $T = (V, E')$, where $E' \subseteq E$. To find such a minimal spanning tree, we may use for instance Prime's algorithm, in which we iteratively add edges to get a tree with the smallest weights.

Segmentation using Euler graphs. This technique explains an algorithm for image segmentation problem using the concept of Euler graphs in the graph theory. The image is treated as an undirected weighted non planar finite graph G , and then image segmentation is treated as graph partitioning problem. This method locates region boundaries or clusters and runs in a polynomial time. Subjective comparison and objective estimations show the efficiency of the method in different image domains. The algorithm begins by randomly choosing an edge and tries to form closed regions. During the process, open paths are formed [2, 26, 30, 33].

Shortest path-based method. Methods based on the shortest paths in graphs. Finding the shortest path between two vertices is a classical problem of the graph theory (for further reading on shortest paths in the graph, see [1, 2, 30, 33]). The best-known algorithm that solves this issue is Dijkstra's algorithm. Segmentation problem, in which we are looking for the best segment boundaries, is reformulated as a problem of finding the shortest path between two vertices. In practical applications, the user interaction is applied, thereby it increasing the accuracy of the segmentation. The live-wire method is used, which allows the user to select the first point on the border. Subsequently, the shortest route between the first point and the actual location of the cursor is displayed in real-time. We also mention a segmentation technique called "Intelligent Scissors", based on the principle of seeking the shortest paths in a graph.

Graph cuts based on Markov random fields. Studying psychology has shown that for interpreting an image is necessary to take into account the context of the image information. It means that if we attempt to identify an object which we do not see entirely, we may not succeed. The image should, therefore, be understood in the visual and spatial context. The theory of Markov random fields allows this approach [1, 26, 33].

For completeness of the survey we refer to classical other methods too:

Thresholding methods [17]. It is the simplest technique of all image segmentation methods. This method is constructed on a threshold value, used to transform a grey-scale image into binary image. The basic objective of this technique is to select a single threshold value. Thresholding is one of the broadly used methods for image segmentation. It is useful in discerning foreground from the background. By selecting suitable threshold value T , the grey level image can be transformed to binary image. The binary image should contain every important information about the position and shape of the objects of interest (foreground). The advantage of gaining first a

binary image is to reduce the complexity and image classification. The best way to convert a grey-level image into a binary image is to select a single threshold value T . Then every grey level value below this T will be classified as black, and those above the value of T will be white. The segmentation problem depends on selecting the proper value for threshold T . In actual applications histograms are more complex, with various peaks and not clear valleys. One disadvantage of the method is that it is not always easy to select the value of T . There are more threshold methods: Minimum Thresholding, Iterative Thresholding, Entropy based Thresholding, Otsu Thresholding, e.t.c.

Region growing methods. This method is based on an expansion of an object detected inside of an object, [1, 34, 35]. We choose object seed pixels (inside an area to be detected) and then we are searching for neighboring pixels with similar intensities as has the object seed pixel (in 4 or 8 different directions). The algorithms work until none of the remaining pixels is similar to each other. Then the whole object is discovered.

Level sets methods. Level sets are an important category of modern image segmentation techniques based on partial differential equations (PDE), i.e. progressive evaluation of the differences among neighboring pixels to find object boundaries, [2, 33]. Ideally, the algorithm will converge at the boundary of the object where the differences are the highest.

We mention also many others methods: Histogram based methods [2], split and merge methods [36], methods based on comparison [37], watershed transformation [1], segmentation using neural networks [27], fuzzy approach [38, 39], active contour models [27], and others.

7.4.1 *Graph Cutting*

We introduce segmentation by graph cutting and its application to real biological data. Segmentation problem has been studied for many years [1, 27, 30, 33]. We focused on graph theoretical approach to segmentation which is called “graph cutting” [24–25, 32–40]. It means that we have data represented as an image. We convert data to the graph representation by a network. In the network we search for a maximal flow which corresponds to a minimal cut in the network. According [24–25] the minimal cut in network is the equivalent problem to finding a segmentation. We created programs in C++ platform which segment images of data and return outputs according to specific requirements for biological purposes of finding minerals in the corresponding data.

7.4.2 *Graph Cuts in Image Processing*

The aim of a segmentation is to divide an image into regions and thus simplify its representation. In our case we will consider two kinds of regions: object regions and background regions. The output is a binary image with one value representing “object” and the other “background”. Both “object” and “background” segments can be composed of several isolated parts. Each region contains pixels with some common characteristics. The segments created in this way should correspond with the input data and the output binary image can be easier to analyze. In general, the segmentation algorithms are based on a similarity and discontinuity properties. Each of the pixels in a region is similar with respect to some characteristic or computed property, such as, e.g., the intensity. Neighboring regions are significantly different with respect to the same characteristic on their border, we usually observe significant intensity changes. The segmentation based on the graph cuts in image processing was first introduced in [24, 41]. More detailed information on origin of a graph theoretical approach in image processing can be found in [24, 41, 30, 33]. First, we explain in details segmentation based on the minimization graph cut algorithm. We discuss methods evaluating the maximal flow for the image to extend a minimal cut in a graph or a network. In graph cutting there are generally two main approaches how to extend the maximal flow. The first one focuses on the algorithms based on augmenting paths in a network or a weighted graph, and the second one is “push-relabel” methods and algorithms. In this part, we deal with the first approach finding the graph cuts based on a maximal flow for image segmentation based on algorithms looking for the augmenting of the shortest paths in the network. The link between graph cuts and segmentation is the following: let vertices of a graph represent pixels (voxels) and let graph edges represent neighborhood relationships between them. The cut part vertices of the graph. The minimal cut determines the segmentation optimal with respect to the weights of a graph. These weights must be set at the beginning in a reasonable way. The techniques use a well-known combinatorial fact that a globally minimal graph cut with two terminal nodes can be obtained in a polynomial time. One of the main advantages of this interactive method is that it provides a globally optimal solution when the cost function is clearly defined. If we compare this method with simulated annealing, where we are never sure that our minimum is global, in our method we get a globally optimal solution. Thus, the segmentation can be controlled more reliably.

7.4.3 *Segmentation as a Minimization Problem*

We describe the segmentation method described in [24–25, 42]. Let us consider a set of data elements P and a neighborhood system given by a set N of all unordered pairs $\{p, q\}$ representing neighboring elements in P . In our case, P determines a set of pixels in a 2D grid and N contains all unordered pairs of pixels p, q . We use

a standard neighborhood system consisting of four nearest neighbors. Let p denote a pixel in an image. Let us have an image with M pixels and let I_p express the intensity of a pixel p . The segmentation is defined as an unambiguous assignment of every pixel to an object or to a background. It will be represented by a vector $Z = (Z_1, Z_1, \dots, Z_M)$ with values from the set “object”, “background”. The cost of segmentation is defined in terms of boundary and region properties. It is determined in such a way that every pixel p is assigned a value of a regional property R_p and a boundary property $B_{(p,q)}$, which can be understood as penalties for a wrong assignment of a pixel to an object or a background, and the resulting segmentation will be the configuration of a vector Z with a minimal penalty value. The user is expected to show a group of pixels (possibly one pixel), representing an object or a background, respectively. These pixels will be called “seeds”. “Hard constraint” of the algorithm means that the object seeds must be in the resulting segmentation set to “object” and the background seeds must be set to “background”. The regional and boundary properties of segmentation can be viewed as “soft” constraints of the segmentation.

Setting of R_p and $B_{(p,q)}$ values: It is based on values of object and background seeds. These “typical” values of objects and a background, denoted as “ I_o ” and “ I_B ”, are set interactively. In this example they are set to average of selected object seeds “ I_o ” and selected background seeds “ I_B ”. Example of a simple setting of a regional property is the given by Eq. 7.8 and 7.9.

$$R_p(Z_p) = |I_o - I_p|, \quad \text{if } = \text{“object”}, \quad (7.8)$$

$$R_p(Z_p) = |I_B - I_p|, \quad \text{if } Z_p = \text{“background”}, \quad (7.9)$$

Let us imagine a noiseless image, a black object on a white background. If the pixel of the black object is by mistake set to background, in the cost function it will be compared with the average background value which results into higher value of the cost function. The boundary property expresses whether a couple of neighboring pixels with different values Z_p in the segmentation vector Z is an edge. An example of a simple linear setting of a boundary property for neighboring pixels p and q is given in Eq. 7.10.

$$B_{(p,q)} = D - |I_p - I_q|, \quad (7.10)$$

where D is the value of a maximal intensity difference in an image. If a pair (p, q) belongs to an edge, $|I_p - I_q|$ is large and $D - |I_p - I_q|$ is small. If an edge pixel of an object is evaluated as a background, $B(p, q)$ is large. Of course, the regional and boundary properties can be influenced by noise. Partially we can cope noise with a help of weighing constant λ mentioned in a following paragraph.

7.4.4 The Segmentation Cost

The segmentation cost depends on regional and boundary properties. We use also the weighing constant λ for a relative importance of the regional property versus the boundary property. Then the segmentation cost of Z be denoted by $E(Z)$ in Eq. (7.11)

$$E(Z) = \lambda \cdot R(Z) + B(Z), \quad (7.11)$$

where

$$R(Z) = \sum_{p \in P} R_p(Z_p)$$

$$B(Z) = \sum_{p, q \in N} B_{(p, q)} \cdot \delta(Z_p, Z_q)$$

and

$$\delta(Z_p, Z_q) = \begin{cases} 1 & \text{if } Z_p \neq Z_q \\ 0 & \text{otherwise} \end{cases}$$

where values $R_p(Z_p)$ are from Eq. (7.8–7.9) and different values Z_p in the segmentation vector Z . For more details see evaluations in [42, 43]. In our case, to find the segmentation means to solve the minimization problem: to find a vector Z with a minimal cost. As a technique, we have chosen, as we have already mentioned, the method of finding the minimal cut in an oriented graph in such a way that the cut corresponds to the border between objects and a modification background. From many methods we have chosen the Ford-Fulkerson algorithm [28, 29], and his modifications. In the following, we show the way how we construct the graph over the input image so that the minimal graph corresponds to our segmentation. Then in the graph, we must set costs of edges in such a way that cost within the object must have sufficiently large capacities not to be separated by a minimal cut, and on the other hand, the edges being the candidates for a boundary of an object must have their capacities low. Thus, the very important step is to set the capacities of the graph edges in a correct way.

7.4.5 Construction of the Graph

We construct a graph $G = (V, E)$, where V is a set of vertices and E is a set of edges. Every edge is assigned a non-negative cost $w_{i, j}$, $i, j \in \{1, 2, \dots, n\}$ (capacity). The vertices correspond to pixels p in V , see Figs. 7.2, 7.3, 7.4, and 7.5. We add two

Fig. 7.2 *N*-links connecting neighboring pixels (un-oriented)

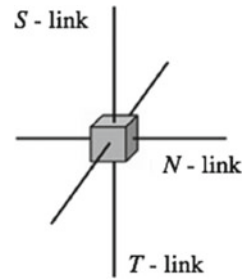


Fig. 7.3 *N*-links connecting neighboring pixels (oriented)

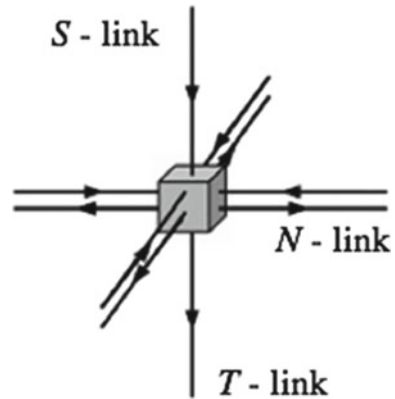
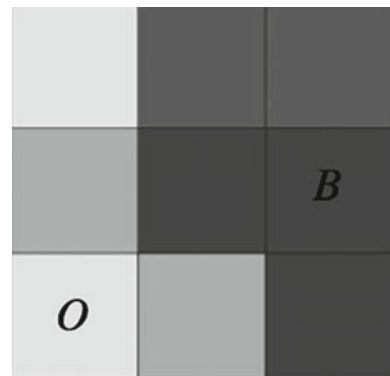


Fig. 7.4 Image with object and background pixels



new vertices, an “object” vertex (source, input *S*) and a “background” vertex (sink, output *T*), i.e., two extra vertices called terminal nodes. Thus $V = P \cup \{S, T\}$. The set of edges *E* consists of two types of unoriented edges: *N*-links (neighboring links) and *T*-links (terminal links). Thus, every pixel *p* has two *T*-links $\{p, S\}$ and $\{p, T\}$ connected to every terminal node. Every couple of neighboring pixels $\{p, q\}$ in *N* is connected by *N*-link. An arbitrary *N*-link connecting neighbors *p* and *q*, is

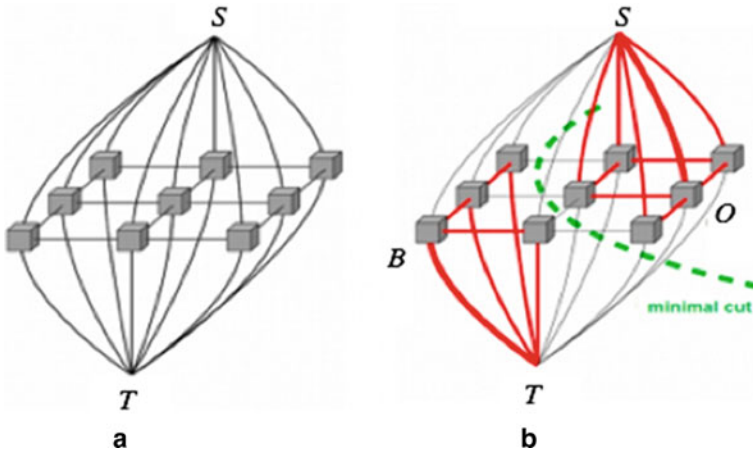


Fig. 7.5 **a** Non-oriented network. **b** Minimal cut

denoted by $\{p, q\}$. Thus, it holds $E = N \cup \{\{p, S\}, \{p, T\}\}$. The cut is a subset of edges $C \subset E$ such that the terminal nodes will be separated on an induced graph $G = (V, E, C)$. The cost C of the graph is defined as the sum of edge weights (cost) as in Eq. 7.12.

$$|C| = \sum_{(i,j) \in C} w_{i,j}. \tag{7.12}$$

For making oriented graph we just double every edge two ways and put here arrows. Such way we will get oriented graphs $G = (V, E)$, of $V = \{1, 2, \dots, n\}$ vertices and $E \subseteq V \times V$ of arcs. Every edge is assigned a non-negative cost $w_{i,j}$ $i, j \in \{1, 2, \dots, n\}$ (capacity).

To find the minimal $S \setminus T$ cut we used the result of Ford and Fulkerson [28, 29] saying that the maximal flow from S to T saturates the set of edges in such a way that it corresponds to the minimal cut (i.e. to minimal $|C|$) and the minimal $S \setminus T$ cut and maximal flow are dual problems. This fact is based in the well-known max-flow min-cut theorem [28, 29]. In the following we introduce the notation and terminology we use later. We recall that the links connecting two neighboring pixels (vertices) p and q are called N -links. Links connecting a pixel with a terminal or T are called T -links (see Fig. 7.3a, b) where a pixel is depicted as a grey cube, N -links as horizontal lines and T -links as vertical lines. The next step is to evaluate the edge capacities of this graph with respect to our demand that the minimum $S \setminus T$ cut divides nodes into an object segment and a background segment. We use the following notation:

P	is the set of all pixels
(p, q)	Is the edge connecting neighboring pixels p and q

(continued)

(continued)

I_p	is the value of the intensity of the pixel p
M	is the maximal value of the intensity of the pixel (of the responsible figure)
D	is the difference of the maximal and minimal value of the intensity of the pixel (of the responsible figure)
O_{avr}	is the average value of the intensity of object seed pixels
B_{avr}	is the average value of the intensity of the background seed pixels
$S(p)$	is the capacity of the edge (link) connecting the sink (the vertex s) and corresponding pixel p
$T(p)$	is the capacity of the edge (link) connected output source (the vertex t) and concrete pixel (the vertex p),
$N(p, q)$	is the capacity of the edge (link) connected neighbors' pixels p and q
λ	is the weighing constant

The weighing constant λ enables us to modify the result of segmentation. If λ is chosen to be low, we prefer the boundary property. We usually get fewer objects with smooth borders. By choosing a large λ , i.e., $\lambda \in (1, \infty)$ we prefer the regional property resulting in more objects with unsmooth border. Usually, at the beginning we select $\lambda = 1$ and then we modify it to get a desirable result. $N(p, q)$ expresses the relationship between intensities of p and q , $S(p)$ and $T(p)$ express the relationship between intensity values of pixels and the values O_{avr} and B_{avr} . N -links connecting links with similar intensities have their capacities greater than those with larger intensity differences. Similarly, T -links, connecting the terminal T and a pixel express the difference between a pixel and O_{avr} : the higher the difference the smaller the capacity. Likewise, the capacities of T -links depend on a “similarity” of a pixel to B_{avr} .

7.4.6 Capacity Evaluation

First, we define values of $S_{(p)}$ and $T_{(p)}$ for those pixels p which have been determined as object and background seed pixels. The value of $S_{(p)}$ will be set as $S_{(p)} = \infty$, if the pixel p is an object pixel. The reason is that $S_{(p)}$ cannot be saturated, while it connects the source S and the object seed pixel. It means that the object seed has to be reached in any case from the source S . If we denote pixel p as object seed pixel, then the cost of $T_{(p)}$ will be zero, it means that $T_{(p)} = 0$. The main point is that this edge has to be saturated and the output T will not be reachable from the pixel p . We set the capacities of edges also for all background seed pixels, but this time $S_{(p)} = 0$ and $T_{(p)} = \infty$. This guarantees that after finishing the segmentation process the object seed pixels belong certainly to the object and background seed pixels will become a background. For pixels p which are neither object nor background seeds, the capacity of N -links and T -links depends on the intensity of corresponding pixels.

The notation $N(p, q)$ expresses the relationship between intensities of p and q , $S(p)$ and $T(p)$ express the relationship between intensity values of pixels and the values O_{avr} and B_{avr} , see [42]. More about relationships between these variables and constants it is possible to find in [42]. In a similar manner the cost of the links will be counted and will be evaluated in Eqs. 7.13, 7.14 and 7.15 [42] (linear evaluation):

$$N_{(p,q)} = D - |I_p - I_q| \quad (7.13)$$

$$S_{(p)} = M - |O_{avr} - I_{avr}| \quad (7.14)$$

$$T_{(p)} = M - |B_{avr} - I_{avr}| \quad (7.15)$$

The setting of capacities can be done in different ways. As it is shown in [32, 42], Eq. 7.16, 7.17 and 7.18 shows functions which are able to use characteristics of the borders of the object, e.g. homogenous region inside an object or a background:

$$B(\Delta I) = \exp\left(-\frac{\Delta I^2}{2\sigma^2}\right) \quad (7.16)$$

$$R_s(I) = -\ln P(I/O) \quad (7.17)$$

$$T_t(I) = -\ln P(I/O), \quad (7.18)$$

where σ is the preselected parameter. $P(I|O)$ and $P(I|B)$ represents the probability that every pixel of given intensity I belongs to the object, or to the background. These values we determine from histograms, which we can obtain from given set of seeds O and B . According to some other authors in what follows [32, 42], and we use linear cost function defined by formulas Eq. (7.13), (7.14), (7.15). It is called as a linear “interaction energy”. Because of the character of the constant M and D the capacities of the edges will be non-negative. Because of the character of constant M and D the capacities of the edges will be non-negative. As we have already mentioned, the minimal cut separates the vertices of the graph into two disjoint sets. This separation corresponds to the segmentation of an underlying image. The minimal cut gives a segmentation which is optimal. The minimal cut is a feasible one. It means that: C cuts exactly one t-link at each p ; $\{p, q\} \in C$ if p, q , are linked to different; terminals if $p \in O$ then the arc $p, T \in C$; if $p \in B$ then the arc $p, S \in C$.

7.4.7 Segmentation Algorithm

Based on the previous results, we can construct the graph corresponding to image like in the Figs. 7.4 and 7.5. Because of the fact that this graph is still not oriented, it

is not possible to and the maximum flow. All pixels (without background pixels and seed pixels) have to be reachable from the source S when we are saturating the edges through T -links. The reason is that in this way we can obtain good and objective segmentation. It is easy to see that the orientation of T -links is from the source S to the sink T , it means “down” direction (from top to bottom) in the Figs. 7.2 and 7.3. According to the previous facts, the following conditions have to be satisfied: if the pixel p is reachable from the pixel q , then the pixel q has to be reachable from the pixel p . This condition will not be satisfied, if the edge (p, q) is oriented. After creating the oriented graph, we let the flow enter the edges and we start to saturate them. The maximal value of the flow is found if there does not exist the augmenting path from the source S to the sink T . After finishing the algorithm, from the saturated graph we can obtain all information needed for the segmentation. Regions of the segmentation (called also segments) will be compound of pixels with similarly intensities and will have non-zero reserve on their N -links. It holds that N -links connected pixels of background have non-zero value of a reserve and the intensity of these pixels is similar to the intensity of B_{avr} . After finding maximal flow, the pixels belonging to object are still reachable from the source S and their intensity is similar to the intensity O_{avr} .

7.5 Results

By following our own procedure in Sect. 7.2 of finding iron presence from a brain cortex sample, we came up with quite satisfactory results. The results were cross-validated by performing multiple levels of thresholding and finding the optimal values of L_i , H_i and S_i , which gives the best results. Equations (7.19), (7.20) and (7.21) represents the values of L_i , H_i and S_i ,

$$L_i = \begin{cases} 180, & i = r, \\ 90, & i = g, \\ 95, & i = b. \end{cases} \quad (7.19)$$

$$H_i = \begin{cases} 200, & i = r, \\ 150, & i = g, \\ 120, & i = b. \end{cases} \quad (7.20)$$

$$S = 74 \quad (7.21)$$

Figures 7.6, 7.7, 7.8, 7.9, 7.10, 7.11 and 7.12 shows seven test samples of original images, while Figs. 7.13, 7.14, 7.15, 7.16, 7.17, 7.18 and 7.19 shows the detection results of these images respectively.

Due to the very small size of the iron dots and constraints in image size for this paper, they might not be visible for the full-size image. Hence, we have included a

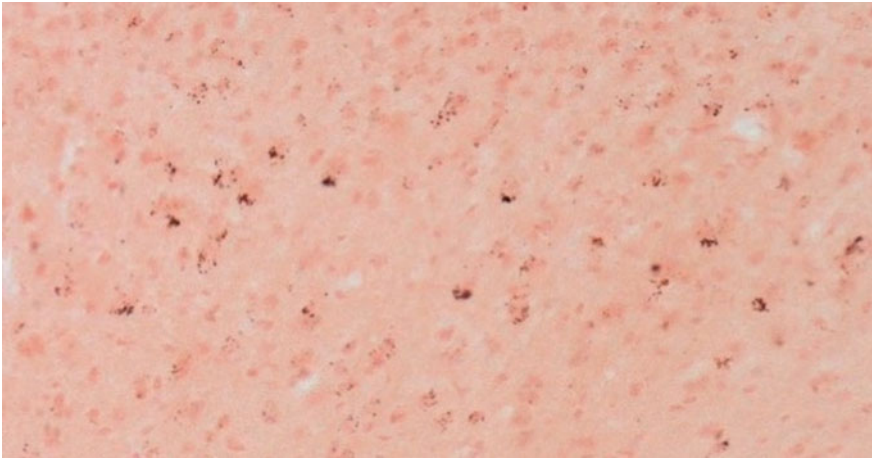


Fig. 7.6 Brain cortex sample (1)

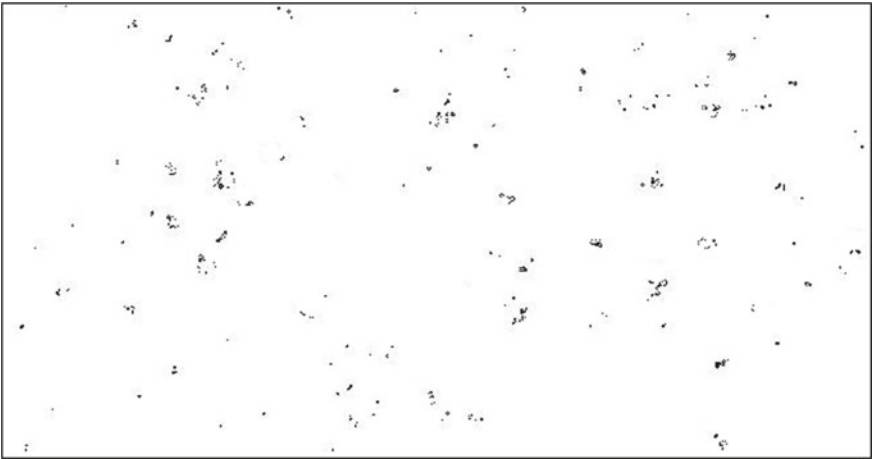


Fig. 7.7 Result of brain cortex sample (1)

cropped image for each sample and the corresponding result for that sample. The calculations done, however, are for the full-size image.

The percentage of pixels detected out of the total image sample can be found out as shown in Table 7.1.

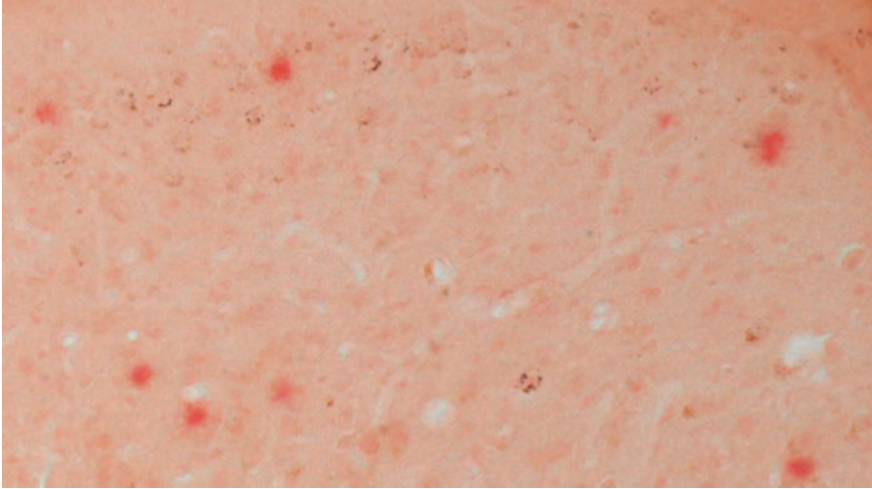


Fig. 7.8 Brain cortex sample (2)

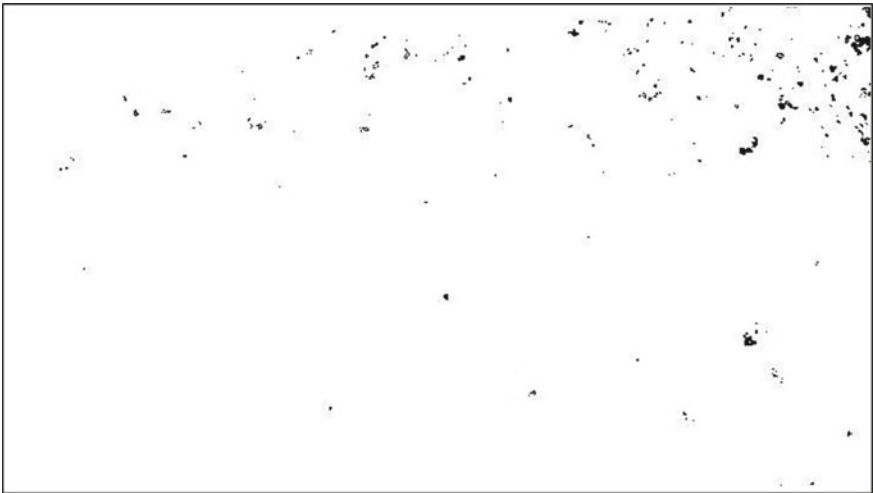


Fig. 7.9 Result of brain cortex sample (2)

7.6 Comparison with Other Techniques

We bring comparison of basic methods. Very deep comparison of graph cuts method and levels set methods was already done by authors: Boykov and Funka Lea [24, 44], as well as in another article [44]. They did very deep analyses based on artificial and as well real data. They studied this problem for very long time. In summary, we can tell that both methods have very good result input, they give similar level of results

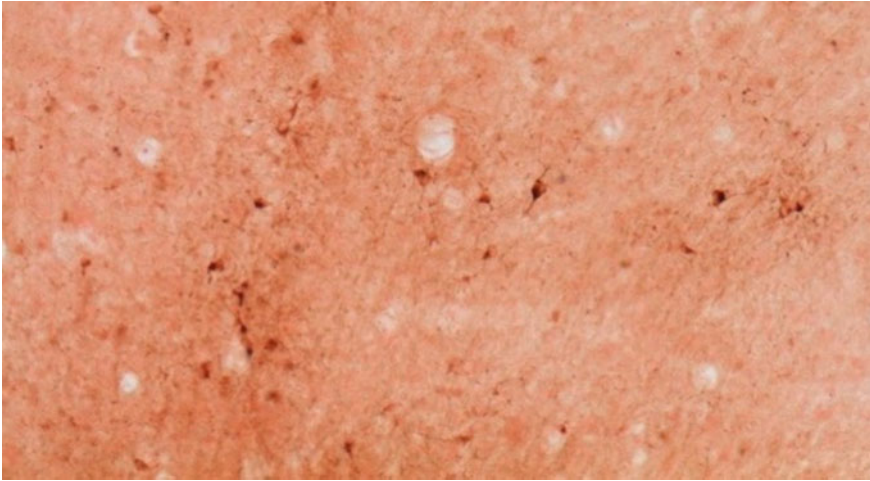


Fig. 7.10 Brain cortex sample (3)

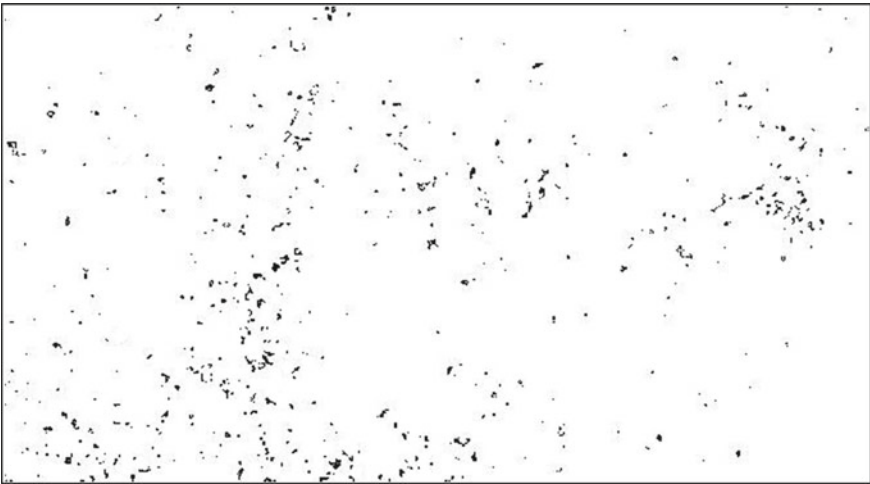


Fig. 7.11 Result of brain cortex sample (3)

and also similar quality of segmentations. Comparisons given by other techniques can be visible in our experiments.

The high effectiveness of the new technique can be further evaluated when compared with other already established techniques that can accomplish the same segmentation. Here, we have performed a comparative study of 3 other techniques: namely the simple threshold technique [45], the Otsu's binarization technique [45] and the adaptive thresholding technique [45] with our devised technique: the RGB based multi-spectral thresholding technique.

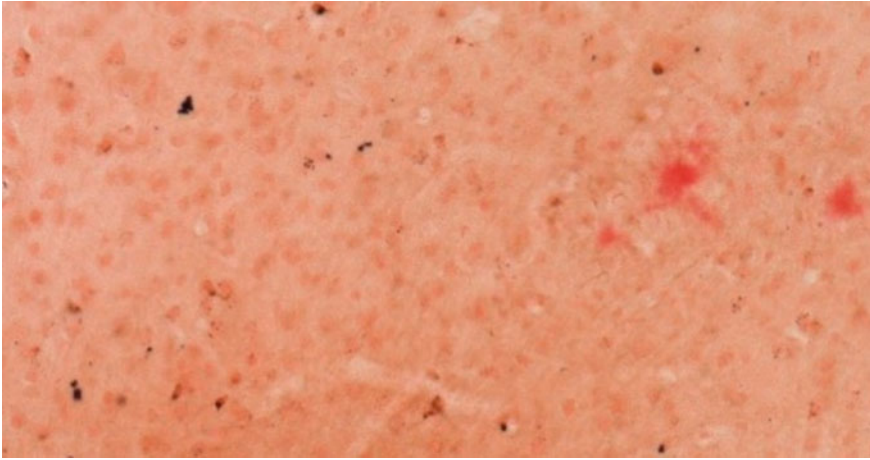


Fig. 7.12 Brain cortex sample (4)

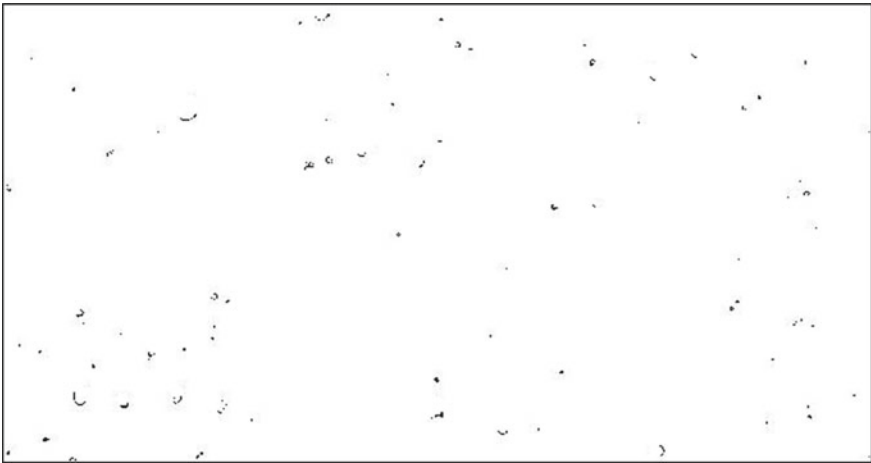


Fig. 7.13 Result of brain cortex sample (4)

Given below, are the respective iron percentages detected for each of the samples, using the 4 techniques (Table 7.2).

Moreover, we have taken the sample 3 as an example to pictorially represent the contrast in effectiveness between the 4 techniques (Figs. 7.20, 7.21, 7.22 and 7.23).

Hence, we see that our algorithm is producing results with much higher accuracy and precision than the already established methods.

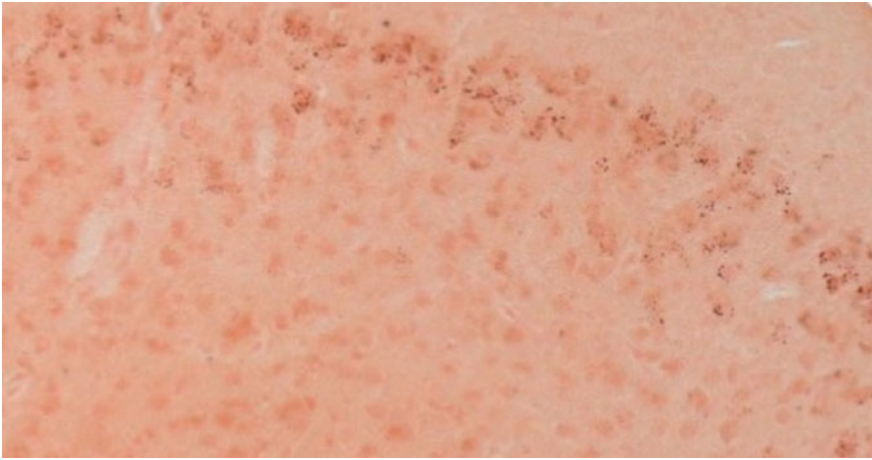


Fig. 7.14 Brain cortex sample (5)

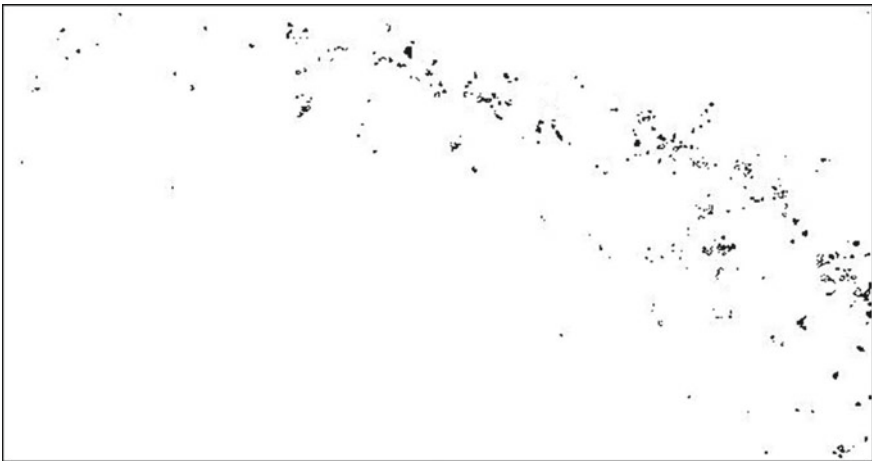


Fig. 7.15 Result of brain cortex sample (5)

7.7 Discussion

Due to our used methods, we confirmed iron changes in the brain cortex in different ages and also, we could see changes in iron amounts between animals with Alzheimer disease and control animals. It is clear, that changes in iron metabolism play a role in this disease, therefore, we suggest that iron is adequate to early diagnostic and that our work could help to better understand to changes in the iron level in the brain. With early diagnostic, patients could receive cure earlier and their life could be prolonged.

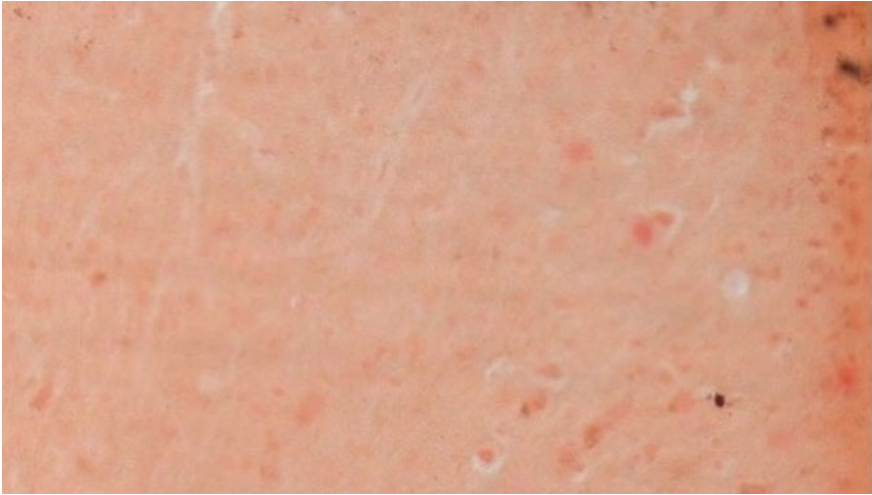


Fig. 7.16 Brain cortex sample (6)

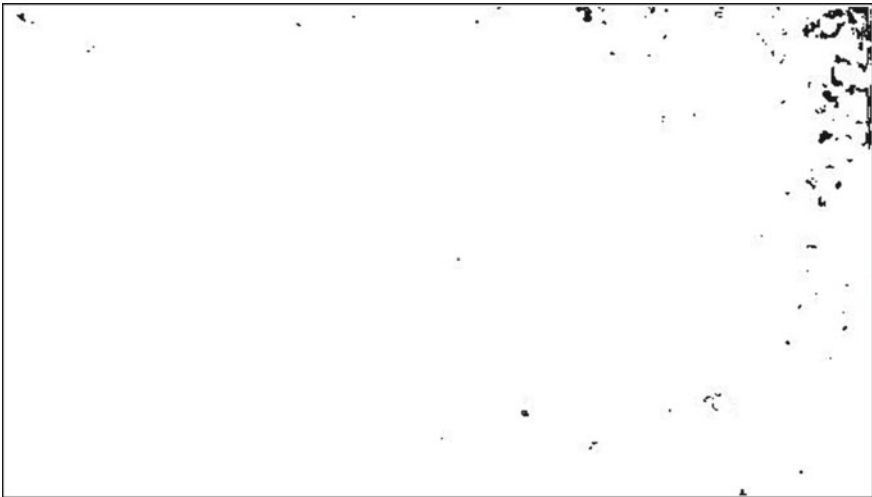


Fig. 7.17 Result of brain cortex sample (6)

Our method can be applied also in different medical problem and in various medical analysis, where image data analysis is essential.

We conclude that in the first phase of preprocessing the usual threshold fail and was not enough for pre-processing of data. Therefore, we needed and discovered a new approach how to pre-process data. We used a combination of threshold and consequently a special case of clustering method. After that application of the graph cutting better results are given.

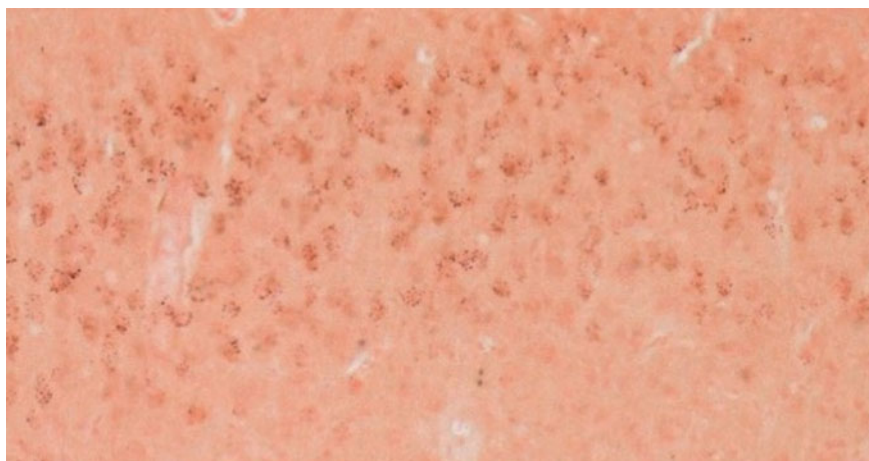


Fig. 7.18 Brain cortex sample (7)

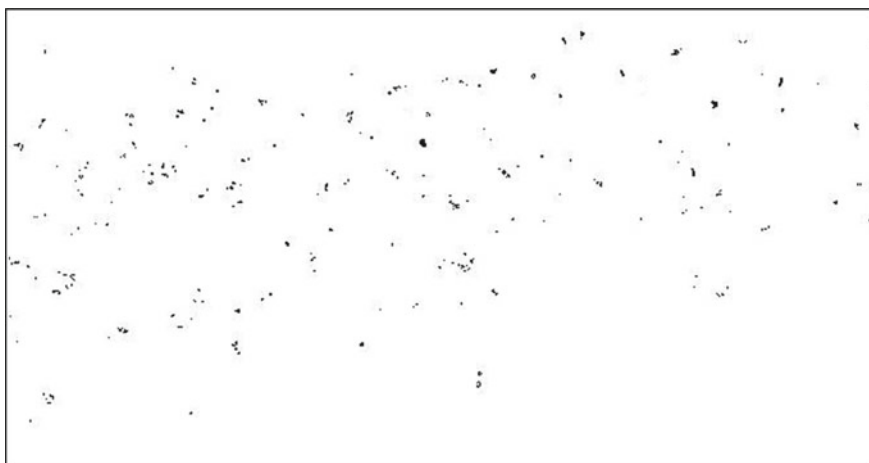


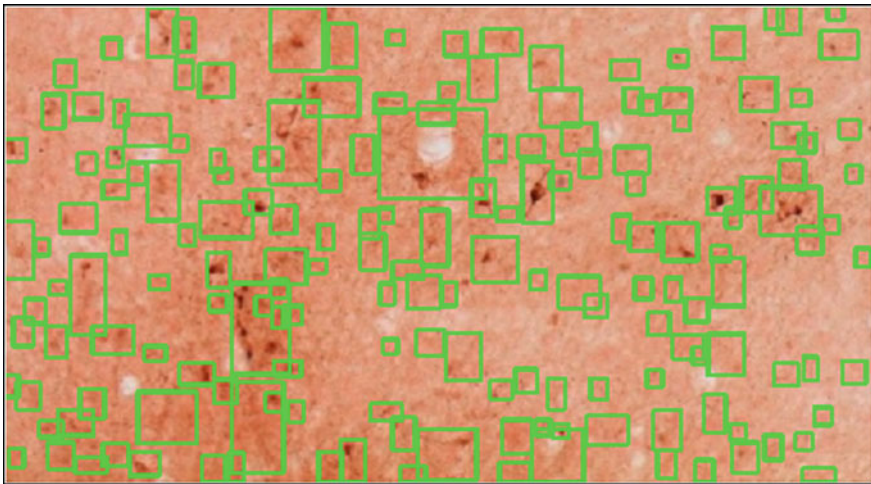
Fig. 7.19 Result of brain cortex sample (7)

Table 7.1 Pixel analysis of the samples

Sample	Total image area (square units)	Net detected area (square units)	Percentage (%)
1	1,651,320	7445	0.451
2	1,651,320	4105	0.249
3	1,651,692	9628	0.583
4	1,645,020	3816	0.232
5	1,649,096	5032	0.305
6	1,644,648	4587	0.279
7	1,652,432	6423	0.389

Table 7.2 Comparative study of 3 established method with our new method

Sample no	Simple thresholding	Otsu's binarization technique	Adaptive thresholding	RGB based multi-spectral thresholding
1	31.175	8.033	0.858	0.451
2	26.392	6.811	2.052	0.249
3	34.452	8.396	4.325	0.583
4	25.989	7.898	1.459	0.232
5	23.798	7.492	1.384	0.305
6	16.261	3.041	1.2	0.279
7	25.638	7.722	0.753	0.389

**Fig. 7.20** Segmentation of Sample 3 using simple thresholding technique

Finally, we were able to present quantitative evaluation of presenting iron in the samples. In the future we want also studying and extend the clustering method for segmentation and also its connection to aggregation function, aggregations linked to cluster used for image analyses.

We conclude that graph theory approach plays an important role in image processing of real data and it needs to be considered among other computer methods. We showed important application of this approach in the bio-medical research in the analyzing of bio-medical data acquired by light microscopy.

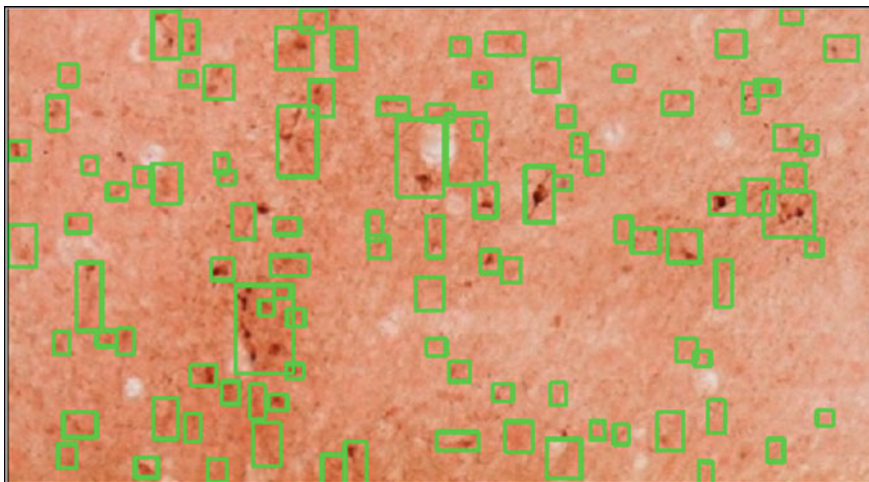


Fig. 7.21 Segmentation of Sample 3 using Otsu's binarization

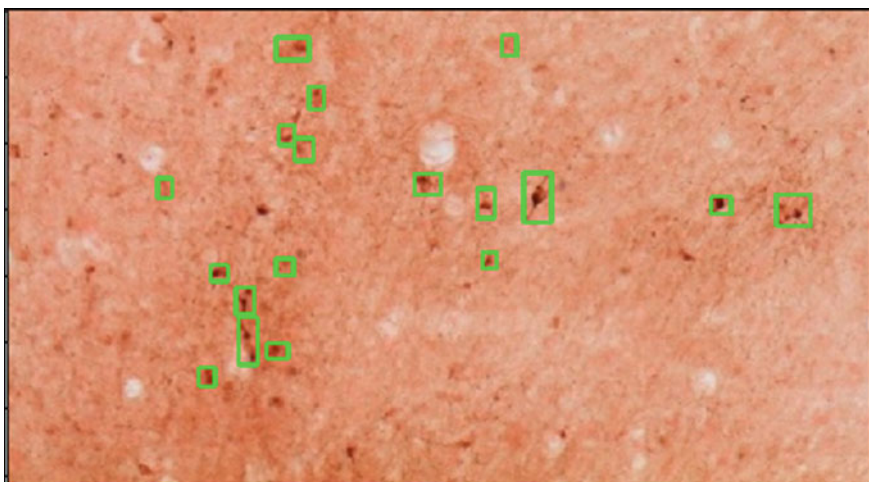


Fig. 7.22 Segmentation of Sample 3 using adaptive thresholding technique

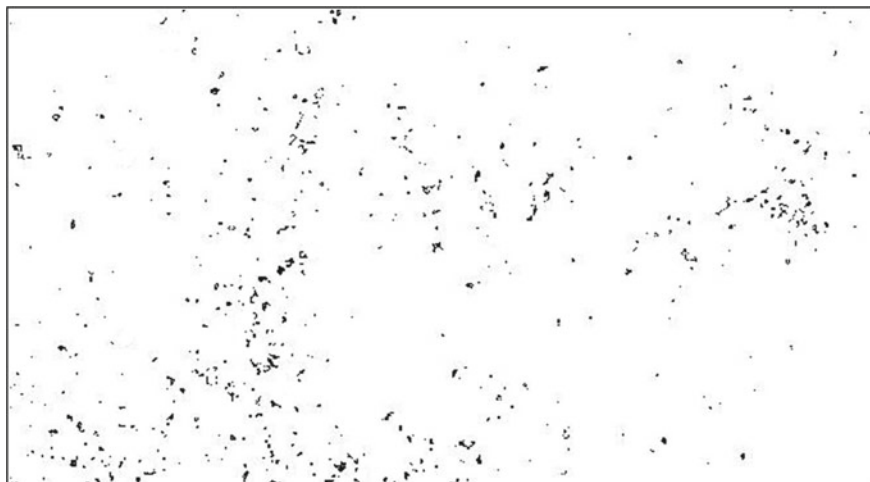


Fig. 7.23 Segmentation of Sample 3 using our devised technique

Acknowledgements M. Ždímalová acknowledges the Scientific Slovak Grant VEGA no. 1/0006/19. This study was accomplished with financial support of Slovak Research and Development Agency APVV-16-0039.

References

1. Stockman, G.C., Shapiro Linda, G.: A text book on “Computer Vision”, pp. 279–325. Prentice Hall, Hoboken (2001)
2. Zhang, D., Islam, M., Lu, G.: A review on automatic image image annotation techniques. *Pattern Recogn.* **45**(1), 346–362 (2012)
3. Aisen, P.S., Cummings, J., Jack, C.R., Morris, J.C., Sperling, R., Frölich, L., Jones, R.W., Dowsett, S.A., Matthews, B.R., Raskin, J., Scheltens, P., Dubois, B.: On the path to 2025: understanding the Alzheimer’s disease continuum. *Alzheimers. Res. Ther.* **9**, 60 (2017)
4. Andersen, H.H., Johnsen, K.B., Moos, T.: Iron deposits in the chronically inflamed central nervous system and contributes to neurodegeneration. *Cell. Mol. Life Sci.* **71**, 1607–1622 (2014)
5. Başar, E., Düzgün, A.: How is the brain working? Research on brain oscillations and connectivity in a new “Take-Off” state (2016)
6. Fernandes, L., Wang, H.: Editorial: mood and cognition in old age (2018)
7. Rao, R., Tkac, I., Unger, E.L., Ennis, K., Hurst, A., Schallert, T., Connor, J., Felt, B., Georgieff, M.K.: Iron supplementation dose for perinatal iron deficiency differentially alters the neurochemistry of the frontal cortex and hippocampus in adult rats. *Pediatr. Res.* **73**, 31–37 (2013)
8. Caselli, R.J., Beach, T.G., Knopman, D.S., Graff-Radford, N.R.: Alzheimer disease: scientific breakthroughs and translational challenges. *Mayo Clin. Proc.* **92**, 978–994 (2017)
9. Dubois, B., Hampel, H., Feldman, H.H., Scheltens, P., Aisen, P., Andrieu, S., Bakardjian, H., Benali, H., Bertram, L., Blennow, K., Broich, K., Cavado, E., Crutch, S., Dartigues, J.F., Duyckaerts, C., Epelbaum, S., Frisoni, G.B., Gauthier, S., Genthon, R., Gouw, A.A., Habert,

- M.O., Holtzman, D.M., Kivipelto, M., Lista, S., Molinuevo, J.L., O'Bryant, S.E., Rabinovici, G.D., Rowe, C., Salloway, S., Schneider, L.S., Sperling, R., Teichmann, M., Carrillo, M.C., Cummings, J., Jack, C.R.: Preclinical Alzheimer's disease: definition, natural history, and diagnostic criteria. *Alzheimers Dement.* **12**(3), 292–323 (2016)
10. Grasso, M., Piscopo, P., Confaloni, A., Denti, M.A.: Circulating miRNAs as biomarkers for neurodegenerative disorders. *Molecules* 6891–6910 (2014)
 11. Falangola, M.F., Lee, S.P., Nixon, R.A., Duff, K., Helpner, J.A.: Histological co-localization of iron in A β plaques of PS/APP transgenic mice. *Neurochem. Res.* **30**, 201–205 (2005)
 12. Singh, N., Haldar, S., Tripathi, A.K., Horback, K., Wong, J., Sharma, D., Beserra, A., Suda, S., Anbalagan, C., Dev, S., Mukhopadhyay, C.K., Singh, A.: Brain iron homeostasis: from molecular mechanisms to clinical significance and therapeutic opportunities. *Antioxid. Redox Signal.* **20**, 1324–1363 (2014)
 13. Dobson, J.: Nanoscale biogenic iron oxides and neurodegenerative disease. *FEBS Lett.* **496**, 1–5 (2001)
 14. Marcus, C., Mena, E., Subramaniam, R.M.: Brain PET in the diagnosis of Alzheimer's disease. *Clin. Nucl. Med.* **39**, 413–426 (2014)
 15. <https://imagej.net/Citing>
 16. <https://cellprofiler.org/citations/>
 17. <https://www.moleculardevices.com/products/cellular-imaging-systems/acquisition-and-analysis-software/metamorph-microscopy#pref>
 18. <https://www.moleculardevices.com/products/cellular-imaging-systems/acquisition-and-analysis-software/metamorph-microscopy>
 19. <https://www.ilastik.org/>
 20. Jankowsky, J.L., Fadale, D.J., Anderson, J., Xu, G.M., Gonzales, V., Jenkins, N.A., Copeland, N.G., Lee, M.K., Younkin, L.H., Wagner, S.L., Younkin, S.G., Borchelt, D.R.: Mutant presenilins specifically elevate the levels of the 42 residue beta-amyloid peptide in vivo: evidence for augmentation of a 42-specific gamma secretase. *Hum. Mol. Genet.* **13**, 159–170 (2004)
 21. Webster, S.J., Bachstetter, A.D., Nelson, P.T., Schmitt, F.A., Van Eldik, L.J.: Using mice to model Alzheimer's dementia: an overview of the clinical disease and the preclinical behavioral changes in 10 mouse models. *Front. Genet.* **5**, 1–23 (2014)
 22. Meguro, R., Asano, Y., Odagiri, S., Li, C., Iwatsuki, H., Shoumura, K.: Nonheme-iron histochemistry for light and electron microscopy: a historical, theoretical and technical review. *Arch. Histol. Cytol.* **70**, 1–19 (2007)
 23. Wilcock, D.M., Gordon, M.N., Morgan, D.: Quantification of cerebral amyloid angiopathy and parenchymal amyloid plaques with Congo red histochemical stain. *Nat. Protoc.* **1**, 1591–1595 (2006). <https://doi.org/10.1038/nprot.2006.277>
 24. Boykov, Y., Funka-Lea, Y.G.: Graph cuts and efficient N-D image segmentation. *Int. J. Comput. Vision* **70**(2), 109–131 (2006)
 25. Boykov, Y., Veksler, O.: Graph cuts in vision and graphics: theories and applications. In: *Handbook of Mathematical Models on Computer Vision*, pp. 100–118. Springer, New York (2006)
 26. Basvapasrad, D., Hegadi, R.S.: A survey on traditional and graph theoretical techniques for image segmentation. *Int. J. Comput. Appl. Recent Adv. Inf. Technol.* 38–46 (2014)
 27. Yi, F., Moon, I.: Image segmentation: a survey of graph-cut methods. In: *IEEE International Conference on Systems and Informatics (ICSAI)*, pp. 193–194 (2012)
 28. Ford, L.R., Jr., Fulkerson, D.R.: Maximal flow through a network. *Can. J. Math.* **8**, 399–404 (1956)
 29. Goldberg, A.V., Tajan, R.E.: A new approach to the maximum flow problem. *J. ACM* **35**(4), 921–940 (1988)
 30. Geman, S., Geman, D.: Stochastic relaxation, gibbs distribution and the bayesian restoration of images. *IEEE Trans. Pattern Anal. Mach. Intell.* **6**(6), 721–741 (1984)
 31. Jiang, X., Zhang, R., Nie, S.: Image segmentation based on level set method. *Phys. Procedia* **33**, 840–845 (2012)

32. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell.* **26**(2), 147–159 (2004)
33. Xy Peng, B., Zhang, L., Zhang, D.: A survey of graph theoretical approaches to image segmentation. *Annu. Rev. Biomed. Eng.* 315–337 (2000)
34. Callara, A.L., Magliaro, Ch., Ahluwalia, A., Vanello, N.: A smart region-growing algorithm for single-neuron segmentation from confocal and 2-photon datasets Italy. *Front. Neuroinformatics* 8–12. <http://doi.org/10.3389/fninf.2020.00009> (2020)
35. Price, R., Ohlander, K., Reddy, K., Ra, D.: Picture segmentation using a recursive region splitting method. *Comput. Graph. Image Process. (CGIP)* 313–333 (1978)
36. Pavlidis, T., Horowitz, S.L.: A Picture Segmentation by a Directed Split and Merge Procedure. *ICPR, Denmark*, pp. 424–433 (1974)
37. Magzhan, K., Matjani, H.: A review and evaluations of shortest path algorithm. *Int. J. Sci. Technol. Res.* **2013**, 6 (2013). ISSN 2277-8616
38. Maeda, J., Ishikawa, C., Novianto, S., Tadehara, N., Suzuki, Y.: Rough and accurate segmentation of natural color images using fuzzy region-growing algorithm. In: *Proceedings 15th International Conference on Pattern Recognition*, vol. 3, pp. 638–641 (2000)
39. Moghaddamzadeh, A., Bourbakis, N.: A fuzzy region growing approach for segmentation of colour images. *Pattern Recogn.* **30**, 867–881 (1997)
40. Shruti, J., Salau, A.O.: An image feature selection approach for dimensionality reduction based on kNN and SVM for AkT proteins, electrical & electronic engineering. *Cogent Eng.* **6**, 1599537, 1–14 (2019). <http://doi.org/10.1080/23311916.2019.1599537>
41. Boykov, Y., Jolly, M., P.: Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. *Proc. Image Segmentations Int. J. Comput. Vis.* **2**, 109–131 (2006)
42. Loucký, J., Oberhuber, T.: Graph Cuts in Segmentation of a Left Ventricle from MRI Data. *Czech Technical University in Prague, Prague* (2010)
43. Ždímalová, M., Krivá, Z., Bohumel, T.: Graph cuts in image processing. In: *APLIMAT Proceeding* (2015)
44. https://www.csd.uwo.ca/~yboykov/Presentations/ECCV06_tutorial_partIIIa_vnk.pdf
45. https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_thresholding/py_thresholding.html
46. Daglas, M., Adlard, P.A.: The involvement of iron in traumatic brain injury and neurodegenerative disease. *Front. Neurosci.* **12** (2018)
47. Gerlach, M., Ben-Shachar, D., Riederer, P., Youdim, M.B.H.: Altered brain metabolism of iron as a cause of neurodegenerative diseases? *J. Neurochem.* **63**, 793–807 (2002)
48. Gong, N., Dibb, R., Bulk, M., van der Weerd, L., Liu, C.: Imaging beta amyloid aggregation Iron accumulation in Alzheimer’s disease using quantitative susceptibility mapping MRI. *Neuroimage* **191**, 176–185 (2019)
49. Piñero, D.J., Connor, J.R.: Iron in the brain: an important contributor in normal and diseased states. *Neuroscience* **6**, 435–453 (2000)
50. https://www.csd.uwo.ca/~yboykov/Presentations/ECCV06_tutorial_partI_yuri.pdf

Chapter 8

On the Development of Directed Acyclic Graphs in Differential Diagnostics of Pulmonary Diseases with the Help of Arterial Oscillogram Assessment



V. P. Martsenyuk, D. V. Vakulenko, L. A. Hryshchuk, L. O. Vakulenko, N. O. Kravets, and N. Ya. Klymuk

Abstract Directed acyclic graphs (DAGs) were used to construct differentiation pathways for previously diagnosed pulmonary and cardiovascular diseases. For this purpose, we have used the indicators obtained from arterial oscillograms registered at the measurement of blood pressure, identified additional sequences of indicators and evaluated their significance. The routes of the graph are characterized with the help of the probabilities determined with the help of C5.0 algorithm. The highest probability of the constructed routes was found in case of chronic obstructive pulmonary disease.

Keywords Directed acyclic graphs · Differential diagnostics · Arterial oscillography · Pulmonary diseases · Tuberculosis · Cardiovascular diseases

8.1 Introduction

In many areas of science and technology, the concept of hierarchy, hierarchical structure plays an important role. For such areas as systems theory, decision making, it is typical to represent the object of research (system) in the form of a structure of subordination of some elements of the system to others. Directed acyclic graphs are a topological model of hierarchical structures.

Directed graph G is an ordered pair $G = \langle V, A \rangle$, where V is the set of vertices or nodes, $A = \{(v_1, v_2) : v_1, v_2 \in V\}$ is the set of ordered pairs of different vertices, called arcs or oriented edges. The presence in the set A of the vector (v_1, v_2) corresponds to the presence of a connection (arc) between the vertices v_1 and v_2 of the graph. If a directed graph contains no cycles (sequences of arcs of the form $(v_1, v_2), \dots, (v_n, v_1)$), then it is called acyclic one.

V. P. Martsenyuk (✉)
University of Bielsko-Biala, Bielsko-Biala, Poland

D. V. Vakulenko · L. A. Hryshchuk · L. O. Vakulenko · N. O. Kravets · N. Ya. Klymuk
Ternopil National Medical University, Ternopil, Ukraine

A directed acyclic graph (DAG) is a directed graph in which there are no directed cycles, but there may be “parallel” paths going out from one node and coming to the final node in different ways. A directed acyclic graph is a generalization of a tree (more precisely, their union is a forest). Directed acyclic graphs are widely used in applications in medical diagnostics, since they represent artificial neural networks without feedback. An important special case of a directed acyclic graph is the tree, which is important for the models of decision making.

Tuberculosis is an infectious disease caused by mycobacterium of tuberculosis and characterized by the formation of specific granules in various organs and tissues (most often in the lungs) and a polymorphic clinical picture. At the same time, tuberculosis is a global disease that occurs in every country in the world. It is the leading infectious cause of death worldwide. The World Health Organization estimates that 1.8 billion people—about a quarter of the world’s population—are infected with mycobacterium tuberculosis. Last year, 10 million became ill with tuberculosis, 1.5 million died. Tuberculosis is responsible for the economic devastation and the onset of poverty and disease, affecting families, communities and even entire countries. Among the most vulnerable are women, children and HIV/AIDS infected [1]. The resistance of mycobacteria to available medicines is increasing, which means that the disease is more difficult to treat. Currently, tuberculosis patients can be classified as at risk for Coronavirus COVID-19. It is known that almost all organs and systems of the human body, including the vascular system, are affected by tuberculosis. Temporary and differential diagnostics of tuberculosis are important [2]. Differential diagnostics of pulmonary tuberculosis is performed with other pulmonary diseases, namely, nonspecific and viral pneumonia, chronic obstructive pulmonary disease, and lung injury in cardiovascular diseases.

In the absence of a randomized controlled trial (RCT), big data analysis methods can be used to differentiate various pathologies, early diagnosis, and evaluate the effectiveness of therapeutic intervention, and provide useful evidence when making health care decisions [3–11]. Even when the results of direct diagnostic methods are convincing, the use of Direct Acyclic Graphs (DAG) methods can provide a more accurate assessment and understanding of the patterns of disease course [3–8]. If the evidence base consists of an RCT network that includes diagnosis by accepted direct methods (laboratory or hardware) or (and) mediated, the results can be synthesized using by means of a network meta-analysis [11–16].

Purpose of the research: to construct DAGs using the indicators obtained in the analysis of arterial oscillograms [17, 18], registered when measuring blood pressure; identify significant additional diagnostic routes and indicators, evaluate their severity, and provide biological interpretation for individuals with previously diagnosed respiratory diseases (137 individuals), cardiovascular (322 individuals), and healthy (471 individuals).

8.2 Materials and Methods

Totally 960 people were under the study. The selection consisted of the following survey categories: respiratory disease (137 individuals), cardiovascular disease (322 individuals), and healthy (471 individuals). The group with pulmonary diseases included persons with active form of tuberculosis (84 persons): of different degree of complexity (with lesions of one and two lungs), chemo resistant tuberculosis, of different sex (men and women), with bad habits (smoking, no smoking) patients with chronic obstructive pulmonary disease (45 people), pneumonia (8 people), patients with cardiovascular disease were mainly diagnosed with early clinical manifestations of coronary heart disease (255 people), and arterial hypertension (67 people).

Arterial oscillograms were recorded while measuring blood pressure, during compression growth, using the electronic tonometer VAT 41-2, (the cuff was applied to the shoulder). Further analysis of the AO was performed using the methods proposed by D. V. Vakulenko, L. O. Vakulenko [11, 17, 18]. The structure of attributes that were used in data mining algorithms is presented in Table 8.1.

DAG is a graphical structure consisting of a set of corresponding nodes, each of which is associated with a random variable and corresponding arrows connecting the nodes, which reflect indicators that are key in the differentiation of pathological conditions [3–8].

Our approach is based on describing the routes in DAG with the help of probabilities obtained as a result of applying C5.0 algorithm [11] for the attributes in Table 8.1.

Table 8.1 Attributes for DAG from the indices of arterial oscillography

Denotation of attribute	Specification of attribute
A1...A9	Morphological analysis
A10	Fractal dimension
A11...A49	Temporal analysis
A50...A475	Spectral analysis
A50...A335	Power spectrum with Fourier transform of oscillations
A406...A455	Power spectrum with Fourier transform of intervalogram
A336...A405	Power of current frequency and phase due to Hilbert-Huang oscillation
A456...A475	Power of current frequency and phase due to Hilbert-Huang transform of intervalogram

8.3 Results and Discussion

Analyzing the Directed acyclic graphs (DAGs) based on blood pressure measurements, with subsequent analysis of the indicators (attributes) obtained from arterial oscillograms in a group of patients 1 lung tuberculosis (31 person), 2-lung tuberculosis (43 person), active tuberculosis (24 person) (Fig. 8.1).

Consider possible routes for identifying tuberculosis patients with two affected lungs.

7 different attributes were needed to differentiate one lung tuberculosis (Teta_per-100-70, LF/HF-100-70, Delta_per_70-end, LF_per_int_p, Hurst-total, S-Hil-HFx13x15_overal-70-end, LF/HF_20-70). In 93.55% (29 people) patients were differentiated correctly, and 6.45% (2 persons) were counted to patients with active tuberculosis.

Consider possible routes for identifying tuberculosis patients with one affected lung.

1. The route consists of 3 nodes S-Hil-HFx25x30_per-total (0.32) → M1 (0.52) → Teta_per-100-70 (0.29) → 1 lung tuberculosis (1), way total probability—0.048

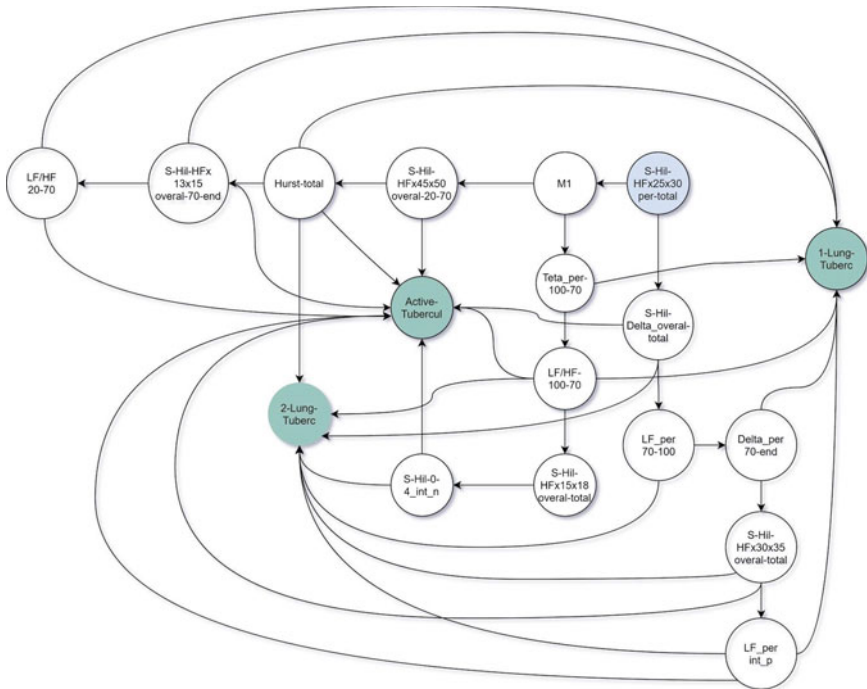


Fig. 8.1 DAG reflecting dependencies between patients 1—lung tuberculosis (31 person), 2—lung tuberculosis (43 person), active tuberculosis (24 person)

2. The route consists of 4 nodes S-Hil-HFx25x30_per-total (0.32) → M1(0.52) → Teta_per-100-70 (0.29) → LF/HF-100-70 (0.077) → 1 lung tuberculosis (1), way total probability—0.004
3. The route consists of 4 nodes S-Hil-HFx25x30_per-total (0.32) → S-Hil-Delta_overal-total (0.104) → LF_per-70-100 (0.142) → Delta_per-70-end (0.167) → 1 lung tuberculosis (1), way total probability—0.001
4. The route consists of 4 nodes S-Hil-HFx25x30_per-total (0.32) → M1 (0.52) → S-Hil-HFx45x50_overal-20-70 (0.63) → Hurst-total (0.7) → 1 lung tuberculosis (0.29), way total probability—(0.021)
5. The route consists of 5 nodes S-Hil-HFx25x30_per-total (0.32) → M1 (0.52) → S-Hil-HFx45x50_overal-20-70 → Hurst-total (0.7) → S-Hil-HFx13x15_overal-70-end (0.83) → 1 lung tuberculosis (0.05), way total probability—0.03
6. The route consists of 6 nodes S-Hil-HFx25x30_per-total (0.32) → M1 (0.52) → S-Hil-HFx45x50_overal-20-70 (0.63) → Hurst-total (0.7) → S-Hil-HFx13x15_overal-70-end (0.83) → LF/HF-20-70 (0.94) → 1 lung tuberculosis (1), way total probability—0.057
7. The route consists of 6 nodes S-Hil-HFx25x30_per-total (0.32) → S-Hil-Delta_overal-total (0.104) → LF_per-70-100 (0.14) → Delta_per-70-end (0.17) → S-Hil-HFx30x35_overal-total (0.17) → LF_per_int_p (0.107) → 1 lung tuberculosis (0.43), way total probability—0.000006.

The highest probability of detecting tuberculosis patients with one affected lung was found in the sixth route (0.057). A key endpoint was the assessment of the ability of the autonomic nervous system to maintain the balance of sympathetic and parasympathetic units during adaptation at the onset of LF/HF-20-70 shoulder compression. When LF HF-20-70 values ≤ 7.97 (16 people), tuberculosis patients with one affected lung, and LF/HF-20-70 > 7.97 patients with tuberculosis with two affected lungs.

When differentiating 2-lung tuberculosis, 8 finite attributes were needed (LF_per_int_p, LF/HF-20-70, Delta_per-70-end, LF/HF-100-70, Hurst-total, S-Hil-HFx45x50_overal-20-70, LF_per-70-100, S-Hill-Delta_overal-total). In 90.70% (39 people) patients were differentiated correctly, 2.33% (1 person) was counted to patients with one lung tuberculosis, 6.98% (3 persons)—to patients with active tuberculosis.

Consider possible routes for identifying tuberculosis patients with two affected lungs.

1. The route consists of 2 nodes S-Hil-HFx25x30_per-total (0.44) → S-Hil-Delta_overal-total (0.54) → 2 lung tuberculosis (0.92), way total probability—0.219
2. The route consists of 3 nodes S-Hil-HFx25x30_per-total (0.44) → S-Hil-Delta_overal-total (0.54) → LF_per-70-100 (0.4) → 2 lung tuberculosis (1), way total probability—0.217

3. The route consists of 3 nodes S-Hil-HFx25x30_per-total (0.44) → M1 (0.34) → S-Hil-HFx45x50_overal-20-70 (0.18) → 2 lung tuberculosis (1), way total probability—0.062
4. The route consists of 4 nodes S-Hil-HFx25x30_per-total (0.44) → M1 (0.34) S-Hil-HFx45x50_overal-20-70 (0.18) → Hurst-total (0.1) → 2 lung tuberculosis(0.29), way total probability—0.001
5. The route consists of 4 nodes S-Hil-HFx25x30_per-total (0.44) → M1 (0.34) → Teta_per-100-70 (0.65) → LF/HF-100-70 (0.85) → 2 lung tuberculosis (0.92), way total probability—0.075
6. The route consists of 6 nodes S-Hil-HFx25x30_per-total (0.44) → S-Hil-Delta_overal-total (0.54) → LF_per-70-100 (0.4) → Delta_per-70-end (03) → S-Hil-HFx30x35_overal-total (0.32) → LF_per_int_p (0.42) → 2 lung tuberculosis (0.58), way total probability—0.002
7. The route consists of 6 nodes S-Hil-HFx25x30_per-total (0.44) → S-Hil-Delta_overal-total (0.54) → LF_per-70-100 (0.4) → Delta_per-70-end (03) → S-Hil-HFx30x35_overal-total (0.32) → LF_per_int_p (0.42) → 2 lung tuberculosis (0.143), way total probability—0.001.

The highest probability of detecting tuberculosis patients with two affected lungs was found in the first route (0.219). A key endpoint was the assessment of instantaneous restorative capacity during adaptation to S-Hil-Delta_overal-total shoulder compression by the instantaneous Hilbert-Huang transformation in the range from 0 to 4 Hz [18, 19].

To differentiate the sheet that responds to patients with active tuberculosis, 7 finite attributes were needed (S-Hil-Delta_overal-total, Hurst-total, LF/HF-100-70, LF/HF-100-70, S-Hil-HFx13x15_overal-70-end, S-Hil-HFx30x35_overal-total, LF_per_int_p). 45.83% (11 people) were differentiated patients correctly, 29.17% (7 persons) were counted to patients with 2 lung tuberculosis, 25% (6 persons) were counted to patients with 1 lung tuberculosis.

Consider possible routes for the identification of patients with active pulmonary tuberculosis with two affected lungs.

1. The route consists of 2 nodes S-Hil-HFx25x30_per-total (0.24) → S-Hil-Delta_overal-total(0.35) → Active tuberculosis (0.077), way total probability—0.0067
2. The route consists of 4 nodes S-Hil-HFx25x30_per-total (0.24) → M1 (0.14) → S-Hil-HFx45x50_overal-20-70 (0.18) → Hurst-total (0.2) → Active tuberculosis (0.43), way total probability—0.0005
3. The route consists of 4 nodes S-Hil-HFx25x30_per-total (0.24) → M1 (0.14) → Teta_per-100-70 (0.059) → LF/HF-100-70 (0.077) → Active tuberculosis (0.083), way total probability—0.00001
4. The route consists of 5 nodes S-Hil-HFx25x30_per-total (0.24) → M1 (0.14) → S-Hil-HFx45x50_overal-20-70 (0.18) → Hurst-total (0.2) → S-Hil-HFx13x15_overal-70-end (0.13) → Active tuberculosis (0.5), way total probability—0.0001

5. The route consists of 5 nodes S-Hil-HFx25x30_per-total (0.24) → S-Hil-Delta_overal-total (0.35) → LF_per-70-100 (0.46) → Delta_per-70-end (0.53) → S-Hil-HFx30x35_overal-total (0.57) → Active tuberculosis (0.89), way total probability—0.01
6. The route consists of 6 nodes S-Hil-HFx25x30_per-total (0.24) → S-Hil-Delta_overal-total (0.35) → LF_per-70-100 (0.46) → Delta_per-70-end (0.53) → S-Hil-HFx30x35_overal-total (0.57) → LF_per_int_p (0.42) → Active tuberculosis (0.41), way total probability—0.0021
7. The route consists of 6 nodes S-Hil-HFx25x30_per-total (0.24) → S-Hil-Delta_overal-total (0.35) → LF_per-70-100 (0.46) → Delta_per-70-end (0.53) → S-Hil-HFx30x35_overal-total (0.57) → LF_per_int_p (0.42) → Active tuberculosis (0.43), way total probability—0.0022.

The highest probability of detecting patients with active tuberculosis was found in the fifth route (0.0107–8 people). A key endpoint was the evaluation of the instantaneous restorative ability of the braking and pulse balance during adaptation to the S-Hil-HFx30x35_overal-total shoulder compression, calculated at an instantaneous Hilbert-Huang transformation in the range from 30 to 35 Hz [18, 20].

Variable rank of indicators in differentiation of the studied conditions was as follows: S-Hil-HFx13x15_overal-20-70 (1.0), Hurst-total (0.99), S-Hil-HFx60_overal-20-70 (0.98), Alpha_per-100-70 (0.98).

Analyzing the Directed acyclic graphs (DAGs) based on blood pressure measurements, with subsequent analysis of the indicators (attributes) obtained from arterial oscillograms in a group of patients with chronic obstructive pulmonary disease (45 persons), active form of pulmonary tuberculosis (24 persons), chemo resistant pulmonary tuberculosis (14 persons) (Fig. 8.2).

Consider possible routes for identifying patients with chronic obstructive pulmonary disease

1. The route consists of 1 node S-Hil-HFx45x50_per (0.54) → Chronic obstructive pulmonary disease (1), way total probability (0.54)

Consider possible routes to identify patients with active pulmonary tuberculosis

2. The route consists of 3 nodes S-Hil-HFx18x21_overal-20-70 (0.29) → S-Hil-HFx35x40_overal-20 (0.63) → S-Hil-HFx18x21_overal-20-70 (0.15) → Active pulmonary tuberculosis (1), way way total probability probability (0.03)
3. The route consists of 3 nodes S-Hil-HFx18x21_overal-20-70 (0.29) → S-Hil-HFx35x40_overal-20 (0.63) → HFx25x30_per-20-70 (0.88) → Active pulmonary tuberculosis (1), way total probability (0.16)

Consider possible routes to identify patients with chemoresistant pulmonary tuberculosis

4. The route consists of 3 nodes S-Hil-HFx18x21_overal-20-70 (0.17) → S-Hil-HFx35x40_overal-20 (0.37) → S-Hil-HFx18x21_overal-20-70 (0.85) → Chemoresistant pulmonary tuberculosis (1), way total probability (0.05)

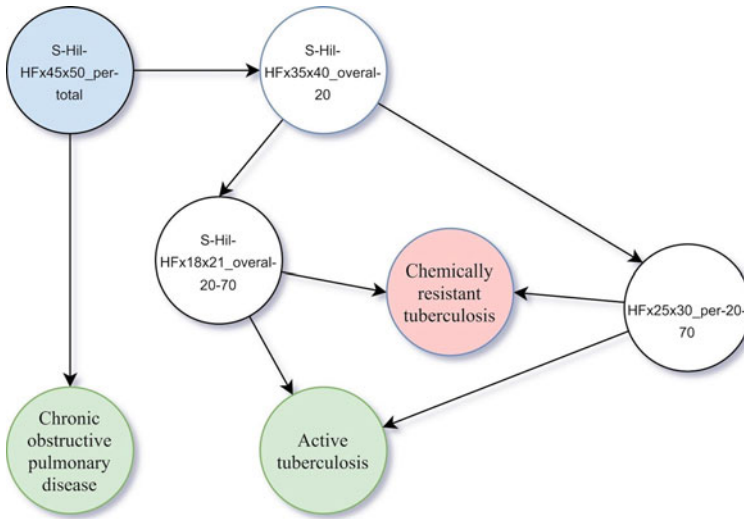


Fig. 8.2 DAG reflecting dependencies between patients with chronic obstructive pulmonary disease (45 persons), active form of pulmonary tuberculosis (24 persons), chemoresistant pulmonary tuberculosis (14 persons)

5. The route consists of 3 nodes S-Hil-HFx18x21_overal-20-70 (0.17) → S-Hil-HFx35x40_overal-20 (0.37) → HFx25x30_per-20-70 (0.12) → Chemoresistant pulmonary tuberculosis (1), way total probability (0.01).

The highest probability of detecting patients with chronic obstructive pulmonary disease was found in the first route, way total probability 0.54 (45 people).

A key endpoint was the evaluation of the instantaneous restorative ability of the braking and excitation balance during adaptation to S-Hil-HFx45x50_per-total shoulder compression—the weight of the instantaneous power of the spectrum by the Hilbert-Huang transform in the range from 45 to 50 Hz [18, 20].

Analyzing the Directed acyclic graphs (DAGs) based on blood pressure measurements, with subsequent analysis of the indicators (attributes) obtained from Arterial oscillograms (D. Vakulenko, L. Vakulenko) in a group of healthy (50 persons), chronic obstructive pulmonary disease (45 persons), active tuberculosis (84 persons), pneumonia (8 persons) (Fig. 8.3).

Consider possible routes to identify Health

1. The route consists of 2 nodes S-Hil-HFx35x40_overal-20 → (0.27) HFx21x25_per-total (0.53) → Health (1), way total probability (0.14)
2. The route consists of 2 nodes S-Hil-HFx35x40_overal-20 (0.27) → HFx21x25_per-total (0.53) TP-70-100 (0.02) → Health (1), way total probability (0.002862)

Consider possible routes to identify patients with Pneumonia

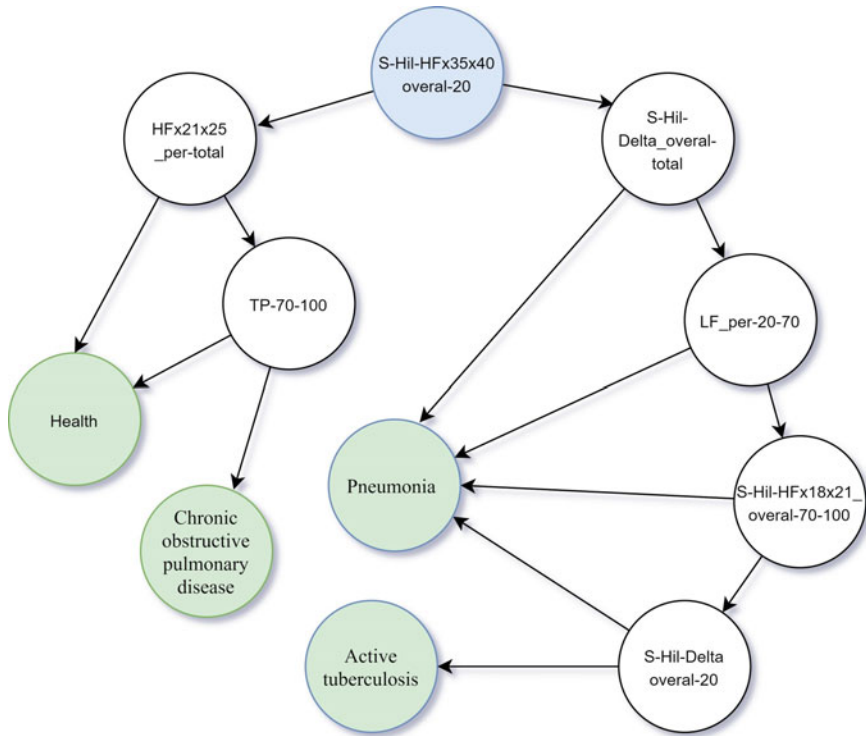


Fig. 8.3 DAG reflecting dependencies between patients of health (50 persons), with chronic obstructive pulmonary disease (45 persons), active tuberculosis (84 persons), pneumonia (8 persons)

3. The route consists of 2 nodes S-Hil-HFx35x40_overal-20 (0.04) → S-Hil-Delta_overal-total (0.09) → Pneumonia (1), way total probability (0.0036)
4. The route consists of 3 nodes S-Hil-HFx35x40_overal-20 (0.04) → S-Hil-Delta_overal-total (0.09) → LF_per-20-70 (0.06) → Pneumonia (1), way total probability (0.000216)
5. The route consists of 4 nodes S-Hil-HFx35x40_overal-20 (0.04) → S-Hil-Delta_overal-total (0.09) → LF_per-20-70 (0.06) → S-Hil-HFx18x21_overal-70-100 (0.03) → Pneumonia (0.67), way total probability (0.0000043416)
6. The route consists of 5 nodes S-Hil-HFx35x40_overal-20 (0.04) → S-Hil-Delta_overal-total (0.09) → LF_per-20-70 (0.06) → S-Hil-HFx18x21_overal-70-100 (0.03) → S-Hil-Delta_overal-20 (0.01) → Pneumonia (1) way total probability (0.0000000648)

Consider possible routes to identify patients with Chronic obstructive pulmonary disease

- 7 The route consists of 3 nodes S-Hil-HFx35x40_overal-20 (0.24) → HFx21x25_per-total (0.47) → TP-70-100 (0.97) → Chronic obstructive pulmonary disease (1), way total probability (0.11)

Consider possible routes to identify patients with Active tuberculosis of lungs

8. The route consists of 5 nodes S-Hil-HFx35x40_overal-20 (0.45) → S-Hil-Delta_overal-total (0.91) → LF_per-20-70 (0.94) → S-Hil-HFx18x21_overal-70-100 (0.97) → S-Hil-Delta_overal-20 (0.99) → Active tuberculosis (1), way total probability (0.37).

In the 8th route, patients with active tuberculosis were found to have the highest way total probability—0.37 (83 people). A key endpoint was the assessment of the instantaneous stress level prior to the onset of S-Hil-Delta shoulder compression—the instantaneous power of the Hilbert-Huang transformation range from 1 to 4 Hz [18, 19]. When Healthy was detected, among the 50 individuals in the study sample, adaptive ability was the key feature, 1-st way total probability 0.14 (49 individuals) during adaptation to shoulder compression by the cuff in the range of 21–25 Hz—total spectrum power calculated by Fourier transform.

Analyzing the Directed acyclic graphs (DAGs) based on blood pressure measurements, with subsequent analysis of the indicators (attributes) obtained from Arterial oscillograms (D. Vakulenko, L. Vakulenko) in a group of Healthy (471 persons), Cardiovascular disease (322 persons), Active tuberculosis of lungs (84 persons) (Fig. 8.4).

Consider possible routes to identify patients with Active tuberculosis of lungs.

1. The route consists of 2 nodes HFx45x50_overal-total (0.1) → HF_overal_int_n (0.15) → Active Tuberc (1), way total probability (0.01)
2. The route consists of 3 nodes HFx45x50_overal-total (0.37) → HF_overal_int_n (0.06) → Hurst-total (0.07) → CVD (1), way total probability (0.00010878)

Consider possible routes to identify patients with Cardiovascular disease (CVD)

3. The route consists of 5 nodes HFx45x50_overal-total (0.37) → HF_overal_int_n (0.06) → Hurst-total (0.07) → Total_power_int_p (0.02) → HFx6x8_overal-total (0.01) → CVD (1), way total probability (0.0000003108)
4. The route consists of 6 nodes HFx45x50_overal-total (0.37) → HF_overal_int_n (0.06) → Hurst-total (0.07) → Total_power_int_p (0.02) → HFx6x8_overal-total (0.01) → LF/HF-70-100 (0.01) → CVD (1), way total probability (0.000000003108)
5. The route consists of 5 nodes HFx45x50_overal-total (0.37) → HFx18x21_overal-20 (0.93) → Total_power_int_p (0.98) → VPR-neg (0.99) → S-Hil-HFx45x50_per-total (1) → CVD (1), way total probability (0.33)

Consider possible routes to identify Healthy

6. The route consists of 6 nodes HFx45x50_overal-total (0.54) → HF_overal_int_n (0.79) → Hurst-total (0.93) → Total_power_int_p (0.98) → HFx6x8_overal-total (0.99) → LF/HF-70-100 (0.99) → Health (1), way total probability (0.38)

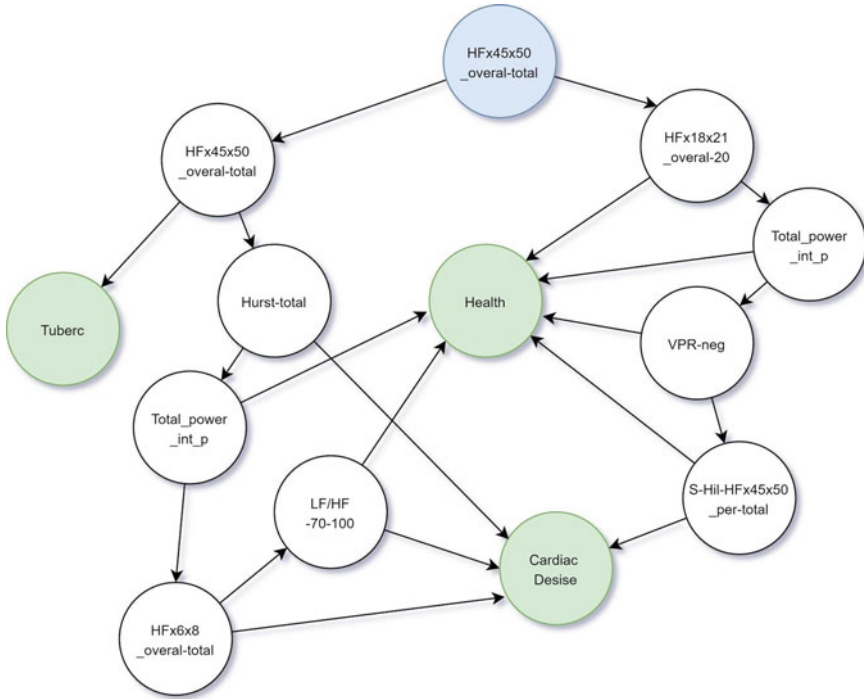


Fig. 8.4 DAG reflecting dependencies between healthy (471 persons), cardiovascular disease (322 persons), active tuberculosis of lungs (84 persons)

7. The route consists of 4 nodes HFx45x50_overall-total (0.54) → HF_overall_int_n (0.79) → Hurst-total (0.93) → Total_power_int_p (0.98) → Health (0.53), way total probability (0.2)
8. The route consists of 2 nodes HFx45x50_overall-total (0.54) → HFx18x21_overall-20 (0.07) → Health (0.8), way total probability (0.03)
9. The route consists of 3 nodes HFx45x50_overall-total (0.54) → HFx18x21_overall-20 (0.07) → Total_power_int_p (0.02) → Health (0.8), way total probability (0.0006048)
10. The route consists of 5 nodes HFx45x50_overall-total (0.54) → HFx18x21_overall-20 (0.07) → Total_power_int_p (0.02) → VPR-neg (0.01) → S-Hil-HFx45x50_per-total (0.04) → Health (1), way total probability (0.0000003024)
11. The route consists of 4 nodes HFx45x50_overall-total (0.54) → HFx18x21_overall-20 (0.07) → Total_power_int_p (0.02) → VPR-neg (0.01) → Health (1), way total probability (0.00000756).

The highest probability of detecting Healthy was found in the sixth route—0.38 (443 persons). A key endpoint was the assessment of the ability of the autonomic nervous system to maintain equilibrium of sympathetic and parasympathetic

units during adaptation at the beginning of LF/HF-70-100 shoulder compression. At LF/HF-70-100 values ≤ 10.9 (443 persons) Healthy, and at LF/HF-20-70 > 10.9 —patients with Cardiovascular diseases.

The next largest route identified patients with Cardiovascular disease, way total probability (0.33). A key endpoint was the evaluation of the instantaneous restorative power of the braking and pulse excitation during adaptation to S-Hil-HFx45x50_per-total shoulder compression of the instantaneous power spectrum of the Hilbert-Huang transform in the range from 45 to 50 Hz [18, 20].

Analyzing the Directed acyclic graphs (DAGs) based on blood pressure measurements, with subsequent analysis of the indicators (attributes) obtained from arterial oscillograms in a group of Patients Smokers and Active tuberculosis (46 persons), Non-smokers and Active tuberculosis (80 persons) (Fig. 8.5).

Consider possible routes for identifying patients Non-smokers and Active tuberculosis

1. The route consists of 2 nodes S-Hil-0-4_per-total (0.63) \rightarrow Total_025_per_int_n (0.23) \rightarrow No-Smoke (1), way total probability (0.15)
2. The route consists of 3 nodes S-Hil-0-4_per-total (0.63) \rightarrow Total_025_per_int_n (0.23) \rightarrow S-Hil-HFx40x45_oval-100-70 (0.09) \rightarrow Non-Smoker (1), way total probability (0.01)
3. The route consists of 6 nodes S-Hil-0-4_per-total (0.63) \rightarrow VLF_per-70-100 (0.74) S-Hil-HFx18x21_oval-20-70 (0.8) \rightarrow S-Hil-HFx18x21_oval-70-end (0.93) \rightarrow S-Hil-0-4_int_p (0.96) \rightarrow Total_025_per_int_n (0.98) \rightarrow Non-Smoker (1), way total probability (0.33)
4. The route consists of 5 nodes S-Hil-0-4_per-total (0.63) \rightarrow VLF_per-70-100 (0.74) \rightarrow S-Hil-HFx18x21_oval-20-70 (0.8) \rightarrow Total_04_per (0.58) \rightarrow Total_025_per_int_n (0.25) \rightarrow Non-Smoker (1) way total probability (0.05)
5. The route consists of 4 nodes S-Hil-0-4_per-total (0.63) \rightarrow VLF_per-70-100 (0.74) \rightarrow S-Hil-HFx18x21_oval-20-70 (0.8) \rightarrow Total_04_per (0.58) \rightarrow Non-Smoker (0.85), way total probability (0.19)

Consider possible routes to identify patients Smokers and with Active tuberculosis

6. The route consists of 3 nodes S-Hil-0-4_per-total (0.37) \rightarrow Total_025_per_int_n (0.77) \rightarrow S-Hil-HFx40x45_oval-100-70 (0.01) \rightarrow Smoker (1), way total probability (0.26)
7. The route consists of 2 nodes S-Hil-0-4_per-total (0.37) \rightarrow VLF_per-70-100 (0.26) \rightarrow Smoker (1) way total probability (0.09)
8. The route consists of 4 nodes S-Hil-0-4_per-total (0.37) \rightarrow VLF_per-70-100 (0.26) \rightarrow S-Hil-HFx18x21_oval-20-70 (0.2) \rightarrow S-Hil-HFx18x21_oval-70-end (0.07) \rightarrow Smoker (1), way total probability (0.0013468)
9. The route consists of 5 nodes S-Hil-0-4_per-total (0.37) \rightarrow VLF_per-70-100 (0.26) \rightarrow S-Hil-HFx18x21_oval-20-70 (0.02) \rightarrow S-Hil-HFx18x21_oval-70-end (0.07) \rightarrow S-Hil-0-4_int_p (0.04) \rightarrow Smoker (1), way total probability (0.0000053872)

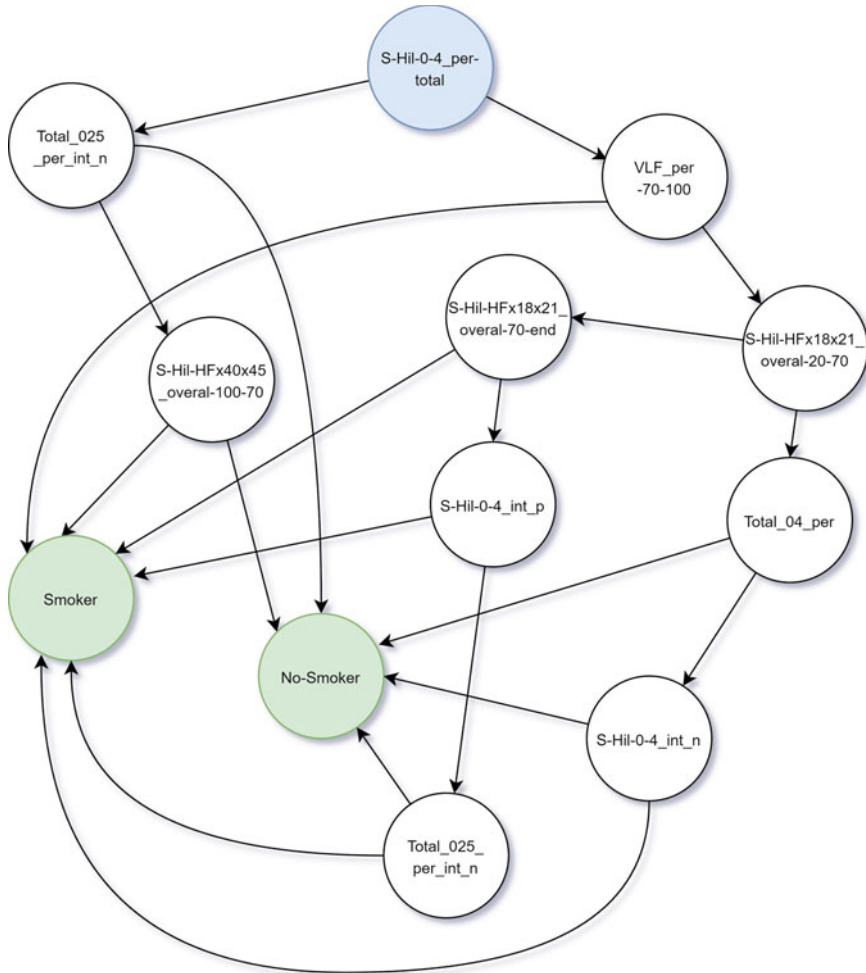


Fig. 8.5 DAG reflecting dependencies between patients smokers and active tuberculosis (46 persons), non-smokers and active tuberculosis (80 persons)

10. The route consists of 5 nodes S-Hil-0-4_per-total (0.37) → VLF_per-70-100 (0.26) → S-Hil-HFx18x21_overal-20-70 (0.2) → Total_04_per (0.42) → Total_025_per_int_n (0.75) → Smoker (1), way total probability (0.01)
11. The route consists of 5 nodes S-Hil-0-4_per-total (0.63) → VLF_per-70-100 (0.74) → S-Hil-HFx18x21_overal-20-70 (0.8) → Total_04_per (0.58) → Total_025_per_int_n (0.25) → Non-Smoker (1), way total probability (0.05).

The highest probability of detecting patients non-smokers with the active form of tuberculosis was found in the third route—0.33 (53 persons). A key endpoint was the evaluation of diastolic adaptive vascular capacity during shoulder compression

by cuff with Total_025_per_int_n. Spectral power in the range from 0 to 4 Hz by the Fourier transform of the lower extrema of the interval.

The next largest route found patients smoking with active tuberculosis, a way total probability of 0.26 (20 people). A key endpoint was the assessment of the instantaneous restorative ability of the inhibition balance and the excitation of impulses during adaptation to compression of the artery with a pinched artery during diastole S-Hil-HFx40x45_overal-100-70 instantaneous power of the spectrum by Hilbert-Huang transform in the range from 45 to 50 Hz. This feature indicates the defeat of the vascular capacity [18, 20].

Analyzing the Directed acyclic graphs (DAGs) based on blood pressure measurements, with subsequent analysis of the indicators in a group of males (79 persons) and females (12 persons) with active tuberculosis (Fig. 8.6).

Consider possible routes to identify patients females with Active lung tuberculosis

1. The route consists of 3 nodes HFx40x45_per-20-70 (0.13) → VLF_per_int_p (0.1) → Total_025_per_int_n (0.08) → Women (1), way total probability (0.00104)

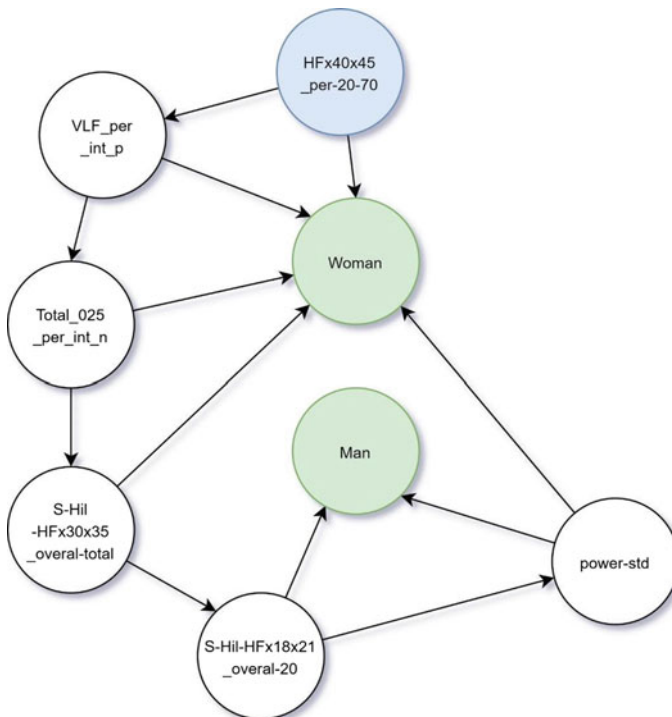


Fig. 8.6 DAG reflecting dependencies between males (79 persons) and females (12 persons) with active tuberculosis

2. The route consists of 4 nodes HFx40x45_per-20-70 (0.13) → VLF_per_int_p (0.1) → Total_025_per_int_n (0.08) → S-Hil-HFx30x35_overal-total (0.06) → Women (1) way total probability (0.0000624)
3. The route consists of 6 nodes HFx40x45_per-20-70 (0.13) → VLF_per_int_p (0.1) → Total_025_per_int_n (0.08) → S-Hil-HFx30x35_overal- total (0.06) → S-Hil-HFx18x21_overal-20 (0.05) → power-std (0.01) → Women (1) way total probability (0.000000312)
4. The route consists of 2 nodes HFx40x45_per-20-70 (0.13) → VLF_per_int_p (0.1) → Women (1), way total probability (0.01)
5. The route consists of 1 node HFx40x45_per-20-70 (0.13) → Women (0.75), way total probability (0.1)

Consider possible routes to identify patients males with Active lung tuberculosis

6. The route consists of 5 nodes HFx40x45_per-20-70 (0.87) → VLF_per_int_p (0.9) → Total_025_per_int_n (0.92) → S-Hil-HFx30x35_overal-total (0.94) → S-Hil-HFx18x21_overal-20 (0.95) → Men (0.63), way total probability (0.4)
7. The route consists of 6 nodes HFx40x45_per-20-70 (0.87) → VLF_per_int_p (0.9) → Total_025_per_int_n (0.94) → S-Hil-HFx30x35_overal-total (0.95) → S-Hil-HFx18x21_overal-20 (0.95) → power-std (0.99) → Men (1), way total probability (0.63).

The highest probability of detecting patients with active tuberculosis was found in the seventh route—0.63 (73 persons). The key endpoint was to estimate the value of the rms amplitude deviation of the arterial oscillogram power-std.

8.4 Conclusions

Using the DAGs we built differentiation routes for previously diagnosed Pulmonary (167), Cardiovascular (322) and Healthy (471) patients. With the help of indicators obtained from arterial oscillograms (D. Vakulenko, L. Vakulenko) registered during measurement of blood pressure we revealed additional sequences of indicators and evaluated their significance.

The highest probability of the constructed routes was found in patients with chronic obstructive pulmonary disease, way total probability 0.54 (45 people). A key endpoint was the evaluation of the instantaneous restorative power of the braking balance and pulse excitation during adaptation to S-Hil-HFx45x50_per-total shoulder compression of the instantaneous power spectrum of the Hilbert-Huang transform in the range from 45 to 50 Hz.

The following probability value for the detection of Healthy was found in the route—0.38 (443 persons). A key endpoint was the assessment of the ability of the autonomic nervous system to maintain equilibrium of sympathetic and parasympathetic units during adaptation at the beginning of LF/HF-70-100 shoulder compression. When LF/HF-70-100 values are less or equal than 10.9 (443 persons) then they are healthy, and at LF/HF-20-70 > 10.9—patients with cardiovascular diseases.

The third most important route in the detection of patients with active tuberculosis was the highest way total probability—0.37 (83 persons). A key endpoint was the assessment of the instantaneous stress level prior to the onset of S-Hil-Delta shoulder compression—the instantaneous power of the spectrum by the Hilbert-Huang transform in the range from 1 to 4 Hz.

Thus, the key endpoint was the evaluation of the instantaneous restoration of the braking and excitation balance during adaptation to shoulder compression—the instantaneous power of the Hilbert-Huang transform range from 30 to 50 Hz. Next in importance was the ability of the autonomic nervous system to maintain equilibrium of sympathetic and parasympathetic units during adaptation at the beginning of the LF/HF-70-100 shoulder compression power of the Fourier transform spectrum [17].

The revealed indicators indicate not only the lung tissue in the group of lung diseases, but also the systemic lesion of different levels of the nervous system involved in the process of adaptation to the compression of the shoulder vessels. This applies primarily to beginning with the instantaneous adaptive capacity of brain activity in the range from 30 to 50 Hz, the predominance of the total power of the sympathetic link of the autonomic nervous system and the increase of activity of the vascular center of the medulla oblongata. Values increased as complications of lung disease grew.

Acknowledgements The work was co-funded by the European Union's Erasmus + Programme for Education under KA2 grant (project no. 2020-1-PL01-KA203-082197 "Innovations for Big Data in a Real World").

References

1. Global Tuberculosis Report 2019—World Health Organization (2019). <https://www.who.int/tb/global-report-2019>
2. Hryshchuk, L., Okusok, O., Boiko, T., Lykhatska, H., Radetska, L.: Functional hepatic disorders in the patients with first diagnosed pulmonary tuberculosis. *Georgian Med. News* **10**(271), 43–55 (2017)
3. Caldwell, D.M., Ades, A.E., Higgins, J.P.T.: Simultaneous comparison of multiple treatments: combining direct and indirect evidence. *Br. Med. J.* **331**(7521), 897–900 (2005)
4. Ioannidis, J.P.A.: Indirect comparisons: the mesh and mess of clinical trials. *The Lancet* **368**(9546), 1470–1472 (2006)
5. Sutton, A., Ades, A.E., Cooper, N., Abrams, K.: Use of indirect and mixed treatment comparisons for technology assessment. *Pharmacoeconomics* **26**, 753–767 (2008)
6. Wells, G.A., Sultan, S.A., Chen, L., Khan, M., Coyle, D.: *Indirect Evidence: Indirect Treatment Comparisons in Meta-Analysis*. ON, Canadian Agency for Drugs and Technologies in Health, Ottawa (2009)
7. Bucher, H.C., Guyatt, G.H., Griffith, L.E., Walter, S.D.: The results of direct and indirect treatment comparisons in meta-analysis of randomized controlled trials. *J. Clin. Epidemiol.* **50**, 683–691 (1997)
8. Song, F., Altman, D.G., Glenny, A.M., Deeks, J.J.: Validity of indirect comparison for estimating efficacy of competing interventions: empirical evidence from published meta-analyses. *Br. Med. J.* **326**(7387), 472 (2003)

9. Lu, G., Ades, A.E.: Combination of direct and indirect evidence in mixed treatment comparisons. *Stat. Med.* **23**(20), 3105–3124 (2004)
10. Salanti, G., Higgins, J.P., Ades, A.E., Ioannidis, J.P.: Evaluation of networks of randomized trials. *Stat. Methods Med. Res.* **17**(3), 279–301 (2008)
11. Mintser, O., Martsenyuk, V., Vakulenko, D.: On data mining technique for differential diagnostics based on data of arterial oscillography. In: Zawiślak, S., Rysiński, J. (eds.) *Engineer of the XXI Century. Mechanisms and Machine Science*, vol 70. Springer, Cham (2020). http://doi.org/10.1007/978-3-030-13321-4_23
12. Lyapandra, A.S., Martsenyuk, V.P., Gvozdetska, I.S., Szklarczyk, R., Rajba, S.A.: Qualitative analysis of compartmental dynamic system using decision-tree induction. In: *Proceedings of the 2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS, 2*, no. 7341391, pp. 688–692 (2015). <http://doi.org/10.1109/IDAACS.2015.7341391>
13. Martsenyuk, V.P., Vakulenko, D.V., Skochylyas, S.M., Vakulenko, L.O.: Modeling and stability investigation of investment of health sector on regional level. In: Wilimowska, Z., Borzemski, L., Świątek, J. (eds.) *Information Systems Architecture and Technology: Proceedings of 40th Anniversary International Conference on Information Systems Architecture and Technology—ISAT 2019*. ISAT 2019. *Advances in Intelligent Systems and Computing*, vol. 1052, pp. 121–131. Springer, Cham (2020)
14. Selskyi, P., Vakulenko, D., Televiak, A., Veresiuk, T.: On an algorithm for decision-making for the optimization of investment of health sector on regional level using neural network clustering. *Fam. Med. Primary Care Rev.* **20**(2), 171–175 (2018)
15. Martsenyuk, V.P., Vakulenko, D.V.: On model of interaction of cell elements at bone tissue remodeling. *J. Autom. Inf. Sci.* **39**(3), 68–80 (2007). <https://doi.org/10.1615/JAutomatInfScien.v39.i3.70>
16. Martsenyuk, V.P., Vakulenko, D.V.: On model of interaction of cell elements in the process of remodeling bone tissue on the basis of nonlinear partial differential equations. *J. Autom. Inf. Sci.* **39**(7), 75–83 (2007). <https://doi.org/10.1615/JAutomatInfScien.v39.i7.60>
17. Vakulenko, D.V., Martseniuk, V., Vakulenko, L.O., Selskyi, P.R., Kutakova, O.V., Gevko, O.V., Kadobnyj, T.B.: Cardiovascular system adaptability to exercise according to morphological, temporal, spectral and correlation analysis of oscillograms. *Fam. Med. Primary Care Rev.* **21**(3), 253–263 (2019)
18. Martsenyuk, V., Vakulenko, D., Vakulenko, L., Kłos-Witkowska, A., Kutakova, O.: Information system of arterial oscillography for primary diagnostics of cardiovascular diseases. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 17th International Conference, CISIM 2018, Olomouc, Czech Republic, pp. 46–56. Springer, Berlin (2018)
19. Rösler, F.: From single-channel recordings to brain-mapping devices: the impact of electroencephalography on experimental psychology. *Hist. Psychol.* **8**(1), 95–117 (2005). <https://doi.org/10.1037/1093-4510.8.1.95>. PMID16021767
20. Orekhova, E.V., Rostovtseva, E.N., Manyukhina, V.O., Prokofiev, A.O., Obukhova, T.S., Nikolaeva, A.Y., Schneiderman, J.F., Stroganova, T.A.: Spatial suppression in visual motion perception is driven by inhibition: evidence from MEG gamma oscillations. *NeuroImage* **213**, 116753 (2020). ISSN 1053-8119. <http://doi.org/10.1016/j.neuroimage.2020.116753>. <http://www.sciencedirect.com/science/article/pii/S1053811920302408>

Chapter 9

Bipartite Graphs—Petri Nets in Biology Modeling



Anna Gogolińska and Wiesław Nowak

Abstract Petri nets (PNs) are bipartite graphs with two types of nodes: places and transitions. Places usually represent objects and transitions represent some actions or reactions. Places may contain tokens, which according to defined rules, can be transferred between places by transitions. The basic idea of PNs is simple. However, it allows to create a wide range of models. This chapter is focused on PNs models in biology. A few examples of PNs applications in this discipline are presented: modeling of Gene Regulatory Networks; metabolic, signaling and regulatory network modeling and analysis of Molecular Dynamic simulations results. Basic definitions, biologically important properties and the most commonly used extensions of PNs are also presented. The chapter shows that PNs can be used to create variety of models: qualitative, quantitative, synchronous, asynchronous, which can be applied to various biological phenomena.

Keywords Petri nets · Modeling · Gene regulatory networks · Reactions modeling · Molecular dynamics simulations · Immune system modeling

9.1 Introduction

Petri nets (PN) formalism belongs to the mathematical languages created to describe the distributed systems. The first concepts of Petri nets [1] were proposed by Carl Adam Petri in 1939. He proposed the currently common graphic representation for nets and demonstrate their application in chemical processes. His famous dissertation “Kommunikation mit Automaten” (Communication with Automata), published

A. Gogolińska (✉)

Faculty of Mathematics and Computer Science, Nicolaus Copernicus University, 87-100 Toruń, Poland

e-mail: anna.gogolinska@mat.umk.pl

W. Nowak

Faculty of Physics, Astronomy and Informatics, Nicolaus Copernicus University, 87-100 Toruń, Poland

© Springer Nature Switzerland AG 2022

S. Zawiślak and J. Rysiński (eds.), *Graph-Based Modelling in Science, Technology and Art*, Mechanisms and Machine Science 107,

https://doi.org/10.1007/978-3-030-76787-7_9

in 1962 [2], is considered the first introduction of Petri nets to science. In that work PNs were used to synchronize communicating automata. This puts the Petri nets among the oldest modeling techniques in the computer science. Since 1962 the Petri nets theory has been greatly developed, some theoretical questions have been posed and solved, and many subclasses of PN have been developed in order to improve specialist systems modeling [3]. Due to their simplicity and universality PNs have been applied in many branches of science. The main field of applications of PNs modeling is engineering. Here Petri nets-based models are used to solve different-scales problems like, for example, production scheduling [4], deadlock control of automated manufacturing [5] or even traffic jump control [6]. They are also often applied in computer science, for studying the properties of communication protocols [7], multimedia architecture [8] or in artificial intelligence [9]. Since the nineties Petri nets have been also applied in modeling of biological systems. Pioneers in this field were Reddy [10] and Hofestädt [11]. After those first works many diverse PNs applications have been published.

The **goal of this chapter** is to show how simple on their idea, bipartite graphs called Petri nets can be used as powerful and flexible biology modeling tool, which allows to create various models in many areas. Examples, presented in the chapter, show variety of applications of PNs. They also are used to demonstrate some problems which may occur using PNs and how to solve those problems, thanks to a wide range of methods available for PNs.

Chapter organisation: in the next section basic definitions related to PNs are presented. Also properties and a few extensions of basic PNs, which are important in modeling of biology, are described. Section 9.3 contains the idea of Gene Regulatory Networks and usage of Petri nets in their modeling. In Sect. 9.4 the applications of PNs in modeling of biological reactions, like metabolic, signaling and regulatory pathways are presented. Section 9.5 describes Petri nets methods of analysis of molecular dynamic simulations results. The chapter is concluded in Sect. 9.6.

9.2 Petri Nets

9.2.1 Basic Definitions

Petri nets [12] have a form of a bipartite graph with two kinds of nodes: places and transitions. Any two places or two transitions cannot be connected by the edge. Many extensions of the basic form of the PNs have been developed, some of them are presented further in this Section. However, we will start with the basic, more classical type of PNs. This type is called in various ways, in this chapter we would use one of the most common name: marked PNs. Often, in respect to the basic, marked type of PNs, its name is skipped, hence in this chapter *Petri nets or PNs* would refer to marked PNs, for extensions and different types their name is always given.

Definition 1 A (marked) Petri net graph is a 5-tuple (P, T, F, W, M_0) , where:

- P is a finite set of places.
- T is a finite set of transitions (or actions), such that $P \cap T = \emptyset$.
- $W : P \times T \cup T \times P \rightarrow \mathbb{N}$ is an arc weight function; if $W(p, t) = 0$ or $W(t, p) = 0$, $p \in P, t \in T$ then p and t are not connected. In other cases p and t are connected.
- $M_0 : P \rightarrow \mathbb{N}$ is an initial marking.

Places may be empty or may contain tokens. If at least one token is inside a place, the place is called marked. Distribution of tokens over the set of places in PN is called a marking. Like it is shown in Definition 1, the initial distribution of tokens is a part of a PN. In modeling, places usually represent some type of resources, objects; transitions correspond to some actions, reactions, processes. Tokens describe number of resources represented by a place.

Petri nets have a useful graphical representation, which often facilitate analysis of the PN models. Places, graphically, are represented as circles, transitions as squares or thick lines. If a place and a transition are connected then an arc is drawn between them. If a weight of an arc is one (the default weight) then it is not presented on the plot, higher values of weights are presented as numbers close to the arc. When it is possible, tokens are depicted as dots inside a place, when there are too many tokens in one place, then the number of tokens is given instead. An example of graphical representation on PN can be seen in Fig. 9.1.

Definition 2 Let $PT = (P, T, F, W, M_0)$ be a Petri net. For every transition $t \in T$ we define: set of *input places* of t : $\bullet t = \{p \in P \mid W(p, t) > 0\}$ and set of *output places* of t : $t \bullet = \{p \in P \mid W(t, p) > 0\}$.

PNs are dynamic structures. This dynamic is a result of relocation of tokens by transitions. This process is strictly defined and varies between different types of PNs.

Definition 3 Let $PT = (P, T, F, W, M_0)$ be a Petri net. Transition $t \in T$ is *enabled* (may be fired) in marking M if $\forall_{p \in \bullet t} W(p, t) \leq M(p)$. Set of all enabled transitions in marking M is denoted $enb(M)$. If t is enabled it can be *fired (executed)*. During firing the current marking M is changed to the new marking $M' = M - \bigcup_{p \in \bullet t} W(p, t) + \bigcup_{p \in t \bullet} W(t, p)$. We denote it by MtM' . Markings which can be obtained (directly or indirectly) from the initial marking by firing of transitions are called *reachable* markings.

Definition 4 Let $PT = (P, T, F, W, M_0)$ be a Petri net. The effect of transition t is defined as follows: $eff(t) = \bigcup_{p \in t \bullet} W(t, p) - \bigcup_{p \in \bullet t} W(p, t)$.

Example of PN before and after firing of transition t_1 is presented in Fig. 9.1. Transition t_1 in the left part of the Figure is enabled (all its input places contain at least so many tokens as the weight between the place and the transition). In the right part one can see that the marking is changed, input places of t_1 contain less tokens

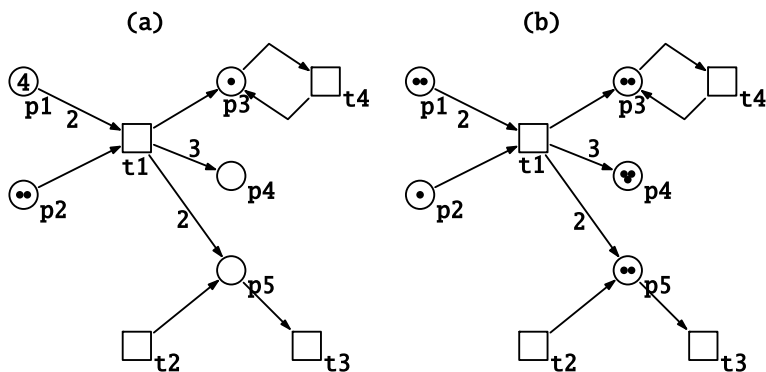


Fig. 9.1 Example of (marked) Petri net. **a** Before firing of transition t_1 , **b** after firing of transition t_1 . Examples of: transition without input places (t_2), transition without output places (t_3) and a self-loop (t_4) are presented

and output places of t_1 contain more—number of transferred tokens is determined by weights. The number of tokens in PNs is not constant, it is even possible to create transitions without input places (like t_2 in Fig. 9.1)—they produce tokens without any conditions; or without output places (like t_3 in Fig. 9.1)—they consume tokens without any conditions. A place can be at the same time an input and an output place for a transition—see transition t_4 in Fig. 9.1. Such cases will be called *self-loops*. If a place and a transition create self-loop, then the transition can fire only when the place is marked. It can be seen as some kind of a switch for the transition.

Graphs usually have some kind of matrix representation. The same is true for Petri nets. PNs can be represented as two matrices with integer coefficients: an input matrix and an output matrix. The input matrix represents weights of arcs from transitions to places and the output matrix represents weights of arcs from places to transitions.

Definition 5 Let $PT = (P, T, F, W, M_0)$ be a Petri net, $|P| = n$, $|T| = m$, then the *input matrix* is $C^+ = (\alpha_{ij})_{n \times m}$, where

$$\alpha_{ij} = \begin{cases} W(t, p); & \text{where } p \in t^\bullet \\ 0 & \text{otherwise} \end{cases}$$

The *output matrix* is a matrix $C^- = (\alpha_{ij})_{n \times m}$, where

$$\alpha_{ij} = \begin{cases} W(p, t); & \text{where } p \in {}^\bullet t \\ 0 & \text{otherwise} \end{cases}$$

The input and output matrices may be used to calculate the incidence matrix.

Definition 6 Let $PT = (P, T, F, W, M_0)$ be a Petri net, C^+ be the input matrix of PT , and C^- be the output matrix of PT . The *incidence matrix* $C = (\alpha_{ij})_{n \times m}$ is defined as $C = C^+ - C^-$.

The element α_{ij} of the incidence matrix represents the token's change at place p_i by firing of transition t_j . Please notice, that self-loops are not visible in the incidence matrix, because the marking of a place in a self-loop does not change regardless of firing the transition in the self-loop. It may cause some problems during analysis.

Definition 7 T-invariant of a Petri net $PT = (P, T, F, W, M_0)$ is a vector $x \in \mathbb{N}^m$, where $|T| = m$, satisfying the equation $C \cdot x = 0$.

Definition 8 P-invariant of a Petri net $PT = (P, T, F, W, M_0)$ is a vector $y \in \mathbb{N}^n$, where $|T| = n$, satisfying the equation $C^T \cdot y = 0$.

T-invariants and p-invariants are very important properties in analysis of biological Petri nets. Firing all transitions from one t-invariant will reproduce a given marking. In biology one t-invariant should correspond to one biological process or a pathway. All t-invariants should have a biological meaning, if some t-invariants do not have any biological sense it suggests the error in the model, hence they can be used to verify the model. In rare cases a t-invariant, which does not correspond to any biological process, may indicate a novel property. A p-invariant represents a set of places over which the weighted sum of tokens is constant. In biological models they are commonly used to model substances conservation.

In some PNs number of t-invariants is very large, for example a few thousands. Analysis of so many t-invariants are almost physically impassible, hence to reduce their number, the most similar t-invariants may be connected into t-clusters. "Similar" in this case means that the t-invariants have many common transitions. The choice of the specific definition of distance (*similarity*) between t-invariants and type of clusterization algorithm depends on the researcher.

The last important concept related to invariant analysis are Maximal Common Transition Sets (MCT-sets). The MCT-set is a set of transitions which occur always together in the considered set of t-invariants. Transitions inside one MCT-set do not have to be connected by places. MCT-sets represent a kind of building blocks of the networks. Their biological meaning should be checked and they may represent reactions which show a similar behavior.

9.2.2 Biologically Important Properties

An important advantage of Petri nets for modeling biological systems is that they are supported by theoretically well-founded techniques and tools for simulation and analysis. The following properties may be studied and give insight into the modeled process:

- *Simulations of PNs, token-games*: this technique involves starting from the initial marking, then firing the enabled transitions (according to different algorithms) and observing the obtained markings. This method is common for quantitative models to give exact levels of substances, but can be also used for qualitative models to observe qualitative changes.

- *Boundedness* ensures that the number of tokens in every place and in every reachable marking is bounded. For biological networks this means that products cannot accumulate.
- *Reachability* of a given marking M from the initial marking M_0 —for marked PNs this problem is decidable (it is not true for some extensions). In biological models this properly allows to check the existence of an evolution of the system from an initial state to a desired state.
- *Liveness* ensures that it is always possible to fire at least one transitions. In some marking may not be any enabled transitions—those marking are called *death markings*. The death markings may represent abnormal flow of a biological process or sometimes expected steady-state of the model.
- *Analysis of invariants* (relevant definitions and descriptions are presented at the end of Sect. 9.2.1, examples of practical use of this method are presented in Sect. 9.4).
- *Siphons and traps* analysis: *Siphons* is a subset S of places, where each transition that puts a token to S also removes a token from S . This implies the preservation of emptiness: once empty, a siphon never again gains tokens. *Trap* is a subset Tr of places where each transition that removes a token from Tr also puts a token to Tr . This implies the preservation of at least one token: if at least one of places in Tr is marked, there always at least one place in Tr is marked.

9.2.3 Extensions of Petri Nets

The classical, marked PNs are suitable to create quantitative models, but to represent complex biological processes often additional elements are useful, like for example time or real numbers. Moreover, the basic idea of marked PNs is quite simple and easy to modify. Hence, during years, many extensions and types of PNs were developed, which allowed to enrich modeling capabilities of PNs.

9.2.3.1 Timed Petri Nets

Time may be added to PNs in many ways. It can be associated with places or with transitions [13], but all those PNs are called Timed Petri nets. The most popular are two formalisms: Ranchamdani's Timed Petri nets (RTPN) and Merlin Time Petri nets (MTPN) [14]. In RTPNs value of a firing time is associated with each transition and a clock state is associated with PN.

Definition 9 A Ranchamdani's timed Petri net is a six-tuple: (P, T, F, W, M_0, f) , where (P, T, F, W, M_0) is a Petri net and $f: T \rightarrow \mathbb{R}^+$ is a firing time function, which assigns a positive real number, called firing time, to each transition.

Definition 10 The clock state is a pair (M, V) , where M is a marking and V is a clock valuation function, $V: \text{enb}(M) \rightarrow \mathbb{R}^+$.

The clock state is defined only for enabled transitions. The first value the clock for a newly enabled transition is given by the firing time function. If an enabled transition is executed its clock state is reseted. If an enabled transition is not executed its clock state is decreased by the clock state value of the executed transition. The transition with the smallest value of the clock state is choose to be executed. This method ensures that when a transition t is enabled, then it will fire after at most $f(t)$ times units.

Definition 11 A Merlin timed Petri net is a six-tuple: (P, T, F, W, M_0, I) , where (P, T, F, W, M_0) is a Petri net and $I: T \rightarrow [\mathbb{R}^+, \mathbb{R}^+]$ is a firing time function, which assigns an interval, called firing interval, to each transition.

An absolute time is assigned to a MTPN. Each enabled transition has its own time interval in which it can be fired. That time depends of the left bound of its firing interval, of the time elapsed since it became last enabled, and of the time in which the rest of the enabled transitions became enabled.

9.2.3.2 Stochastic Petri Nets

In Stochastic Petri nets (SPNs) enabled transitions fire with an exponentially distributed time delay. The idea of SPNs was presented almost simultaneously in [15] and [16].

The basic definition of the SPN is as follows [17]:

Definition 12 An *SPN* is a six-tuple: $SPN = (P, T, I, O, M_0, \lambda)$, where (P, T, W, M_0) is the marked PN and $\lambda = (\lambda_1, \lambda_2, \dots, \lambda_n)$ is an array of (possibly marking dependent) firing rates associated with transitions.

A firing rate is associated with each transition. It specifies the amount of time that must elapse before the transition can fire. This firing rate is a random variable with negative exponential probability distribution function. During the firing process each enabled transition has to sample its firing rate. The transition which obtains the minimum value is the one to be executed. Then the marking is changed according to standard rules.

9.2.3.3 Continuous and Hybrid Petri Nets

Continuous Petri nets (CPNs) can be easily defined from marked PNs. The main difference is a type of tokens, which in CPNs are real numbers. Also weights are real values.

Definition 13 A continuous Petri net is a 5-tuple (P, T, F, W, M_0) , where: P is a finite set of places, T is a finite set of transitions, $W: P \times T \cup T \times P \rightarrow \mathbb{R}$ is an arc weight function, and $M_0: P \rightarrow \mathbb{R}$ is an initial marking.

The definition of CPNs can be changed to allow weights which are not only numbers but functions. These features make continuous Petri nets fully capable of representing chemical reactions. Furthermore, the dynamics of a continuous Petri net can be expressed in terms of Ordinary Differential equations in agreement with the standard mass action kinetics of chemical reactions [18].

Hybrid Petri nets (HPNs) usually contain classical, discrete places and transitions as well as continuous places and transitions.

Definition 14 A hybrid PN is $Q = (P, T, h, W, M_0)$, where P and T are the sets of places and transitions respectively; $h : P \cup T \rightarrow \{D, C\}$ indicates for every place or transition whether it is a discrete or continuous one. A non-negative integer called the number of token is always associated with a discrete place and a non-negative real numbers called the mark is always associated with a continuous place. $W(p, t)$ ($W(t, p)$) is a function that define arc from a place p (a transition t) to a transition t (a place p), where the arc has a weight of non-negative integer (non-negative real value) if $h(p) = D$ ($h(p) = C$).

HPNs are usually like in Definition 14. However, other types of PNs may be also included and extend the meaning of the word *hybrid*.

9.2.3.4 Colored Petri Nets

Colored Petri nets (CPNs) is one of the most complex type of PNs. In CPNs places and tokens have types, token can be inside a place only when their types are compatible. Instead of weights, functions, even very complex one, may be assigned to arcs. Those functions operate on tokens and may transform then, for example change their types. Guards, which have a form of logical expressions, may be defined for transitions. To determine if transition may be executed, not only sufficient number of tokens is required in input places but also the guard expression must be true (if it is defined for the transition).

Values of tokens are assigned to variables in functions defined for arcs and variables in a guard. This process is called *binding*. The formal definition of CPNs is as follows.

Definition 15 ([19]) A (non-hierarchical) *colored Petri net* is a nine-tuple $CPN = (P, T, A, \Sigma, V, C, G, E, I)$, where:

- P and T are finite, disjoint sets of *places* and *transitions* (similar to the case of inhibitor nets);
- $A \subseteq P \times T \cup T \times P$ is a set of *directed arcs*;
- Σ is a finite set of non-empty *color sets*;
- V is a finite set of *typed variables* such that $Type(V) \in \Sigma$ for all variables $v \in V$;
- $C : P \rightarrow \Sigma$ is a *color set function* that assigns a color set to each place;
- $G : T \rightarrow EXP R_V$ is a *guard function* that assigns a guard to each transition t such that $Type[G(t)] = Bool$;

- $E : A \rightarrow EXP R_V$ is an *arc expression function* that assigns an arc expression to each arc a such that $Type[E(a)] = \mathbb{N}^{C(p)}$, where p is the place connected to the arc a ;
- $I : P \rightarrow EXP R_\emptyset$ is an *initialisation function* that assigns an initialisation expression to take each place p such that $Type[I(p)] = \mathbb{N}^{C(p)}$.

9.2.3.5 Fuzzy Petri Nets

Fuzzy Petri nets (FPNs) allow to join graph theory, dynamic of PNs and fuzzy reasoning mechanism.

Definition 16 ([20]) FPN can be defined by an 8-tuple set as follows: $FPN = (P, T, D, I, O, f, \alpha, \beta)$ where:

- P is a finite set of places
- T is a finite set of transitions
- $D = \{d_1, d_2, \dots, d_n\}$ is a finite set of propositions, and $P \cap T \cap D = \Phi$, $|P| = |D|$.
- $I : T \rightarrow 2^{|P|}$ is the input function which determines the input places to a transition,
- $O : T \rightarrow 2^{|P|}$ is the output function which determines the output places to a transition,
- $f : T \rightarrow [0, 1]$ is an association function, a mapping from transitions to real values between 0 and 1;
- $\alpha : P \rightarrow [0, 1]$ is an association function, a mapping from places to real values between 0 and 1;
- $\beta : P \rightarrow D$ is an association function, a bijective mapping from places to propositions.

Let A is a set of directional arcs, $A \subseteq P \times T \cup T \times P$. If $p_j \in I(t_i)$, then there exists a directional arc $a_{ji} \in A$, from the place p_j to the transition t_i . If $p_k \in O(t_i)$, then there exists a directional arc $a_{ik} \in A$, from the transition t_i to the place p_k . If $f(t_i) = \mu_i$, $\mu_i \in [0, 1]$, then the transition t_i is said to be associated with a real value μ_i which denotes the certainty factor of t_i . If $\beta(p_i) = d_i$, and $d_i \in D$, then the place p_i is said to be associated with the proposition d_i . The tokens of a place $p_i \in P$ is denoted by $\alpha(p_i)$, where $\alpha(p_i) \in [0, 1]$. If $\alpha(p_i) = y_i$, $y_i \in [0, 1]$, and $\beta(p_i) = d_i$, then it indicates that the truth value of proposition d_i is y_i .

9.3 Modeling of Gene Regulatory Networks

Genes are sequences of nucleotides in DNA or RNA that encodes proteins or other products of their synthesis. However, in this section we will focus on proteins only. Every cell in an organism contains full set of genes specified for that organism. For example each cell in human body contains more than 43 thousands of genes [22].

Naturally, not all of genes are expressed. The expression of a gene is a process in which information from the gene is read and used to synthesize a protein encoded by the gene [21]. A gene, which is expressed is called *active*, a gene which is not expressed is called *inactive*. In different cells and in different moments of time, a set of expressed genes is also different. Very important factor in gene expression is also an environment. Many proteins are designed to be used in specific situations, for example in reaction for some type of stress (heat stress [23], cold stress [23], etc.) or in reaction for presence of specific substances i.e. presence of lactose in *Escherichia coli* [24]. To save energy and substances those proteins are synthesized (genes are expressed) only when it is required.

The gene expression process is very complex. Like it was mentioned above, presence of some substances or environmental factors activates genes. However, a very important role in regulation of gene expression is played by the genes themselves. Some genes activate others, some inhibit others, self-promotion and self-inhibition is also possible. Those interactions between genes are usually not direct, but by the products of their expression.

All those interactions among genes and environment can be described as a complex network and are referred to as Gene Regulatory Networks (GRN) [25]. In order to understand and study the sophisticated behavior of GRNs various modeling methods have been used. The most important characteristics of those models are:

- *quantitative*—in quantitative models the exact numeric results are obtained. The main disadvantage of those methods is: they require also exact numeric data for construction, and that type of data is difficult to obtain in biological system.
- *qualitative*—those models do not give exact numeric outcomes, which may be consider as a flaw. They only present qualitative interactions among genes, proteins and metabolites. However, those models are much easier to construct.
- *synchronous*—in those models state of all variables (genes or other substances) are updated at once, in one step. This methodology gives a deterministic results, but is not fully consistent with biological experiments. In nature, different regulatory interactions have various rates, so they happen at different times.
- *asynchronous*—only one variable is updated in a single step, using the current values of other variables. A disadvantage of this method is: they are non-deterministic due to fact, that every variable is equally likely to be actualized in every step and this leads to very rich behaviors, not all of which are observed in the nature. However, in general, those models are considered more realistic and consisted with biological experiments—like mentioned above, regulation processes happen at different times.

One of the most popular quantitative method are Ordinary Differential Equations (ODEs). On the other hand, the most common qualitative modeling framework (probably even the most popular in general) are Boolean networks [26]. A Boolean network consists of a set of Boolean variables (the nodes of the network) each associated with a Boolean function defined on a subset of the variables. Each variable can be in active state (value 1), which means the gene is active, the metabolite is present; or in inactive state (value 0), which represents the inactivation of the gene or lack of

Table 9.1 Truth tables for a simple Boolean network presented in Fig. 9.2

g_2	g_3	g_1
0	0	1
0	1	1
1	0	0
1	1	1

g_1	g_3	g_2
0	0	0
0	1	0
1	0	0
1	1	1

g_1	g_3
0	1
1	0

Right part of tables describes the next state of entities

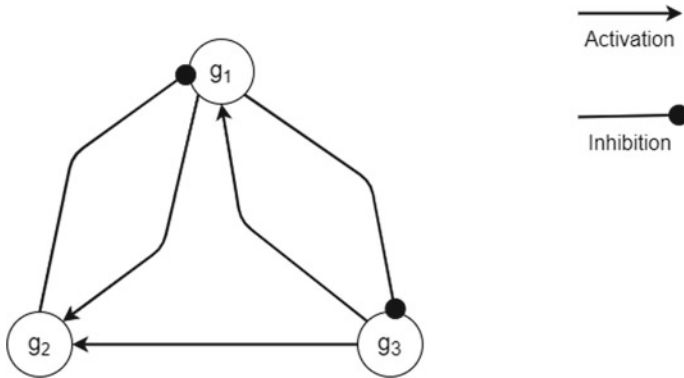


Fig. 9.2 Simple Boolean network with three genes as entities. Two types of edges are presented. Interactions among genes are described in Table 9.1

the substance. Boolean networks describing GRP are usually presented as directed graphs, where nodes represent variables and with two types of edges. The first type of the edge, graphically visualized as regular arrow, corresponds to the promotion, activation between nodes—a positive influence. The graphical representation of the second type varies between dashed arrows, arrows with dots or line as an arrowhead; and corresponds to inhibition of the node—a negative influence. Boolean networks as directed graphs are usually obtained directly from the lab experiments performed to infer a GRN, when the exact Boolean functions are often unknown. The example of a Boolean network (represented as truth tables and a graph) are presented in Table 9.1 and Fig. 9.2.

9.3.1 Petri Nets in GRN Modeling

Due to the nature of (marked) Petri nets, they are the most suitable to model GRN in a qualitative and asynchronous way. In most methods of GRN modeling, Petri nets are generated from Boolean networks or Boolean functions. The most straightforward method, one may say the most “classical” one is presented in [27]. The authors proposed a novel idea to represent one gene (or other entity) by two places in PN.

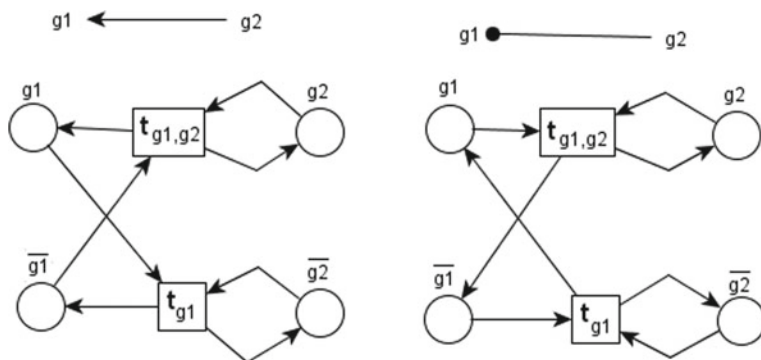


Fig. 9.3 BRPNs created from genetic regulatory Boolean networks presented above. On the left, g_1 without any impact is inactive, on the right g_1 without any impact is active (it is indicated by the arc between g_1 and t_{g_1}). The initial marking will describe state of g_2 and that state cannot change (in those examples, for different Boolean networks it may be possible)

For entity g_i the first place is denoted simply as g_i and it represents the active state of the entity. When the place is marked it corresponds to the situation when g_i is active. The second place is named usually \bar{g}_i and model the inactive state of g_i . When a token is present in \bar{g}_i it means that entity g_i is not active. Due to those assumptions, the Petri net corresponding to the GRN is constructed according to the following rules:

- every place g_i has its corresponding place \bar{g}_i
- only one of places g_i and \bar{g}_i may be marked at the given time
- sum of tokens in g_i and \bar{g}_i is always one.

This method, based on two places for one entity was adapted by many other authors.

The authors in [27] used gene regulatory Boolean network to create marked Petri net. Obtained PN was called Boolean regulatory Petri net (BRPN). In BRPN two places and a set of transitions $\{t_{g_i, X}\}$ are created for every gene g_i . The set X is an empty set or X contains every combination of genes, which have impact on g_i . Effect of transition t_{g_i} (when set X is empty) corresponds to the state of g_i when there are not any factors which affect g_i —gene is by default active or inactive. To determine effect of other transitions the author used *dynamical graph* where vertices represent states of the system (i.e. n -tuples giving the expression levels of the n genes), and edges represent transitions between states. The simple example of two BRPN are presented in Fig. 9.3. They describe the basic interactions between two genes: activation and inhibition. In [27] the authors presented two real life applications: Drosophila Cell Cycle and Flowering in Arabidopsis. They created BRPNs for both study cases and used them to determine death markings. Biological meaning of those death markings was also explained.

The described method is asynchronous and has the same disadvantages as it was mentioned in Sect. 9.3. However, Petri nets allow also to create synchronous models of GRNs. In [28] the authors presented approach for constructing qualitative Petri net

models. As the starting point, they took a synchronous Boolean network and the truth tables for every entity. Then, using logical minimization, two Boolean expressions for each entity are extracted. The first one describes when the entity becomes active and the second specifies when it becomes inactive. Those expressions capture the behavior of a Boolean network. The next step is the translation of logical equations into a corresponding Petri net (similarly, like in [27]) to implement that specific behavior. To model the synchronous semantics with the naturally asynchronous Petri nets, the authors created the two phases protocol of updating Petri net markings. In the first phase, each entity decides what its next state will be. This decision is not immediate but is stored in one of two additional places. Moreover, for every entity the third additional place is added, to mark that the decision has been made. When all entities decided, the second phase starts, when the state of each entity is updated, according to the stored decision. In [28] the authors presented the sporulation in *B. subtilis* study case. The created Petri net was used to perform a range of simulations to gain insight into the modeled behavior. They also used the model to investigate experimental hypotheses.

The most straightforward asynchronous model presented in [27] is not the only one obtained using Petri nets. In [29] the authors created more complex one, but it does not exhibit so strong non-deterministic behavior, which is the main flow of this type of models. They used techniques from speed-independent (SI) circuits, which tend to be deterministic, though they can handle certain kinds of non-determinism. The authors interpreted GRNs as SI circuit, then SI circuit was implemented by Signal Transition Graphs (STGs). STGs are Petri nets in which transitions are labeled with rising and falling edges of circuit signals. They can capture the behavior of the SI circuit and its environment. The authors shown that when such implementation is not possible, it indicates the areas of the STG which need to be refined and the additional information about the environment's behaviour or relative reaction rates are required. In [29] the Lysis-Lysogeny Switch in phage λ study case is also presented and used to show the capabilities of the model.

All so far described GRN Petri nets models are based on marked Petri nets. However, as it was presented in Sect. 9.2.3 PNs universe is noticeably richer. Many extensions of Petri nets have been also used to model and study GRNs.

Using marked Petri nets only quantitative models can be created. Continuous Petri nets dynamic is consistent with ODEs, hence this type of Petri nets is sufficient to represent quantitative features of GRNs and it is fully capable of representing chemical reactions. However, exact numeric data in biology is difficult to obtain, hence more popular are Hybrid Petri nets (HPNs) models. In [30] the authors used HPNs to effectively observe the dynamics of concentrations of proteins in λ phage. They shown some computational results of simulations, which gave insight into the modeled processes.

In [31] an hybrid Petri net model of the fission yeast cell cycle regulation mechanism was presented. Based on the simulation of the model, the authors draw some conclusions about the regulation mechanism under study. Their results are consistent with known biological facts, as well as results obtained through ODEs models.

The *lac* operon gene regulatory mechanism is studied in [32]. The authors used

Hybrid Petri nets (more precisely Hybrid Functional Petri nets, with functions as weights) to create their model. Using obtained PN the following cases were examined: the wild type (without any mutations), the *lac* repressor mutant and the *lac* operon mutant. All their observations are supported by experimental results.

Also Stochastic Petri nets have been used to model GNRs, like for example in [33]. The authors prepared the yeast cell cycle model. The SPN model catches the behavior of the wild type budding yeast cells and a variety of mutants. Their results match some characteristics of budding yeast cells that cannot be found with the deterministic methods. The model was based on deterministic ODEs and it belongs to quantitative type.

In [29] the authors presented a fuzzy Petri net (FPN) model to design the GRNs. Their results shown that the presented method is feasible and acceptable to design the genetic regulatory network and investigate the dynamical behaviors of gene network. Also Colour Petri Nets can be used to study GRNs, like it is shown in [35].

9.4 Modeling of Biological Reactions

Like it was mentioned in Sect. 9.3 the process of gene regulation is very complex. It is not the only complicated mechanism taking place in a cell. Life of the whole cell is based on sophisticated interactions between DNA, RNA, proteins, cell's organelles and chemical components. Moreover, each cell is in constant interaction with its environment. Different substances constantly leave or enter the cell; proteins and hormones bind to membrane receptors and trigger various reactions. In more complex organisms cells cooperate with each other and create specialized tissues. One may say that the whole life is based on numerous interactions on different levels of focus, from interactions between atoms, through interactions between biomolecules to interactions between cells. All those biological reactions may be seen as a very complex network of dependencies. Unfortunately, such network will be hierarchical and enormous on every level and so far we are not able to create such a complex model (because of for example lack of knowledge). However, it does not stop researchers from trying to study smaller parts of it. Here, Petri nets come to aid, as one of the more popular methods used to model various types of biological reactions.

9.4.1 *Petri Nets Models of Metabolic, Regulatory and Signaling Networks*

The complex network of biological interactions on the biomolecules level may be divided into parts, called *metabolic pathways*. One metabolic pathway is a series of chemical reactions occurring within a cell, where product of one reaction is a substrate for the next. The reactants, products, and intermediates of those reactions

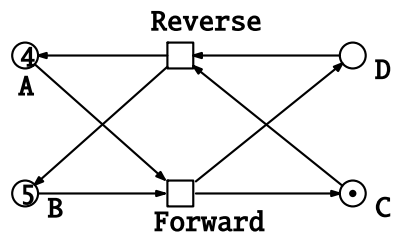
are known as metabolites. Very important role in metabolic pathways is played by enzymes—proteins which accelerate reactions and are necessary for them to occur, but are not consumed during the reactions. Metabolic pathways interact with each other via common metabolites. Due to serialized character of metabolic pathways they are often presented as metabolic networks. Among biological interactions we may also distinguish regulatory and signaling networks, which are strongly related to metabolic networks and differences between them are rather vague. All those types of networks play an enormous role in living cells—they form a bridge between genotype and phenotype.

Regulatory networks are associated with mechanism responsible for the changes in concentration or activity of a products. Details of the regulation, such as concentration, exact level of substances, are often omitted and reduced to activation or inhibition interactions. Semantics underlying those interactions defines logical formalism. Subtype of regulatory networks are GRNs (described separately in Sect. 9.3). However, in general regulatory networks are more universal and are not focused only on gene regulation but on regulatory interactions that guide all types of biological reactions. Signaling networks usually describe (i) an interaction between extracellular substances and cells through molecular receptors; and (ii) intracellular biochemical interactions and activation events of biomolecules in response to the substances.

Metabolic, regulatory and signaling networks are more complicated than regular graphs, where reactions would correspond to arcs and metabolites to vertices. Mathematically, biological networks are a sort of hypergraphs ([36]). However, they are suitable to be modeled by Petri nets. Bipartite nature of Petri nets allows to describe biological interactions with two types of vertices, where the first one (transitions) corresponds to reactions and the second one (places) to metabolites. Example of such a method is presented in Fig. 9.4.

The usual Petri net modeling process is to start with qualitative model. The model is extracted and build from biology literature and verified by qualitative analyses. Usually, basic, marked type of Petri nets is used. Many analysis methods from Petri nets universe can be applied (see Sect. 9.2.2) to study the obtained model. It gives insights in the biological reactions process. The next step, if it occurs, should concentrate on adding stochastic and/or deterministic information about quantities. Then the model can be transform from quantitative to qualitative one [37] and would be represented with different Petri nets extensions.

Fig. 9.4 Petri net representation of reaction $A + B \leftrightarrow C + D$



One of many metabolic models created using the basic, marked Petri nets is presented in [38]. The authors provided model of the hemojuvelin (HJV)–hepcidin axis involved in the maintenance of the human body iron homeostasis. Created PN has 48 places and 65 transitions, all of them representing exact metabolite (in respect to places) or biological process (in respect to transitions). The main part of the analysis was based on t-invariants analysis (see Sect. 9.2.1). The model contains 197 minimal t-invariants, no p-invariants and 14 non-trivial MCT-sets. Number of t-invariants is significant, hence the authors grouped them into t-clusters using the Mean Split Silhouette (MSS) method. To find the best number of clusters Calinski–Harabasz coefficient has been calculated for the chosen method of clustering. Finally, 21 t-clusters were obtained. Analysis of the most interesting, from the biological point of view, t-clusters, MCT-sets and t-invariants and relationships between them, gave some biological findings. The study demonstrated differences in conversion of m-HJV to s-HJV between extrahepatic tissues and the liver. Moreover, the authors have found that martiptase-2 and furin levels depend on the iron resources in the human body. Those results can be used to determine which parts of the analyzed process require an in-depth analysis, including laboratory verification.

The Petri net model presented in [38] was pure—it does not contain any self-loops. Like it was mentioned before (Sect. 9.2.1) self-loops are not reflected in the incidence matrix, hence those connections will not have any impact on invariants. Due to this fact, the invariant analysis may be incomplete. However, it is not a an unsolvable problem in PN universe. The method of unfolding self-loops is presented in [39]. In that paper, model of the human body iron homeostasis is presented. The authors obtained the Petri net with 47 places and 57 transitions (all with exact biological meaning). P-invariants, t-invariants, MCT-sets and t-clusters were analyzed. The model may help to study the human body iron homeostasis process, which is not completely understood, although playing an important role in the human organism.

Although, the t-invariant analysis is a basic method in studies of metabolic PN models, it is not always an easy process and researches should keep this in mind. In paper [40] the authors presented the PN model of the metabolism of *Arabidopsis thaliana*. *Arabidopsis thaliana* is a model system for the analysis of the basic physiological and metabolic pathways of plants. Created PN contained 134 places and 243 transitions. In this case, the authors were not able to perform typical t-invariant analysis, because the number of t-invariants was so large that it was impossible to calculate them. The authors divided the model into four biology-driven subnetworks and applied a graph-theoretic reduction to the model. Then, they obtained 27,646 t-invariants, which they classified into a few groups to allow analysis. They studied in details one, biologically important t-invariant and demonstrated some interesting biological results.

Not only t-invariant analysis can give important biological results in studies of marked Petri nets models. In [41] the PN model of signaling network was presented—more precisely the model of Epidermal Growth Factor Receptor (EGFR) signaling to the Ras-Raf-Mek-Erk (Ras-MAPK) pathway. That pathway has been strongly implicated in the development and progression of cancer and may be choose as a target for drugs. The siphons analysis allowed to find nodes within the model from

which a signal is irrecoverable once lost. Metabolites included in siphons represent candidate drug target(s) for combination therapy.

9.4.2 *PN Model of Immune System*

In [47] the authors presented PN's model of the human Immune System (IS). IS is the main defense against pathogens like bacteria, viruses, protozoa, harmful substances and others. Presence of those pathogens triggers the immune response, which is a very complex process based on interactions between various cell types, proteins and substances like cytokines. All those factors cooperate and create complicated network of reactions. PNs are a suitable tool to present those interactions. Moreover, the immune response can be classified into different types: cellular and humoral response or innate immune response and adaptive immune response.

The PN model of IS is created using marked Petri nets and is presented in Fig. 9.5. It contains 105 places and 112 transitions, all with exact biological meaning. The model presents the adaptive part of the immune response and the reaction of macrophages, and can be divided into five parts: the reaction of dendritic cells, the proliferation of helper T cells, the cellular and humoral responses and the macrophages reaction (this division is also shown in Fig. 9.5).

The t-invariant analysis was performed to verify the IS model and 46 t-invariants were found, all with biological meaning. Some of them were quite similar, hence clusterization of t-invariants was performed and resulted in 13 t-clusters. Also 19 MCT-sets were recognized and analyzed.

The model was used to study correlation between Autism Spectrum Disorder (ASD) and the immune system. ASD is a range of severe medical conditions without a known cure. Its symptoms are: communication impairments, social deficits, problems with the acceptance of changes, repetitive behaviors, etc. The etiology of ASD is not known, and a few risk factors have been identified. Mutations of genes are perhaps one of the most probable causes [48]. However, some authors suggest that possible ASD risk factors is related to the immune system [49–51]. The studies shown that in autistic people levels of some cytokines are altered [50, 51]. Another premise for the association between ASD and IS may be observation that in some autistic cases fever improves the condition of children with ASD [52]. The PN model of IS was used to study those phenomena. The impact of fever was added to the model, by changing weights of some transitions. Those changes correspond to higher levels of activation of some immune cells and stimulation of processes during the immune response, like for example faster differentiation and proliferation of Tc lymphocytes. ASD was increment to the model by increase of levels of cytokines whose levels were reported to be elevated in autistic cases.

To study qualitative changes in levels of selected cytokines number of simulations were performed (Sect. 9.2.2). Four types of simulations were carried out: simulations of the model without impact of the fever and ASD, with impact of ASD only, of fever only, and with both fever and ASD. During the simulations the levels of the following

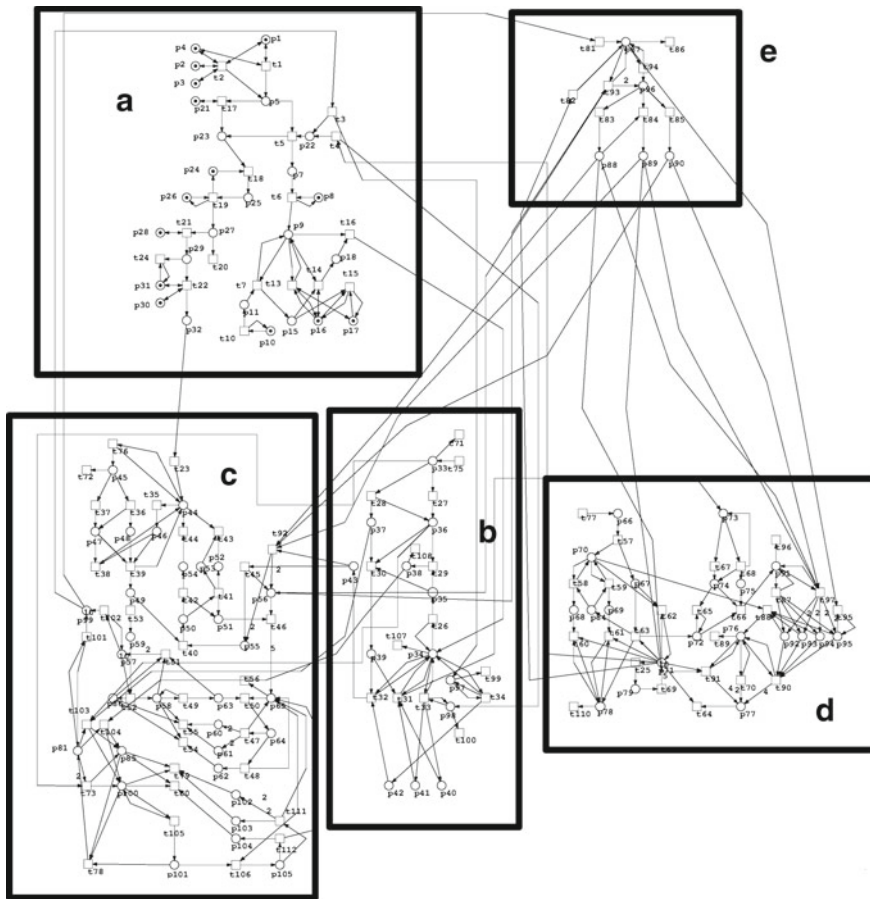


Fig. 9.5 The Petri net model of the immune system. Parts of the model corresponding to different phases of the immune response are marked by black frames: A—the reaction of dendritic cells, B—the proliferation of helper T cells, C—the cellular response, D—the humoral response, E—the macrophages reaction

cytokines were calculated: $INF - \gamma$, $IL - 1$, $IL - 6$ and $TNF - \alpha$. Results for $IL - 6$ during both the humoral and cellular responses and during only the humoral response are presented in Fig. 9.6.

For $IL - 6$ one can easily notice that with fever level of observed cytokine is lower than without the fever, especially for the humoral response only is it almost the same like for cases without ASD. That is a promising result, because observations [52] show that the fever does not always improve the state of autistic children. The hypothesis is that the improvement in mental abilities is more significant when only the humoral response is present. Verification of such a statement requires further studies and more experimental data. The general findings in the study presented in

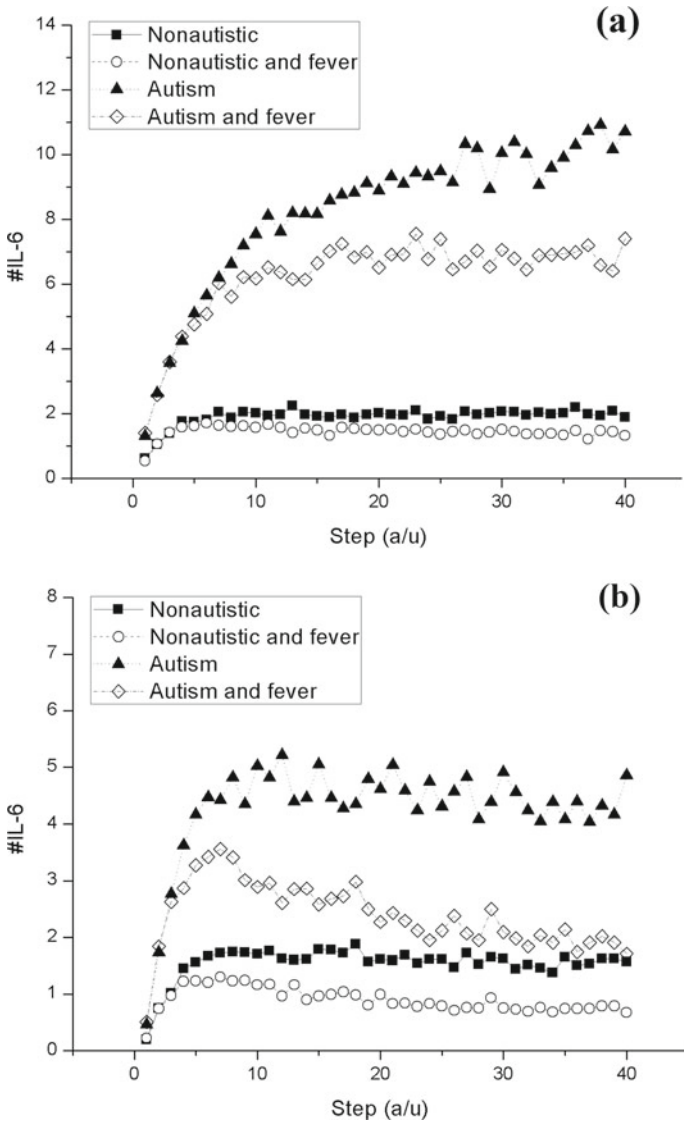


Fig. 9.6 Levels of $IL - 6$ obtained during simulations of the PN model of IS. Four cases are presented: without impact of ASD and fever, only with fever, only with ASD and with impact of both fever and ASD. Results for both types of immune response (humoral and cellular) are presented on the left **(a)**, results only for humoral response on the right **(b)**

[47] was that the fever changes the amount of cytokines and brings it closer to the level observed in healthy children or qualitative changes are the same in cases with and without ASD.

9.4.3 Extensions of PNs

Similarly, like for GRNs not only qualitative Petri nets models can be created and not only marked Petri nets can be used. Various types of PNs can be utilized, which extends models capabilities. Moreover, in this Section, it is easiest to see how flexible PNs are. Researches can, according to their needs, freely join different types of PNs. This process results in the new PNs extensions, which allow to highlight different accepts of the modeled system.

In [42] the authors used stochastic Petri nets to model the apoptosis signalling pathways. Moreover, the authors studied a method to construct a PN model of signaling pathways, for example how precisely present various reaction types in signaling pathways. In the paper type of applied PNs was called *timed*, however, according to definitions presented in this chapter, they should called *stochastic*. Stochastic Petri nets had been also used in [43], where the model of ColE1 Plasmid Replication is studied.

Different types of hybrid PNs are also widely applied in biology reactions modeling. In [44] hybrid functional Petri nets were used to create model of Xenopus Cell Cycle Pathway together with the mechanism for cell division control and checkpoint processes. The model was used to simulates dynamic cell division processes of the early Xenopus embryo. Other type of hybrid PNs was presented in [45]. The authors called them generalized hybrid PNs. They are hybrid in the sense that they combine continuous, stochastic and discrete transitions to represent deterministic, stochastic and control behaviors. In the paper model of the eukaryotic cell cycle, which is a very complex process, is presented and used to test different simulation methods. Timed hybrid Petri nets were used in [46]. This type of PNs combines features of timed, continues and discrete (marked) PNs. The authors created model of the crosstalk of AMPK, PI3K and MAPK signaling networks and simulated its dynamic to obtain levels of crucial metabolites versus time. Their results shows that during early sleep hours PI3K and MAPK pathways are active which is followed by the activation of AMPK in late sleep hours.

9.5 Biological Reactions on Atomic Scale

All biological reactions are based on interactions between atoms. Every metabolic, signaling or regulatory process is typically carried out and controlled by proteins, like enzymes or receptors. Proteins are able to perform their functions due to interactions between their atoms or their atoms and atoms of other substances. Studies

of biological reactions on the atomic level are very difficult, because we do not have convenient insights into that world. One of the methods to study atomic interactions are Molecular dynamics (MD) simulations.

MD simulations are a well-established method [53] used to generate trajectories of atomic complexes, such as proteins and/or nucleic acids, in their conformational spaces. In the MD approach numerical algorithms are used to solve sets of classical, differential equations of motion. Positions of every atom from the system are calculated in consecutive time points. The most important outcome of the MD run is an ordered sequence of frames called a trajectory. However, those output trajectories are usually a huge datasets and are hard to analyze, especially by a human inspection, because at temperature of 300 K all the atoms are in constant motion. Therefore it is very difficult to separate interesting conformational changes or functionally important motions from random atomic movements.

In [54] the authors presented PN method to study MD trajectories. The whole PN framework to analyze MD data was created by the authors, in [54] only a part of that framework, One Place One Conformation (OPOC) algorithm was described. Input for the OPOC algorithm are MD trajectories, and algorithm, as an output, generates a marked Petri net corresponding to the input trajectories. In obtained PN one place corresponds to one conformation¹ of a simulated protein and transitions represent changes between conformations. A token marks the current conformation of the system studied. The very important part of the algorithm are calculations of structural alignments, which are used to determine whether two conformations are different or the same. The authors proposed their own structural alignment algorithm, however others, well known algorithms like CA or FATCAT can be also used. Depending on the threshold given to the structural alignment algorithm more or less similar conformations can be recognized as the same and assigned to one place—clusterization of conformations is possible, hence the OPOC is a clustering algorithm. Clusters themselves can provide useful information, therefore such type of MD data analysis is not new. Clusters of structures may be generated by numerous clustering algorithms. However, the novel feature of the OPOC algorithm is a way of presentation of relationships between clusters. Thanks to transitions shown in PN graphs one can see from which conformation a protein can transform into another one and how those transformations proceed. The OPOC generated PN describes a journey of the protein through the conformational space and can be easily displayed. Analysis of that PN allow to decide what is the general character of the studied dynamical process, like for example whether in the analyzed MD trajectory a protein finds any more stable conformations or constantly probes new regions of the conformational space. Additionally, during OPOC generation of PN additional data can be collected to transform obtained marked Petri net to other types of PN, like for example Stochastic Petri nets. Hence, varied types of analysis would be possible. Moreover, the OPOC algorithm allows to join data from many MD trajectories into one PN, which is a quite unique feature.

¹ Conformation is the three-dimensional arrangement of atoms in an amino acid-chain molecule. It describes the shape of the protein.

In [54] the authors presented three studied cases. One of them is SMD simulation of forced dissociation of chemokine MCP-1 from its antibody. A few selected frames from that simulation and PN generated are by OPOC are presented in Fig. 9.7.

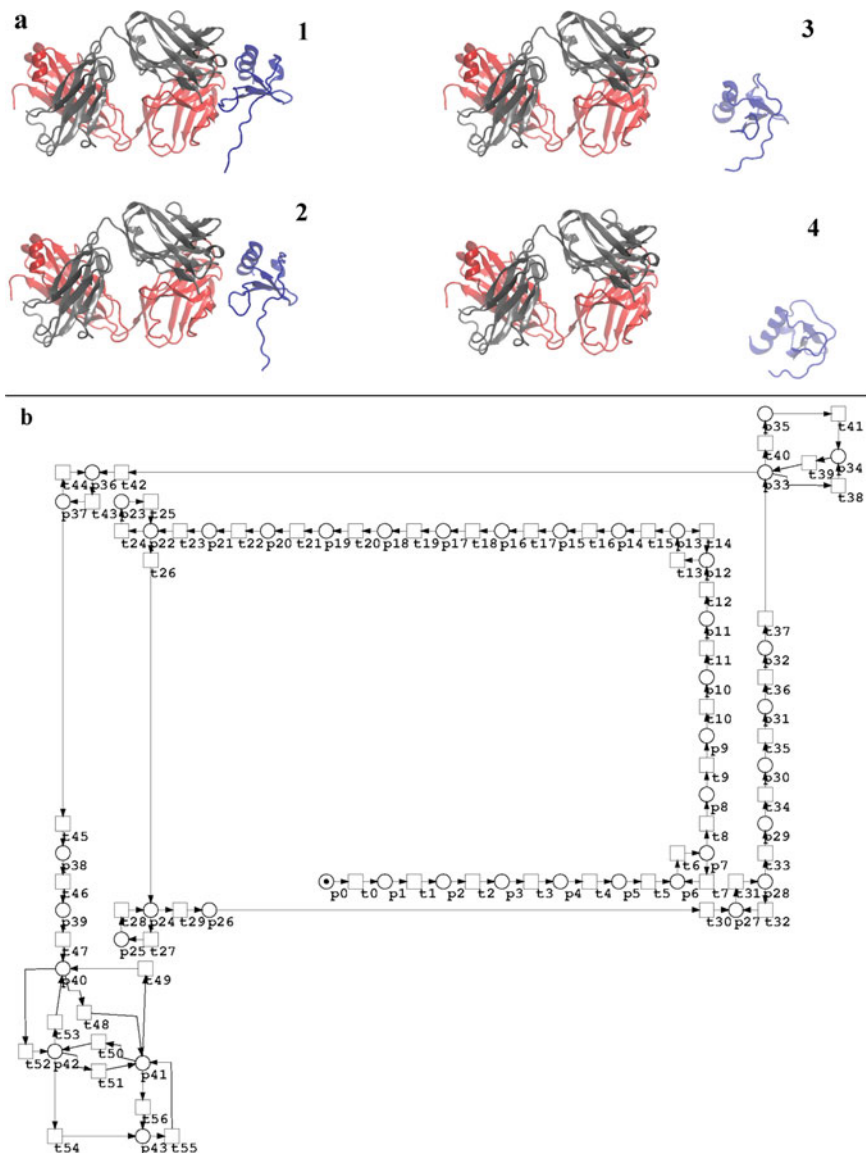


Fig. 9.7 **a** Four frames from SMD simulation of MCP-1 (the smaller molecule on the right) dissociation from the antibody (the bigger molecule on the left). 1—the starting frame, 2—the structure when bonds between the antibody and the antigen are broken, 3—conformation changes during dragging, 4—conformation obtained at the end of the simulation. **b** A Petri net generated by the OPOC algorithm for the same SMD trajectory. The high resolution figure is available [here](#)

In SMD simulations positions of some atoms are fixed, here positions of all atom in the antibody were fixed, and to some atoms additional force is added, here force was added to all C_α atoms of MCP-1. In the first part of the simulation, MCP-1 is progressively deformed because of two opposite forces i.e. the pulling force and attraction forces between the antigen and the antibody. Finally those bonds broke and MCP-1 was separated from the antibody. The MCP-1 antigen was changing its shape even after the dissociation, probably to adopt a new conformation—more suitable for the unbound state, for example the long loop at the end of the MCP-1 molecule came closer to the main body of the protein. It can be seen in PN as a sequences of consecutive places and transitions, with only a few repeated conformations (like for example p6, p7, p33, p34, p35). In the final part of the simulation the chemokine is more stable. This can be observed in PN as last part of the network (places 40–43, transitions 48–55), where MPC-1 changes freely its conformations back and forth. One can easily see that events in the simulation and generated PN correspond to each other.

9.6 Conclusions

The goal of this chapter was to show that Petri nets are useful and flexible modeling language suitable for creating biological models. PNs have been successfully applied to GRNs modeling. They allow to create all, the most important types of GRNs models. Marked Petri nets has been utilized in qualitative modeling and extended types of PNs in quantitative modeling of GRNs. Asynchronous nature of PNs allows to create asynchronous models of GRNs, but also synchronous models can be obtained by using additional places and transitions. PNs are also frequently used in modeling of different types of other biological reactions, like metabolic, signaling and regulatory pathways. These processes are the base of cells life, and thus entire organisms. Hence, their better understating is crucial in biological studies. PNs models allow to better grasp modeled processes, can be used to test hypothesis, identify elements which should be investigated in details, and even recognize new relations or phenomena. PNs can be applied not only to reactions on metabolites level but also to studies at more fundamental level—the level of atomic interactions. PNs can model journey of proteins through their conformational space, recognize important fragments of MD simulations and present the general character of the simulation.

In the chapter applications of many PNs types are also presented. They allow to extend PN models capabilities and add new properties of the modeled system to the Petri net. Elements from different PNs types can be also join into new PNs extensions, which gives a lot of freedom to researchers.

This chapter certainly does not exhaust the topic of biological applications of PNs. It contains only a few examples of PNs used in biological modeling. Many more can be find, like for example modeling of catalysis reactions [55] or usage of PNs in ecology [56]; and new studies are published frequently. However, it was not

the goal of this chapter. Presented examples were selected to show usability of PNs models in biology, but also highlight some problems which may occur during the the modeling process and solutions for those problems.

References

1. Murata, T.: Petri nets: properties, analysis and applications. *Proc. of the IEEE* **77**(4), 541–580 (1989)
2. Petri, C.A.: *Kommunikation mit automaten* (1962)
3. Diaz, M. (ed.): *Petri Nets: Fundamental Models, Verification and Applications*. Wiley, London (2013)
4. Tuncel, G., Mirac Bayhan, G.: Applications of Petri nets in production scheduling: a review. *Int. J. Ad. Manuf. Technol.* **34**(7–8), 762–773 (2007)
5. López-Grao, J.-P., Colom, J.-M., Tricas, F.: The deadlock problem in the control of flexible manufacturing systems: an overview of the Petri net approach. In: *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*. IEEE (2014)
6. Di Febbraro, A., Giglio, D., Sacco, N.: Urban traffic control structure based on hybrid Petri nets. *IEEE Trans. Intell. Transport. Syst.* **5**(4), 224–237 (2004)
7. Pura, M.L., Buchs, D.: Model checking ARAN ad hoc secure routing protocol with algebraic Petri nets. In: *2014 10th International Conference on Communications (COMM)*. IEEE (2014)
8. Bo, C., Chen, J., Deng, M.: Petri net based formal analysis for multimedia conferencing services orchestration. *Expert Syst. Appl.* **39**(1), 696–705 (2012)
9. Zouaghi, L., et al.: Mission-based online generation of probabilistic monitoring models for mobile robot navigation using Petri nets. *Robot. Autonom. Syst.* **62**(1), 61–67 (2014)
10. Reddy, V.N., Mavrovouniotis, M.L., Liebman, M.N.: Petri net representations in metabolic pathways. In: *ISMB*, vol. 93 (1993)
11. Hofestädt, R.: A Petri net application to model metabolic processes. *Syst. Anal. Model. Simul.* **16**(2), 113–122 (1994)
12. Reisig, W.: *Petri Nets: An Introduction*, vol. 4. Springer Science and Business Media, Berlin (2012)
13. David, R., Alla, H.: *Discrete, Continuous, and Hybrid Petri Nets*, vol. 1. Springer, Berlin (2005)
14. Silva, J.R., del Foyo, P.M.G.: Timed petri nets. *Petri Nets: Manufacturing and Computer Science*. InTech, 359–378 (2012)
15. Florin, G., Natkin, S.: Evaluation based upon stochastic Petri nets of the maximum throughput of a full duplex protocol. In: *Application and Theory of Petri Nets*, pp. 280–288. Springer, Berlin, Heidelberg (1982)
16. Molloy, M.K.: On the integration of delay and throughput measures in distributed processing models, 3339 (1982)
17. Marsan, M.A. Stochastic Petri nets: an elementary introduction. In: *European Workshop on Applications and Theory in Petri Nets*. Springer, Berlin (1988)
18. Gilbert, D., Heiner, M.: From Petri nets to differential equations—an integrative approach for biochemical network analysis. In: *International Conference on Application and Theory of Petri Nets*. Springer, Berlin (2006)
19. Jensen, K., Kristensen, L.M.: *Coloured Petri Nets: Modelling and Validation of Concurrent Systems*. Springer Science and Business Media, Berlin (2009)
20. Sun, J., Qin, S.-Y., Song, Y.-H.: Fault diagnosis of electric power systems based on fuzzy Petri nets. *IEEE Transactions on Power Systems* **19**(4), 2053–2059 (2004)
21. O'Connor, C.M., Adams, J.U., Fairman, J.: Essentials of cell biology. *NPG Educ.* **1**, 54 (2010)
22. Pertea, M., et al.: Thousands of large-scale RNA sequencing experiments yield a comprehensive new human gene list and reveal extensive transcriptional noise. *BioRxiv*: 332825 (2018)

23. Sørensen, J.G., et al.: Full genome gene expression analysis of the heat stress response in *Drosophila melanogaster*. *Cell Stress Chaperones* **10**(4), 312 (2005)
24. Vilar, J.M., Guet, C.C., Leibler, S.: Modeling network dynamics: the lac operon, a case study. *J. Cell Biol.* 161
25. Peter, I.S., Davidson, E.H.: *Genomic Control Process: Development and Evolution*. Academic, New York (2015)
26. Somogyi, R., Sniegoski, C.A.: Modeling the complexity of genetic networks: understanding multigenic and pleiotropic regulation. *Complexity* **1**(6), 45–63 (1996)
27. Chaouiya, C., et al.: Qualitative modelling of genetic networks: from logical regulatory graphs to standard petri nets. In: *International Conference on Application and Theory of Petri Nets*. Springer, Berlin (2004)
28. Steggles, L.J., et al.: Qualitatively modelling and analysing genetic regulatory networks: a Petri net approach. *Bioinformatics* **23**(3), 336–343 (2007)
29. Banks, R., Khomenko, V., Steggles, L.J.: A case for using signal transition graphs for analysing and refining genetic networks. *Electron. Notes Theo. Comput. Sci.* **227**, 3–19 (2009)
30. Doi, A., et al.: Protein dynamics observations of lambda phage by hybrid Petri net. *Genome Inform.* **10**, 217–218 (1999)
31. Vasireddy, R., Biswas, S.: Modeling gene regulatory network in fission yeast cell cycle using hybrid Petri nets. In: *International Conference on Neural Information Processing*. Springer, Berlin (2004)
32. Doi, A., et al.: Constructing biological pathway models with hybrid functional Petri net. In *Silico Biol.* **4**(3), 271–291 (2004)
33. Mura, I., Csikász-Nagy, A.: Stochastic Petri net extension of a yeast cell cycle model. *J. Theor. Biol.* **254**(4), 850–860 (2008)
34. Hamed, R.I., Ahson, S.I., Parveen, R.: Designing genetic regulatory networks using fuzzy Petri nets approach. *Int. J. Autom. Comput.* **7**(3), 403–412 (2010)
35. Chaouiya, C., Remy, E., Thieffry, D.: Qualitative Petri net modelling of genetic networks. *Trans. Comput. Syst. Biol.* **VI**, 95–112 (2006)
36. Klamt, S., Haus, U.-U., Theis, F.: Hypergraphs and cellular networks. *PLoS Comput. Biol.* **5**(5) (2009)
37. Koch, I., Reisig, W., Schreiber, F. (eds.) *Modeling in Systems Biology: The Petri Net Approach*, vol. 16. Springer Science and Business Media (2010)
38. Formanowicz, D., et al.: Hemojuvelin–hepcidin axis modeled and analyzed using Petri nets. *J. Biomed. Inform.* **46**(6), 1030–1043 (2013)
39. Sackmann, A., et al.: An analysis of the Petri net based model of the human body iron homeostasis process. *Comput. Biol. Chem.* **31**(1), 1–10 (2007)
40. Koch, I., Nöthen, J., Schleiff, E.: Modeling the metabolism of *Arabidopsis thaliana*: application of network decomposition and network reduction in the context of Petri nets. *Front. Genetics* **8**, 85 (2017)
41. Behinaein, B., Rudie, K., Sangrar, W.: Structural analysis of Petri nets for modeling and analyzing signaling pathways. In: *2014 IEEE 27th Canadian Conference on Electrical and Computer Engineering (CCECE)*. IEEE (2014)
42. Chen, L., et al.: Modeling and simulation of signal transductions in an apoptosis pathway by using timed Petri nets. *J. Biosci.* **32**(1), 113–127 (2007)
43. Goss, P.J.E., Peccoud, J.: Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. *Proc. Nat. Acad. Sci.* **95**(12), 6750–6755 (1998)
44. Matsui, M., et al.: Simulated cell division processes of the xenopus cell cycle pathway by genomic object net. *J. Integr. Bioinform.* **1**(1), 27–37 (2004)
45. Herajy, M., Schwarick, M., Heiner, M.: Hybrid Petri nets for modelling the eukaryotic cell cycle. *Trans. Petri Nets Other Models Concurrency* **VIII**, 123–141 (2013)
46. Bibi, Z., et al.: Modeling and analysis of the signaling crosstalk of PI3K, AMPK and MAPK with Timed Hybrid Petri Nets approach. In: *2017 17th International Conference on Computational Science and Its Applications (ICCSA)*. IEEE (2017)

47. Gogolinska, A., Nowak, W.: Petri nets approach to modeling of immune system and autism. In: International Conference on Artificial Immune Systems. Springer, Berlin (2012)
48. Freitag, C.M.: The genetics of autistic disorders and its clinical relevance: a review of the literature. *Mol. Psychiatry* **12**(1), 2–22 (2007)
49. Meltzer, A., Van de Water, J.: The role of the immune system in autism spectrum disorder. *Neuropsychopharmacology* **42**(1), 284–298 (2017)
50. Goines, P.E., Ashwood, P.: Cytokine dysregulation in autism spectrum disorders (ASD): possible role of the environment. *Neurotoxicology and teratology* **36**, 67–81 (2013)
51. Ashwood, P., Wills, S., Van de Water, J.: The immune response in autism: a new frontier for autism research. *J. Leukocyte Biol.* **80**(1), 1–15 (2006)
52. Curran, L.K., et al.: Behaviors associated with fever in children with autism spectrum disorders. *Pediatrics* **120**(6), e1386-e-1392 (2007)
53. Berendsen, H.J.C., et al.: Molecular dynamics with coupling to an external bath. *J. Chem. Phys.* **81**(8), 3684–3690 (1984)
54. Gogolinska, A., Jakubowski, R., Nowak, W.: Petri nets formalism facilitates analysis of complex biomolecular structural data. *RAIRO-Oper. Res.* **50**(2), 401–411 (2016)
55. Barylska, K., et al.: Reversing computations modelled by coloured Petri nets. *ATAED@ Petri Nets/ACSD* (2018)
56. Seppelt, R., Temme, M.-M.I.: Hybrid low level Petri nets in environmental modeling-development platform and case studies. In: *Integrative Systems Approaches to Natural and Social Dynamics*, pp. 181–201. Springer, Berlin (2001)

Chapter 10

Labeled Graphs in Life Sciences—Two Important Applications



Piotr Formanowicz, Marta Kasprzak, and Piotr Wawrzyniak

Abstract Life sciences and mathematics are usually considered as quite distant areas of research. But in fact there are close relationships between them, especially in recent years, when computational biology and bioinformatics rapidly evolve. The spectacular developments in the area of biological sciences, particularly those related to sequencing genomes, made evident that an application of formal mathematical and computer science methods is necessary for further discovering the nature of the living world. Among many areas of mathematics being useful in this context, graph theory plays especially important role. It is also worth to remember that, despite the fact that graphs are intensively applied in biology during last three decades, they were used in chemistry (being a basement of molecular biology) more than a century ago. In this chapter a short review of selected applications of labeled graphs in biology and chemistry is given. Some graph theory problems concerning molecules of chemical compounds and DNA sequencing are presented.

Keywords Labeled graphs · Molecular graphs · DNA sequencing · Structural formulas

P. Formanowicz (✉) · M. Kasprzak · P. Wawrzyniak
Poznan University of Technology, Institute of Computing Science,
Piotrowo 2, 60-965 Poznań, Poland
e-mail: Piotr.Formanowicz@cs.put.poznan.pl

M. Kasprzak
e-mail: Marta.Kasprzak@cs.put.poznan.pl

P. Wawrzyniak
e-mail: Piotr.Wawrzyniak@cs.put.poznan.pl

P. Formanowicz
Polish Academy of Sciences, Institute of Bioorganic Chemistry,
Noskowskiego 12/14, 61-704 Poznan, Poland

© Springer Nature Switzerland AG 2022
S. Zawiślak and J. Rysiński (eds.), *Graph-Based Modelling in Science, Technology and Art*, Mechanisms and Machine Science 107,
https://doi.org/10.1007/978-3-030-76787-7_10

10.1 Introduction

Computational biology is an interdisciplinary branch of science which evolves at the intersection of computing science, mathematics and biological sciences, especially molecular biology. Its main goal is to develop mathematical models of biological phenomena and algorithms solving various problems arising when biological processes and objects are analyzed using strict methods. Many of the studied phenomena have discrete nature, hence methods based on various branches of discrete mathematics are often very useful to analyze them. Here, especially important is graph theory, since many of the analyzed biological processes and objects have structure which can be described in a natural way using graphs.

But an application of graph theory in life sciences can be dated a century before an emergence of computational biology. Indeed, in the second half of nineteenth century, graphs were used to describe and analyze molecules of chemical compounds. It was a significant impulse to develop important ideas not only in chemistry but also in graph theory.

In this paper a short review of selected applications of labeled graphs to two groups of life sciences problems is presented. The first group concerns determining structural formulas of chemical compounds. The problem has a quite long history, but it has become more important recently when mass spectrometers are becoming more and more precise and more available in biological laboratories.

The second group of considered problems concerns DNA sequencing by hybridization. It is the problem which brought an attention of computer scientists into molecular biology in the late 1980s and in fact caused a rapid development of computational biology. Many interesting theoretical results have been obtained since that time. Some of them are mentioned in this chapter.

10.2 Molecular Graphs—Labeled Graphs in Chemistry

Problems considered with graphs have a long history, dating back to 1736 when Leonhard Euler considered the issue of the Königsberg bridges. The word “graph” appeared later when in 1878, in an article published in *Nature*, James Joseph Sylvester introduced the term “chemigraph” along with its abbreviated version of “graph.” It can be said that the application of graph theory to solving chemistry problems gave birth to this science under the name we know. This close relationship between graphs and chemistry will be presented here.

10.2.1 Correspondence Between Graph Theory and Chemistry

The possibility of using graphs for the mathematical description of chemical molecules seems obvious when we compare the visual representation of a graph (Fig. 10.1) and the structural formula (Fig. 10.2) of a chemical compound.

The main differences between these Figs. 10.1 and 10.2 are symbols of elements in a chemical compound. The need to assign different symbols to the vertex brings us to the essential requirement of using a graph to represent a chemical compound. We have to assign a label to each vertex in the graph. After such an operation, each vertex identifies the atom of a specific chemical element in the molecule. Modeling a chemical compound by a graph requires more such relationships between the graph and the chemical compound. There exists more such one-to-one correspondences, like edge–bond, degree of vertex–valence of the atom, and others. They are listed in Table 10.1.

These analogies in constructing a graph and a chemical compound do not exhaust the analogy between graph theory and chemistry. Many problems from the graph theory correspond to the problems of chemistry (see Table 10.2).

Fig. 10.1 Graph

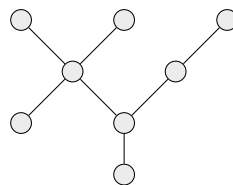


Fig. 10.2 Molecule

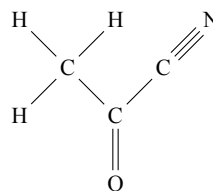


Table 10.1 Graph and chemical terms

Graph theory term	Chemical term
Vertex	Atom
Vertex label	Chemical element symbol
Vertex degree	Atom valency
Edge	Chemical bond
Parallel edges/weighted edges	Multiple chemical bonds

Table 10.2 Graph and chemical problems

Graph theory	Chemistry
Graph isomorphism	Structural isomers
Topological indices (quantitative structure properties)	Chemical or physical properties of substances (e.g. boiling point)
Enumeration problems	Counting all possible molecules for given molecular formula
Planarity	Chirality
Subgraph	A molecule fragment or chemical group

10.2.2 Definitions

There are two basic ways to define a molecular graph:

- (i) a labeled multigraph [20]
- (ii) a labeled graph with labeled edges [32].

Each graph can be denoted as $G = (V, E)$, where V is a set of vertices matching the atoms of the compound. Each vertex $u \in V$ has assigned a label $l(u)$ matching the chemical element of the atom. Depending on the way of representation, E is (i) a multiset or (ii) a set of edges. Edge $e \in E$ is an unordered pair $\{u, v\} : u, v \in V \wedge u \neq v$, the inequality of vertex ensures that the graph is a loop free. In the case where the graph is not a multigraph, i.e. (ii), each edge has assigned a label $w(e)$ which corresponds to the type of a chemical bond.

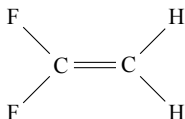
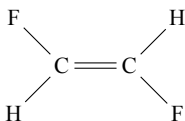
10.2.3 Graph Isomorphism and Chemical Isomerism

In chemistry, molecules with the same molecular formulae can be completely different compounds. Because apart from the elemental composition, the bonds between atoms are an essential feature of a chemical compound. Such molecules, made from the same number of atoms of each element but connected differently, are called structural isomers. To investigate whether two molecules are the same, we need to answer the question: *Is the structure of connections between atoms the same?* The answer can be found by checking whether the corresponding molecular graphs are isomorphic or not.

Isomorphism is one-to-one correspondence (a bijection) between vertex sets of two compared graphs, $G = (V, E)$ and $G' = (V', E')$:

$$f : V \rightarrow V' \text{ such that: } \forall_{u,v \in V} \{u, v\} \in E \iff \{f(u), f(v)\} \in E'$$

This definition of isomorphism for the basic graph does not distinguish all differences between molecular graphs. The graphs can have different labels in the

**Fig. 10.3** 1,1-Difluoroethylene**Fig. 10.4** 1,2-Difluoroethylene

same place in the structure. Figure 10.3 (1,1-Difluoroethylene) and Fig. 10.4 (1,2-Difluoroethylene) present two different chemical compounds with the same molecular graph structure.

It is easy to notice that these molecular graphs have the same structure but differs in vertex labels. To distinguish between these cases, an extra condition to check label equality must be added to the isomorphism bijection function [25]:

$$f : V \rightarrow V' \text{ such that:}$$

$$(\forall_{u,v \in V} \{u, v\} \in E \iff \{f(u), f(v)\} \in E') \wedge (\forall_{u \in V} l(u) = l(f(u)))$$

Graph isomorphism is a widely studied problem, and many ways to solve it have been proposed. However, no polynomial-time algorithm for this problems has been found. Hence it is not known whether the problem belongs to class P. On the other hand, it was also not possible to confirm its belonging to the class of NP-complete problems [44]. Fortunately, molecular graphs correspond to the natural chemical compounds. They have many limitations, a finite number of labels (corresponding to the symbols of elements), or a limited degree of vertices (corresponding to the valence of an atom in the compound) [1]. Using this knowledge, we can use an isomorphism algorithm for bounded degree graphs, and this problem can be solved in polynomial time [29]. This approach and most other considered graph problems concern simple graphs (without parallel edges and labeled vertices), called them just a “graph”. However, a fast method having polynomial complexity of transforming molecular graphs into simple graphs has been presented [20]. It consists of two steps:

1. Removing parallel edges: each parallel edge is split into two parts by adding an extra vertex between the adjacency source vertices. Such dummy vertices always have a degree equal to 2.
2. Removing labels: each label of a vertex is replaced by attaching a different number of dummy vertices to the labeled vertex. Of course, the number of vertices added this way is unique for each unique label. In this case, the dummy vertices always have a degree equal to 1. For example, the number of added dummy vertices can

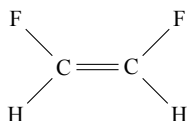


Fig. 10.5 cis-1,2-Difluoroethylene

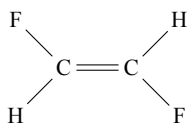


Fig. 10.6 trans-1,2-Difluoroethylene

be set to $Z = Z(a) + Z_0$, where $Z(a)$ is the atomic number of a , and Z_0 is the maximum valence in the compound.

After such transformation, the molecular graph becomes a simple one and still has a limited degree. In such a case, the isomorphism can be checked in polynomial time.

Chemical compounds can have not only the structural isomers, but there are also stereoisomers. This situation occurs for the mentioned earlier 1,2-Difluoroethylene, the substance that has the cis- (Fig. 10.5) and trans- (Fig. 10.6) isomers.

The molecular graph cannot differentiate among such spatial isomers [21]. However, this information (chiral centers, cis/trans isomerism) can be encoded by bonds labels (as in wedge and dash notation) [2]. For part of such problems, e.g., for the cis and trans isomers, a solution has been proposed through the concept of virtual bonds forming virtual cis rings [37].

10.2.4 Counting of Molecular Graphs / Isomer Enumeration

As mentioned earlier, the word graph was introduced in 1878 to denote what is now known as a molecular graph. Even before that date, in 1874, Arthur Cayley first applied the graph theory in chemistry to enumerate alkenes isomers [6]. Moreover, more recently, the DENDRAL [15] enumerate molecules program is called the first expert system [40].

The chemical problem of isomers enumeration is to count the number of different compounds with the same molecular formula. This problem can be solved using graph theory and counting all topologically distinct nodes labelings in the graph [14]. Pólya's enumeration formula gives the most general method of such a calculation.

The Pólya algorithm is based on the symmetry recognition of the molecule under study. The method is most often explained on the example of benzene and the substitution of its hydrogens with any other monovalent element, e.g., by fluorine. The

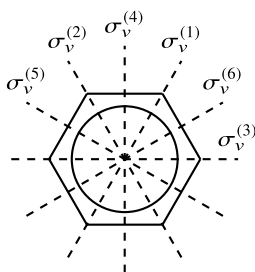


Fig. 10.7 Symmetry axes

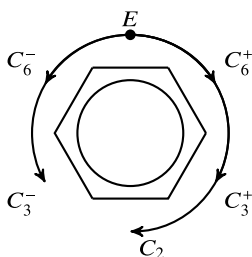


Fig. 10.8 Symmetry rotations

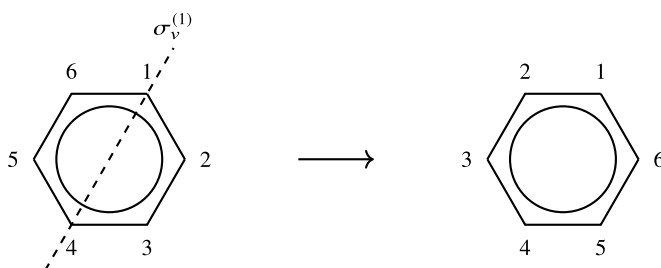


Fig. 10.9 Example of $\sigma_v^{(1)}$ symmetry, permutation group $(1)(4)(2\ 6)(3\ 5)$

six axes of benzene symmetry are shown in Fig. 10.7, while Fig. 10.8 presents six rotations in it, each with a step of 60° . The last 360-degree rotation is identity [35].

Each of these symmetries can be written as a permutation. After each binding site in benzene has been indexed, the applied symmetric operation can be described as a sequence of indexes changes that make up the symmetry's resulting indexing. When vertex with index 1 is not changed, we can denote it as a permutation $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$. When the vertex 2 is replaced with 6 and 6 with 2, it can be marked as a permutation $\begin{pmatrix} 2 & 6 \\ 6 & 2 \end{pmatrix}$. Permutations written in two-line notation can be changed to cyclic notation (1) and $(2\ 6)$. An example permutation group is presented in Fig. 10.9.

Table 10.3 All symmetries of benzene

Symmetry	Permutation	Cycle index
E	(1)(2)(3)(4)(5)(6)	z_1^6
C_6^+	(1 2 3 4 5 6)	z_6^1
C_6^-	(6 5 4 3 2 1)	z_6^1
C_3^+	(1 3 5)(2 4 6)	z_3^2
C_3^-	(5 3 1)(642)	z_3^2
C_2	(1 4)(2 5)(3 6)	z_2^3
$\sigma_v^{(1)}$	(1)(4)(2 6)(3 5)	$z_1^2 z_2^2$
$\sigma_v^{(2)}$	(3)(6)(1 5)(2 4)	$z_1^2 z_2^2$
$\sigma_v^{(3)}$	(2)(5)(1 3)(4 6)	$z_1^2 z_2^2$
$\sigma_v^{(4)}$	(1 6)(2 5)(3 4)	z_2^3
$\sigma_v^{(5)}$	(1 4)(2 3)(5 6)	z_2^3
$\sigma_v^{(6)}$	(1 2)(3 6)(4 5)	z_2^3

From such notations, we can extract the cycle index needed by Pólya's algorithm. For each permutation α , shown in Table 10.3, the cycle index z_k^i describes the number i of the permutation cycles of a given length k .

The Pólya's formula $Z(A)$ is a multivariate polynomial, counting the cycle index for a group of permutations A [38],

$$Z(A) = \frac{1}{|A|} \sum_{\alpha \in A} \prod_{k=1}^n z_k^{c_k(\alpha)}$$





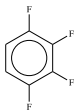
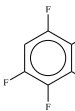
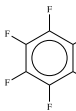

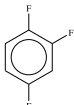

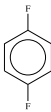

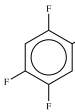
where:

- (i) $|A|$ is the number of elements in the permutation group A ,
- (ii) α is a permutation from group A ,
- (iii) k describe the permutation length,
- (iv) z_k is a variable representing cycles in permutations of the length k ,
- (v) $c_k(\alpha)$ is the number of cycles of length k in permutation α .

In the product $\prod_{k=1}^n z_k^{c_k(\alpha)}$, for each permutation α the sum of products of cycles length k and cycles number of such length $c_k(\alpha)$, of course, always equals the number of elements in a permutation α , e.g., for z_1^6 $1 \cdot 6 = 6$, for z_2^3 $2 \cdot 3 = 6$ or for $z_1^2 z_2^2$ $1 \cdot 2 + 2 \cdot 2 = 6$.

Following the Pólya theorem, to obtain number of isomers, every occurrence of variable z_k has to be replaced by the generating function $c_k(x)$. Having only two different chemical elements to choose from (hydrogen and fluorine), the simple generating function $c_k(x) = 1 + x^k$ can be used [18].

Table 10.4 All k-fluoro-benzene isomers matched to Pólya's polynomial terms

$1 \times x^0$	$1 \times x^1$	$3 \times x^2$	$3 \times x^3$	$3 \times x^4$	$1 \times x^5$	$1 \times x^6$
						
						
						

Replace in Pólya's formula $Z(A)$ for benzene example:

$$Z(A) = \frac{1}{12}(z_1^6 + 4z_2^3 + 2z_3^2 + 2z_6^1 + 3z_1^2 z_2^2)$$

each variable z_k by generating function $c_k(x) = 1 + x^k$, results in the following formula:

$$Z(A) = \frac{1}{12}((1+x)^6 + 4(1+x^2)^3 + 2(1+x^3)^2 + 2(1+x^6)^1 + 3(1+x)^2(1+x^2)^2)$$

$$Z(A) = 1 + x + 3x^2 + 3x^3 + 3x^4 + x^5 + x^6$$

The coefficient of the x^k term in the polynomial describe the number of isomers with k fluorine atoms (Table 10.4).

10.2.5 Molecular Graphs Generation / Isomer Construction

The graphs shown in Table 10.4 are only an illustration presenting the correctness of the Pólya algorithm operation. The construction of such graphs is, however, a task that has many practical applications. One of them is identifying chemical molecules based on experimental data, most often from mass spectrometry. In the simplest case, the mass spectrometry can provide very accurate information about a molecule's mass. This information can then be converted into a simple molecular formula [13] like

$C_6H_4F_2$. Creating all the structural isomers of such a molecule can be defined as the following graph problem.

The molecular formula is changed to a multiset of labels L (corresponding to the symbols of elements), and $d(l)$ function assigning to each label $l \in L$ a numeric value (corresponding to the valence of the element). The task is to construct a set S of connected graphs with labels from set L and corresponding degrees $d(l)$ that $\forall_{g,h \in G}$ g and h are not isomorphic. There are two approaches to solve this problem:

- (i) We create an unlabeled graph basing on the list of given vertices degrees $d(l)$. It is a well-known problem of graph realization [23]. Then, to obtain molecular graphs, we assign labels to all unique graphs created in this way. The assignment of these labels also has to keep the graphs topologically distinct. The algorithm for labeling these graphs can, like in the Polya algorithm, use symmetry groups and algebraic structures based on them [14].
- (ii) We start from the empty graph on labeled vertices from a multiset of labels L . Then, we assign another kind of label to each vertex in the form of an index from 1 to $|L|$. In particular the vertex set for the Difluoroethylene ($C_2F_2H_2$) can be presented as $\{C_1, C_2, F_3, F_4, H_5, H_6\}$. Next, starting from this empty graph, we extend it by one edge at each step, following the orderly generation algorithm by Read and Faradzev [17, 39]. By introducing the order of added edges, this algorithm eliminates the need to check the created graphs' isomorphism with each other.

Each of these approaches allows the application of additional conditions, e.g., the existence of a specific subgraph, which corresponds to the existence of specific chemical groups in the molecule. These algorithms are applied in various tools [3, 19, 22, 34, 43] used by chemists to discover the chemical structure of unknown compounds, or to prepare libraries of new compounds for biochemical screening.

10.3 Sequencing Graphs—Labeled Graphs in Biology

De Bruijn graphs are labeled graphs known for their technical application in modeling communication networks or parallel computer architectures. Their special form well fits such real-world schemas and guarantees a short path between any pair of nodes in a network. What is more, a polynomial-time solution to the problems of the directed Hamiltonian cycle or path, being generally NP-hard, is possible for such graphs. Here, we describe another, biological application of de Bruijn graphs and related classes of digraphs.

10.3.1 Definitions

The following statements refer to directed graphs. Graph $G = (V, A)$ has V as the set of its vertices and A as its set of arcs. Arc $a \in A$ is an ordered pair (u, v) : $u, v \in V$, where u and v can be the same vertex. Let $e(v)$ be the label of vertex v , $\text{suf}_i(s)$ the suffix of length i of string s , and $\text{pre}_i(s)$ the prefix of length i of s .

A path in a directed graph is a sequence of vertices (v_1, v_2, \dots, v_p) , where $(v_i, v_{i+1}) \in A, i = 1, \dots, p - 1$. A cycle is a path, for which the condition $(v_p, v_1) \in A$ is satisfied. A path (cycle) containing every vertex of a graph exactly once is a Hamiltonian path (Hamiltonian cycle). A path (cycle) traversing every arc of a graph exactly once is an Eulerian path (Eulerian cycle).

For an alphabet of size α and labels of constant length k ($k > 1, \alpha > 0$), *de Bruijn graph* $B(\alpha, k) = (V, A)$ has α^k vertices, every one labeled by a different word over the alphabet. For all $u, v \in V, (u \neq v \Rightarrow e(u) \neq e(v))$ and $((u, v) \in A \Leftrightarrow \text{suf}_{k-1}(e(u)) = \text{pre}_{k-1}(e(v)))$ [16]. *Adjoint* $G = (V, A)$ of a graph $H = (U, V)$ has vertices corresponding to arcs of H , and $(u, v) \in A$ if and only if the head of arc u in H is the tail of arc v [5]. Adjoint G is always a 1-graph (i.e., it has no multiple arcs), H need not be. If H is a 1-graph, its adjoint G is a *directed line graph* [8].

1-graph $G = (V, A)$ is an adjoint if and only if, for all $u, v \in V$, the following property is satisfied:

$$N^+(u) \cap N^+(v) \neq \emptyset \Rightarrow N^+(u) = N^+(v),$$

where $N^+(u)$ is the set of immediate successors of vertex u [5]. 1-graph $G = (V, A)$ is a directed line graph if and only if the following property is satisfied for all $u, v \in V$:

$$N^+(u) \cap N^+(v) \neq \emptyset \Rightarrow N^+(u) = N^+(v) \wedge N^-(u) \cap N^-(v) = \emptyset,$$

where $N^-(u)$ is the set of immediate predecessors of vertex u [8]. Digraph $G = (V, A)$ is a *quasi-adjoint graph* if and only if, for all $u, v \in V$, the following property holds [12]:

$$N^+(u) \cap N^+(v) \neq \emptyset \Rightarrow N^+(u) = N^+(v) \vee N^+(u) \subset N^+(v) \vee N^+(v) \subset N^+(u).$$

Quasi-adjoint graphs, unlike adjoints, can be multigraphs.

According to definitions in [8], a directed 1-graph $G = (V, A)$ belongs to class \mathcal{L}_k^α (can be (α, k) -labeled) if it is possible to assign labels to vertices such that, for all $u, v \in V, (u \neq v \Rightarrow e(u) \neq e(v))$ and $((u, v) \in A \Leftrightarrow \text{suf}_{k-1}(e(u)) = \text{pre}_{k-1}(e(v)))$, where $k > 1$ is the length of labels and $\alpha > 0$ is the alphabet size. *Labeled graphs* (*uniquely labeled graphs*) are these graphs that belong to a class \mathcal{L}_k^α for some α and k . Graphs satisfying the above requirements except the condition that labels must be different are called *non-uniquely labeled graphs*. A *self-adjoint* is defined as a graph isomorphic to its adjoint [24]. *Alphabet overlap digraphs* are a generalization of de Bruijn graphs. Given three integers, $\alpha \geq 1, k \geq 2$ and $1 \leq i < k$, alphabet overlap digraph $O(\alpha, k; i) = (V, A)$ is defined as a graph labeled with all possible words of

length k over an alphabet of size α , where i is a fixed offset in overlaps of vertex labels [28] (see also [27]). For all $u, v \in V$, where $|V| = \alpha^k$, ($u \neq v \Rightarrow e(u) \neq e(v)$) and $((u, v) \in A \Leftrightarrow \text{suf}_{k-i}(e(u)) = \text{pre}_{k-i}(e(v)))$.

10.3.2 Combinatorial Modeling of DNA Sequencing

We refer to one of fundamental issues associated with molecular biology, DNA sequencing. It is a process of recognizing a sequence of nucleotides of a DNA fragment. Such a sequence determines some aspects of functioning of an organism, and knowing it is a first step toward understanding biological mechanisms. Over the years, several techniques were developed to carry through this process, from small-scale laboratory methods [31, 41] to high-throughput automated modern sequencing (e.g., the Illumina sequencing), here we focus on an approach involving an algorithmic stage, the DNA sequencing by hybridization [4, 33, 42].

Not going into biological details (for those see the above references or the algorithmically oriented review [11]), the DNA sequencing by hybridization provides a set S of short words over the alphabet $\{‘A’, ‘C’, ‘G’, ‘T’\}$, where the letters stand for four nucleotides encoding genetic information of an organism: adenine, cytosine, guanine, and thymine. The words are identified via a biological hybridization experiment as parts of a DNA chain, and the goal of the computational problem is to reconstruct the chain from these words. In the case of the classical approach to the hybridization experiment, the words have the same length k and we call them k -mers (where k usually takes values from 8 to 12), they are also assumed to be different within a set. During the algorithmic stage of the process, the words from S are ordered to obtain a final nucleotide sequence of the examined fragment of a DNA chain (usually of the length a few hundreds of nucleotides). If the hybridization experiment was performed without any error (the theoretical case considered here), S is complete and the properly ordered words overlap exactly on $k - 1$ letters in pairs of neighbors, thus they form a sequence of $|S| + k - 1$ letters.

The computational problem of DNA sequencing by hybridization without any error in S was initially solved without special combinatorial models, via exhaustive search. Soon two nice graph models were proposed. The first one, by Lysov and co-workers, places words from S in vertices of a directed graph. Two vertices u and v are connected by arc (u, v) if and only if $\text{suf}_{k-1}(e(u)) = \text{pre}_{k-1}(e(v))$. In such a graph, a Hamiltonian path is looked for, which corresponds to a solution of the problem, i.e., the properly ordered sequence of all words from S [30]. In the second model by Pevzner, words from S correspond to arcs and their prefixes and suffixes of length $k - 1$ to vertices. Arcs are directed from the prefix of a word to the suffix of the same word, and the solution is an Eulerian path [36]. Figure 10.10 shows the two models in an example. It also shows a common problem in bioinformatics, ambiguity of a solution, which cannot be solved without additional information, e.g., coming from other experiments or expert knowledge.

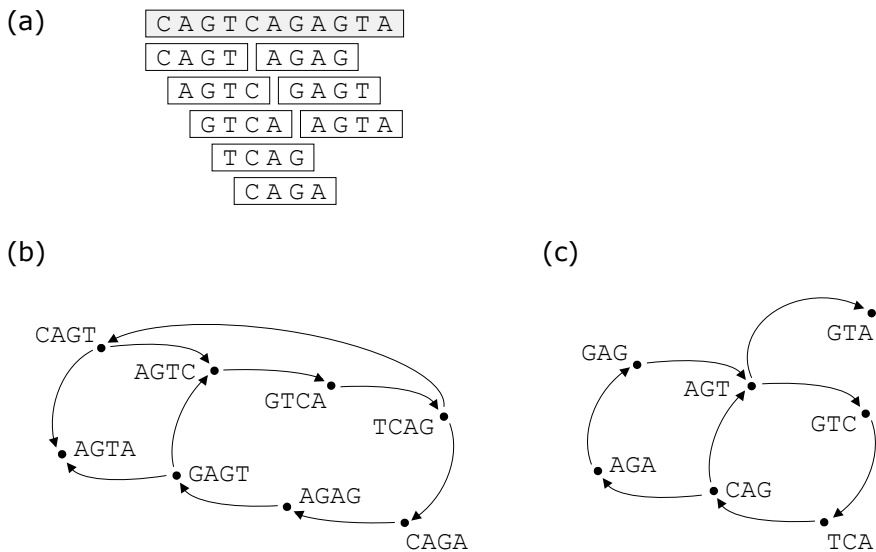


Fig. 10.10 Graph models for the problem of DNA sequencing by hybridization without errors. **a** A nucleotide sequence of a DNA fragment and k -mers identified as its parts, here $k = 4$ and $S = \{‘AGAG’, ‘AGTA’, ‘AGTC’, ‘CAGA’, ‘CAGT’, ‘GAGT’, ‘GTCA’, ‘TCAG’\}$. **b** The graph by Lysov *et al.* constructed for S , where there are two Hamiltonian paths corresponding to two possible solutions of the problem: ‘CAGTCAGAGTA’ and ‘CAGAGTCAGTA’. **c** The graph by Pevzner constructed for S with two Eulerian paths resulting in the same two nucleotide sequences

10.3.3 Directed Line Graphs, De Bruijn Graphs, and Others

The equivalence of the two graph models became a subject of research a decade later. Why, in this case, is the transformation from the NP-hard problem of the Hamiltonian path to the polynomially solvable Eulerian path possible? The answer was given in [8], the Lysov graph is a directed line graph of the Pevzner graph constructed for the same set S , and the problems of the Hamiltonian path or cycle in directed line graphs are polynomially solvable. In [8] also a wider analysis of labeled graphs was done. The graphs of Lysov and Pevzner have labels at vertices, and the overlapping labels in Lysov graphs imply the presence of arcs (it is not the case of Pevzner graphs). Actually, only the Lysov graphs can be classified as the labeled graphs, they belong to classes \mathcal{L}_k^4 . Lysov graphs are also called *DNA graphs*, especially when not restricted to errorless S . DNA graphs are vertex-induced subgraphs of de Bruijn graphs with $\alpha = 4$. Pevzner graphs are subgraphs of DNA graphs, thus in consequence, subgraphs of de Bruijn graphs.

Currently, a widest superclass of labeled graphs that is ‘easy’ for the Hamiltonian cycle/path problem is the class of quasi-adjoint graphs [12]. In general, such graphs cannot be labeled because sets of immediate successors of two vertices are no longer the same or disjoint for all the pairs within a graph. Figure 10.11 shows how graph classes mentioned here relate to each other.

10.3.4 Other Variants of DNA Sequencing

The DNA sequencing by hybridization is usually considered with errors accompanying the experiment. Then, it is also modeled as a graph problem but without the useful property of the polynomial-time solvability. Presence of any errors in an instance of the DNA sequencing problem makes the problem (i.e., the reconstruction of the original nucleotide sequence of a DNA fragment) strongly NP-hard [9]. Both methods of graph construction, by Lysov *et al.* and Pevzner, still work for S with errors, but the Hamiltonian or Eulerian path cannot be expected there. The class of DNA graphs, as presented in Fig. 10.11, covers graphs constructed according to Lysov *et al.* for all possible sets S (with or without experimental errors) that do not contain repetitions nor words of different lengths.

With next steps going beyond the original Lysov’s method, we lose ties with graphs from Fig. 10.11. If we allow overlaps of k -mers with an offset greater than 1 but constant, we still obtain a (non-uniquely) labeled graph, a vertex-induced subgraph of an alphabet overlap digraph. But for a variable offset allowed, the resulting graph is outside the class of quasi-adjoint graphs. Look at the following example, a pair of vertices ‘TGATAT’ and ‘CCATAT’, and their sets of successors { ‘GATATA’, ‘ATATTA’ } and { ‘ATATTA’, ‘CATATT’ }, respectively. The sets are not disjoint, nor equal, nor one contained in the other, thus do not match the property for quasi-adjoint graphs.

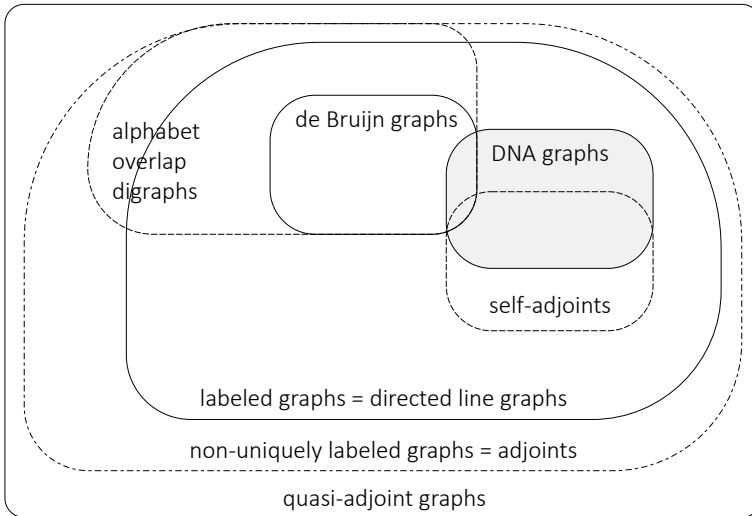


Fig. 10.11 Relations between the classes of uniquely and non-uniquely labeled digraphs, their subclasses, and quasi-adjoint graphs [26]. DNA graphs (Lysov graphs) are used as models of DNA sequencing. All the graphs included here are polynomial-time solvable instances of the problems of searching for the Hamiltonian cycle or the Hamiltonian path

A non-classical approach to the DNA sequencing by hybridization, the isothermic DNA sequencing, produces a set of nucleotide subsequences that can be characterized by the same ‘temperature’ (melting temperature of DNA duplexes) but differing in length [7]. A graph proposed as a model for this variant, for the errorless case, can be either a directed line graph, an adjoint not being a directed line graph, or a quasi-adjoint graph not being an adjoint [10]. Therefore, this problem is also polynomially solvable.

The DNA sequencing by hybridization was replaced by a newer technique, a high-throughput automated sequencing, which is realized without an algorithmic stage. However, algorithms and models are still necessary for such sequencing data at the next level of organizing them. Although the output data from these two sequencing approaches have different scale and contain different errors, basically the processes of composing partial sequences into a final DNA fragment are very similar. Consequently, graph models of Lysov *et al.* and Pevzner are still in use, after some necessary adjustments to new circumstances (for a detailed description see, for example, [11]). One of the adjustments is the permission for non-exact overlaps of sequences in the Lysov’s model. However, this one change makes a graph not satisfying the property for quasi-adjoint graphs.

10.4 Conclusions

In this chapter a short review of selected applications of labeled graphs in life sciences has been given. Graph theory is a very important and useful tool in solving various problems appearing in many areas of biological sciences. Its application helped to make a progress in DNA sequencing and mass spectrometry, among others. On the other hand, problems arising in biology (especially molecular biology) are inspirations for new directions of theoretical research in graph theory (DNA graphs and molecular graphs being examples). So, the intersection of these two seemingly not very closely related areas, i.e., biology and graph theory, is a source of many interesting problems, results and inspirations for both of them.

References

1. Akutsu, T., Nagamochi, H.: Comparison and enumeration of chemical graphs. *Comput. Struct. Biotechnol. J.* **5**(6), e201302,004 (2013)
2. Andersen, J.L., Flamm, C., Merkle, D., Stadler, P.F.: An intermediate level of abstraction for computational systems chemistry. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **375**(2109), 20160,354 (2017)
3. Badertscher, M., Korytko, A., Schulz, K.P., Madison, M., Munk, M.E., Portmann, P., Junghans, M., Fontana, P., Pretsch, E.: Assemble 2.0: a structure generator. *Chemometrics Intell. Lab. Syst.* **51**(1), 73–79 (2000)
4. Bains, W., Smith, G.: *J. Theor. Biol.* **135**, 303–307 (1988)

5. Berge, C.: *Graphs and Hypergraphs*. North-Holland Publishing Company, London (1973)
6. Biggs, N., Lloyd, E.K., Wilson, R.J.: *Graph Theory*, pp. 1736–1936. Oxford University Press, Oxford (1986)
7. Blazewicz, J., Formanowicz, P., Kasprzak, M., Markiewicz, W.: Sequencing by hybridization with isothermic oligonucleotide libraries. *Discrete Appl. Math.* **145**, 40–51 (2004)
8. Blazewicz, J., Hertz, A., Kobler, D., de Werra, D.: On some properties of DNA graphs. *Discrete Appl. Math.* **98**(1–2), 1–19 (1999)
9. Blazewicz, J., Kasprzak, M.: Complexity of DNA sequencing by hybridization. *Theor. Comput. Sci.* **290**, 1459–1473 (2003)
10. Blazewicz, J., Kasprzak, M.: Computational complexity of isothermic DNA sequencing by hybridization. *Discrete Appl. Math.* **154**, 718–729 (2006)
11. Blazewicz, J., Kasprzak, M., Kierzyńska, M., Frohberg, W., Swiercz, A., Wojciechowski, P., Zurkowski, P.: Graph algorithms for DNA sequencing—origins, current models and the future. *Eur. J. Oper. Res.* **264**, 799–812 (2018)
12. Blazewicz, J., Kasprzak, M., Leroy-Beaulieu, B., de Werra, D.: Finding Hamiltonian circuits in quasi-adjoint graphs. *Discrete Appl. Math.* **156**, 2573–2580 (2008)
13. Böcker, S., Lipták, Z., Martin, M., Pervukhin, A., Sudek, H.: DECOMP—from interpreting mass spectrometry peaks to solving the money changing problem. *Bioinformatics* **24**(4), 591–593 (2008)
14. Brown, H., Hjelmeland, L., Masinter, L.: Constructive graph labeling using double cosets. *Discrete Math.* **7**(1–2), 1–30 (1974)
15. Brown, H., Masinter, L.: *An algorithm for the construction of the graphs of organic molecules*. Stanford University (1973)
16. de Bruijn, N.: A combinatorial problem. *Proc. Koninklijke Nederlandse Akademie van Wetenschappen* **49**, 758–764 (1946)
17. Faradzev, I.: Constructive enumeration of combinatorial objects. *problèmes combinatoires et théorie des graphes* **260**, 131–135 (1978)
18. Faulon, J., Visco, D.P., Roe, D.: Enumerating molecules. *Rev. Comput. Chem.* **21**, 209 (2005)
19. Faulon, J.L.: On using graph-equivalent classes for the structure elucidation of large molecules. *J. Chem. Inform. Comput. Sci.* **32**(4), 338–348 (1992)
20. Faulon, J.L.: Isomorphism, automorphism partitioning, and canonical labeling can be solved in polynomial-time for molecular graphs. *J. Chem. Inform. Comput. Sci.* **38**(3), 432–444 (1998)
21. García-Domenech, R., Gálvez, J., de Julián-Ortiz, J.V., Pogliani, L.: Some new trends in chemical graph theory. *Chem. Rev.* **108**(3), 1127–1169 (2008)
22. Gugisch, R., Kerber, A., Kohnert, A., Laue, R., Meringer, M., Rücker, C., Wassermann, A.: Molgen 5.0, a molecular structure generator. In: *Advances in Mathematical Chemistry and Applications*, pp. 113–138. Elsevier, Amsterdam (2015)
23. Hakimi, S.L.: On realizability of a set of integers as degrees of the vertices of a linear graph. *J. Soc. Ind. Appl. Math.* **10**(3), 496–506 (1962)
24. Hao, J.: The adjoints of DNA graphs. *J. Math. Chem.* **37**, 333–346 (2005)
25. Hsieh, S.M., Hsu, C.C., Hsu, L.F.: Efficient method to perform isomorphism testing of labeled graphs. In: *International Conference on Computational Science and Its Applications*, pp. 422–431. Springer, Berlin (2006)
26. Kasprzak, M.: Classification of de Bruijn-based labeled digraphs. *Discrete Appl. Math.* **234**, 86–92 (2018)
27. Kozak, A., Glowacki, T., Formanowicz, P.: On a generalized model of labeled graphs. *Discrete Appl. Math.* **161**(13–14), 1818–1827 (2013)
28. Li, X., Zhang, H.: Embedding on alphabet overlap digraphs. *J. Math. Chem.* **47**, 62–71 (2010)
29. Luks, E.M.: Isomorphism of graphs of bounded valence can be tested in polynomial time. *J. Comput. Syst. Sci.* **25**(1), 42–65 (1982)
30. Lysov, Y., Florentiev, V., Khorlin, A., Khrapko, K., Shik, V., Mirzabekov, A.: Determination of the nucleotide sequence of DNA using hybridization with oligonucleotides. A new method. *Doklady Akademii Nauk SSSR* **303**, 1508–1511 (1988)

31. Maxam, A., Gilbert, W.: A new method for sequencing DNA. *Proc. Nat. Acad. Sci. USA* **74**, 560–564 (1977)
32. Minkin, V.I.: Glossary of terms used in theoretical organic chemistry. *Pure Appl. Chem.* **71**(10), 1919–1981 (1999)
33. Pease, A.C., Solas, D., Sullivan, E.J., Cronin, M.T., Holmes, C.P., Fodor, S.: Light-generated oligonucleotide arrays for rapid DNA sequence analysis. *Proc. Nat. Acad. Sci.* **91**(11), 5022–5026 (1994)
34. Peironcely, J.E., Rojas-Chertó, M., Fichera, D., Reijmers, T., Coulier, L., Faulon, J.L., Hanke-meier, T.: Omg: open molecule generator. *J. cheminformatics* **4**(1), 21 (2012)
35. Pevac, S., Crundwell, G.: Polyá's isomer enumeration method: a unique exercise in group theory and combinatorial analysis for undergraduates. *J. Chem. Educ.* **77**(10), 1358 (2000)
36. Pevzner, P.: l-tuple DNA sequencing: computer analysis. *J. Biomol. Struct. Dyn.* **7**, 63–73 (1989)
37. Pogliani, L.: From molecular connectivity indices to semiempirical connectivity terms: recent trends in graph theoretical descriptors. *Chem. Rev.* **100**(10), 3827–3858 (2000)
38. Polyá, G., Read, R.C.: *Combinatorial enumeration of groups, graphs, and chemical compounds*. Springer Science & Business Media, Berlin (2012)
39. Read, R.C.: Every one a winner or how to avoid isomorphism search when cataloguing combinatorial configurations. In: *Annals of Discrete Mathematics*, vol. 2, pp. 107–120. Elsevier, Amsterdam (1978)
40. Russell, S., Norvig, P.: *Artificial intelligence: a modern approach* (2002)
41. Sanger, F., Nicklen, S., Coulson, A.: DNA sequencing with chain-terminating inhibitors. *Proc. Nat. Acad. Sci. USA* **74**, 5463–5467 (1977)
42. Southern, E.: Analyzing polynucleotide sequences. International patent application PCT/GB89/00460 (1988)
43. Sutherland, G.: Dendral—a computer program for generating and filtering chemical structures. Technical report, Department of Computer Science, Stanford University of California (1967)
44. Torán, J.: On the hardness of graph isomorphism. *SIAM J. Comput.* **33**(5), 1093–1108 (2004)

Part III
Graph-Based Modelling in Art, Design
and Network Modelling

Chapter 11

Graph-Based Analysis of Drama Text—Case Study Based on Shakespeare’s ‘Measure for Measure’



V. Marceniuk, S. Zawiślak, and J. Kopeć

Abstract In the present paper some analyses of a drama text are performed. The adequate bibliography was listed. Graph theoretical approach is utilised. The modelling is done via steps (exemplary): (a) the graph vertices represent heroes and the graph edges their mutual connections, (b) the graph vertices represent words and the graph edges show relationships between particular words. This approach gives a new inside into the drama texts. Deeper understanding the vocabulary and characters connections and their mutual links in versatile aspects allow us for new evaluation and comparisons of dramas.

Keywords Drama · Graph-based analysis · Networks of terms and heroes

11.1 Introduction

Graphs [34, 35] have versatile application in many fields of knowledge like e.g. mechanics [39], logistics, computer science however also in humanities which is not widely known. An idea of text analysis, in general, has different aspects like e.g. content analysis, visualization of heroes family ties, showing relationships among drama heroes (like e.g. friendship, hatred, subordination or co-operation), plot analysis or vocabulary features etc. These directions of investigations were extensively developed in last decade. Modern computer software allows for preparation of versatile statistics, automatic summarizing, multi-aspect comparisons, word frequencies, word collocations and many other. The R package allows for versatile text analysis and then presenting them—especially—in form of graphs. The package is dedicated

V. Marceniuk · S. Zawiślak (✉) · J. Kopeć
University of Bielsko-Biala, Bielsko-Biala, Poland
e-mail: szawislak@ath.bielsko.pl

J. Kopeć
e-mail: jkopec@ath.bielsko.pl

for statistical analysis. Nevertheless, presenting results of theatre plays' texts analyses by means of weighted graphs allows for their easier understanding and catching the general view or structure at the first glimpse.

Graph theory [34, 35] is a branch of discrete mathematics which—to a certain degree—is unexpectedly applied also for humanistic sciences especially in last 20 years e.g. in course of the international conferences “Digital Humanities” and “Bridges” [32] and other [13]. Graph $G(V, E)$ is a pair of sets, where: V is a nonempty set of vertices and E is a set of edges. So, graph is an abstract notion however its image is immanently drawn in such a manner that vertices are represented as dots, circles or other pictograms and edges are shown as intervals of a straight line or arcs or parts of other curves connecting pairs of vertices. Simple graph is equivalent to a matrix and a relation. Therefore the mystery of graph modeling is hidden in equivalence of these three notions, as well! If we recognize characters as graph vertices then their mutual ‘relations’ (connections, ties) can be considered as graph edges. Other assignments of vertices and edges are also proposed.

Measure for Measure [27, 28] is one of the most famous and most widely known William Shakespeare's plays, which was published in 1604. Here, the rule was adopted that the play title is written in italics to avoid quotation marks. The action of the play was placed by the Author in Vienna, a town being a capital of a Roman Catholic country. Monks and nuns were among the heroes of the plays. However in Great Britain of that time—the homeland of the drama Author—Anglicanism was introduced by the King Henry the VIII (1491–1547). In consequence, monasteries were totally canceled. Therefore such a location was chosen in the discussed play. The same trick appears similarly in other Shakespeare's masterpieces like e.g. *Macbeth* or *Venice Merchant*. The plot is intriguing: Vincentio the Duke would like to recognize how is his ruling evaluated by the citizens. He had announced that he would like to travel abroad for some period. He officially transferred his ducal power to his courtiers. However, he stays in situ disguised as a friar using a big hood to be unrecognized.

Unfortunately, the people who remained in power shown their most ugly faces: they are cruel, disloyal as well as one of them would like to abuse a nun. The hidden Duke protects her against this villainy in a tricky way. Finally, he decides to come back to his tenable and due activities. Now, knowing the bad aspect of reality he “repairs” and restores the power practice and image—becoming the ruler again. Moreover, he announces some marriages which leads to the happy end—expected but surprising. Fortunately, due to the fact that the nun was a novice only therefore she could be a legal wife of the Duke. One marriage is forced by him to overcome the grievances suffered by a Prostitute. The *Measure for Measure* played in the “Teatr Polski” in Bielsko-Biala, Poland (see Fig. 11.1) has been an inspiration for the present considerations. The actors' costumes were proposed as contemporary clothes which even emphasized the universal message of the play. All aspects of the performance were striking. It is a really confirmation of the fact the Shakespeare's masterpieces are evergreens. The smart suites represent power as the corporate bosses (a, c), prisoners are in orange uniform like in some US movies (f, g), other members of the social

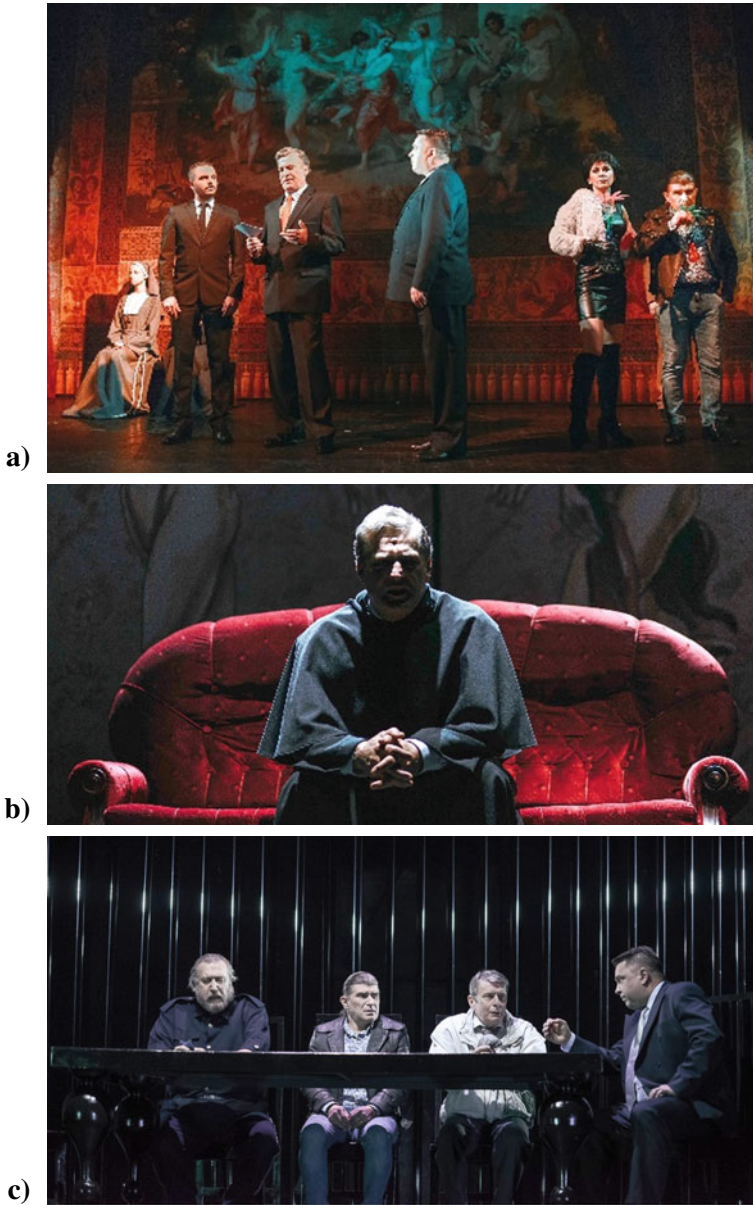


Fig. 11.1 Characters on the stage in theatre “Teatr Polski” of Bielsko-Biala **a**, **b** the Duke considers how cut the unfair plans of his co-workers, **c** meeting of provisional authorities, **c**, **f** nun with her rescued brother; **e** friar and **g** final scene—an announcement of termination of the masquerade and an offer of marriage



Fig. 11.1 (continued)

groups have also adequate contemporary typical clothes e.g. monks and nuns (b, d, e) instead of artificial theatrical costumes—see Fig. 11.1.

In the paper we propose to do analyses in two aspects: (i) creating graphs of heroes and subgroups of them; (ii) analyzing words spoken by heroes.

Together with other papers dedicated to the Shakespeare plays—the proposed graph-based analyses give a deeper insight into His idea of creating plots and their turns, capturing attention of spectators and readers, making groups of heroes simultaneously simple and untypical—in every case interesting—which are highlighted via the graphs presented in the current chapter.

11.2 Related Work

The topic of graph-based investigations of drama texts has been recently discussed in the series of papers in versatile aspects. However, these problems are generalized

not only on poetry and novels, but also on computer games, movies and all kinds of internet texts. We can roughly gather the cited publications in thematical groups. As we all know, the text is built of words so words themselves could be analyzed [2, 7, 13, 17]. In 1962 in [2] only statistical analyses were done. The statistical word analysis was performed by current authors in [37] as a base for simplification (rearrangement) of the considered graphs and digraphs. In [7], the problem is posed for the network texts. Works [13, 17] are dedicated to recognize proper meaning of the words in an automatic way—which is extremely important in automatic translation software. We have synonyms, phrases, blends or word clusters, adages or bywords, where proper meaning is sometimes hidden, at first glance even for an ordinary reader. So, in works [13, 17], it is explained how via graph-based analyses the adequate, current (only proper) meaning can be assigned or extracted via an electronic, algorithmic approach. The next investigated aspect is genre, plot and places analyses [1, 8, 11, 14, 26]. Work [14] is the pioneering one and it is recognized as a classical in the discussed field. In [11], methodology of network analysis is utilized (especially centrality) and in [26] multidimensional approach is proposed. The next aspects discussed recently in relation to drama analysis are social networks—in our case—related to plays, actors and directors or spectators [3, 7, 10, 16, 18, 19, 22, 30, 29]. The publications are dedicated to networks of characters in TV dramas [10, 16], texts [7, 18] or movies [3, 19]. The idea of clustering is also used in drama and cinema theoretical description [8]. Many William Shakespeare's plays were analyzed via graph-based approach e.g. *Romeo and Juliet* was considered in [2, 9, 11, 13]. The exemplary graph for heroes of *Romeo and Juliet* is drawn there in such a manner that the vertices are positioned regularly on the circle [4], size of the ellipsis or circle representing a particular character is adequate to its meaning in the discussed play. Some other analyses were made by the present authors in [37, 38], as well. Frank Harary—the icon of graph theory science—can be also recognized as the pioneer of this field graph based analysis of dramas [5, 6]. The research group of Professor Amalia Sparavigna prepared series papers in the discussed field, but here we enter on the bibliography list only three representative their papers [31, 30, 29]. Their graphs for heroes are presented for some world-wide known play i.e. *Hamlet* [31, 30] and *Divine Comedy* [29].

Unexpectedly—a classical dancing layouts could be modelled via graphs, as well, which was presented in one of the previous BRIDGES conferences [32]. The versatile reference analyses and reviews were published in [9, 14, 21, 33, 36, 37]. The really wide review is presented recently in [9]—where 311 bibliography items are listed and described. In the last type of publications, which could be recognized, practical tips are given for didactics related to an application of graphs in drama analysis [4, 23]. Item [4] gives us a link to webpage, where the classes scenario is described as well as the tasks for student are formalized via specially crated hand-outs. The classes are dedicated for students of the University Nebraska-Omaha (USA), where the idea of graph-based drama analysis is a regular one within the applied graph theory subject.

New and the most modern directions of investigations are interactive dramas and games, artificial plot, story generators and versatile text generators, especially for

social media. Usage of AI methodologies (e.g. agents) which can be assigned to some general trend—so called ‘reverse problems’ [12, 20, 21, 25] are also under rapid development. In [12], the inactive drama is describe. The user can choose arbitrary a turn points in the plot. The same ideas are described in [20] fully based on graph theoretical models. In [25], the most shocking idea of proto-narrative-ness is discussed.

In case of hand writing recognition we divide the methods depending on Latin or Chinese scripture—here Bengali [15], Early English [23] or Greek [22] texts are distinguished due to specific university knowledge needed for performance of their analysis. So, the proposed chapter joins in the popular and important trend of investigations.

11.3 Graphs Representing Heroes of the “Measure for Measure”

First considered approach is: play heroes are taken into account as graph vertices. The relations among them are presented e.g. in Fig. 11.2. They know each other therefore we can see the clique. Some graphs given in [7, 13] are prepared taking into account all the heroes in one graph, the same approach was utilized by the present author in [12]. However, it does not give too much. So, in [13] color edges as well as different size of vertices were used. Moreover, in [12] some graph transformations had been proposed aiming for simplification of the graphs which led to unexpected evaluations and conclusions. Moreover, here, the set of characters was divided in three groups arbitrarily based upon their social class membership which simplifies the adequate graphs.

To conclude, all the characters related to power are solely gathered like graph $G1$ shows. The vertex “prisoner” is shown by means of dotted line because he is not present on the stage. However he was beheaded and his cut head was presented on the stage. The ruler of the state—Duke is highlighted via double line vertex—as a ruler and a perpetrator of the whole affair. In case of graph G_2 , the tree presented in $G1$ via bold lines—is drawn separately. Bold edges represent a sub-relation of subordination of consecutive levels of power and prisoners without any power and civil liberties/rights are placed at the bottom of the scheme. Graph $G2$ was drawn using idea of so called physical approach where vertices have the same electric charge and edges are limp. The well-known method of graph drawing [40]. Therefore, the graph image is symmetrical and elegant. Structure is rigid and the heights of power are drawn in hierarchy one by one.

In case of graph $G3$, bold lines are utilized in the general clique K_7 . The protagonists know roughly each other, like among neighbors. Prostitute is not on the stage—her vertex, like previously, is drawn via dotted line. The bold line edges are between women which have more intensive relations (real friends) as well as between the low-life representatives. They create a clique K_3 . It symbolizes the fact that we usually

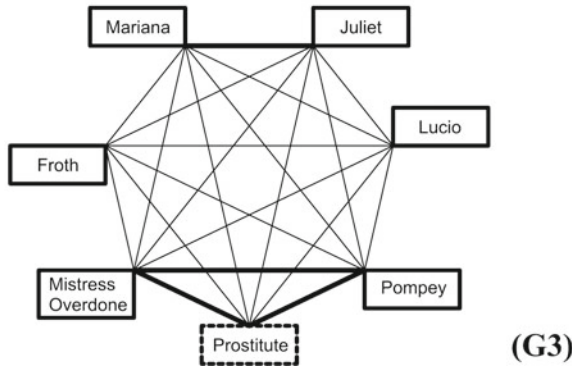
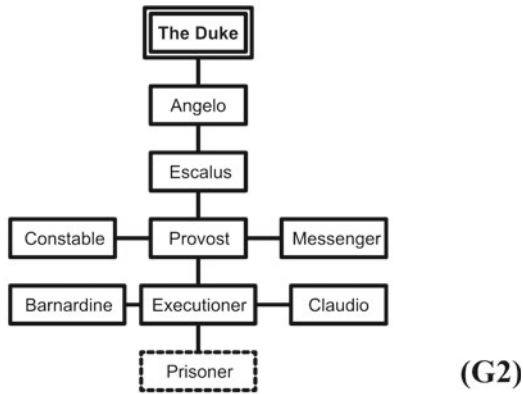
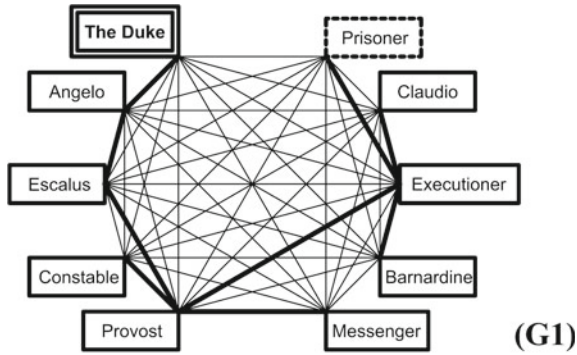


Fig. 11.2 Graphs representing the relations among character: *G1*: people of power and their subordinates; *G2*: extracted tree showing directly their subordination; *G3*: members of society

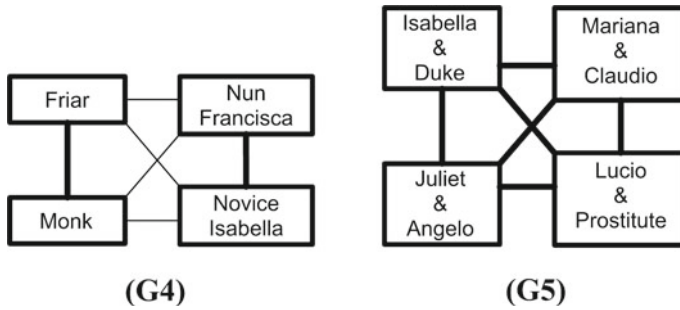


Fig. 11.3 Graphs representing the relations among church representative $G4$ and $G5$: representing the ‘happy end’ of the play

consider them as organized crime which was even recognized by Shakespeare more than 400 years ago.

In general, observations of the genial playwright are relevant not only to his contemporary state but also to many present governments and/or rulers in now-a-days countries.

The accusations, court trials and sentences on women abused are now disgusting, shocking and taking place in countries recognized as civilized, rich and modern.

In poor countries where power is weak, the abused people could not expect any protection or justice. Another graph approach was proposed by one author in paper [38].

In Fig. 11.3, the graph $G4$ is simple and the bold lines confirm independence between monks and nuns, they have different cloisters and different congregations. Taking part in social, ordinary live they know each other. There is exception for so called closed religious orders. In general, Shakespeare shows church as positive institution having clear but useful rules which are explained by the prioress (Francisca) to the novice (Isabel), in interesting and touching scene. Church is not interfere in the state affairs or activities, rather being active among the faithful.

The last presented graph (Fig. 11.3) represents the ‘happy end’ of the play invented by the Author. There are four pairs which are turn into four marriages. The graph has four symmetry axis, so it is regular, elegant and magic... We can only ask if such final solution could be possible nowadays. Duke proposes wedding to Isabella who should leave/reject a novitiate and in consequence she would not make definitive and final religious vows. Claudio—dismissed prisoner and Mariana are happy via the ducal solution, as well. Shakespeare also invented—taking into account responsibility and dignity—the marriage of Lucio. It seems also a very fair and generous act. In geometrical interpretation via the last discussed graph, we can see an object having many symmetry axes what suggests order, beauty, stability and harmony. So, happy end is touching, festive and uplifting but also unexpected—really a masterpiece.

11.4 Graphs Representing Some Word Connections in *Measure for Measure*

Additionally, we can analyze the text looking for phrases, collocations, frequency of usage of some words as well as connections between heroes. The considerations related to the text analysis of the play *Measure for Measure* are based upon utilization of the dedicated text mining technology. The tasks are performed via the functionality *tidytext* in the R language and adequate software. The function, methods etc. related to that language are written underneath in italics. The play text was read in using special file i.e. *fileName*, the loading was implemented with help of the method *readChar*:

```
text <- readChar(fileName, file.info(fileName)$size). dChar(fileName,
file.info(fileName)$size).
```

As a result we have a character vector of length equal to the number of items read. Further we transform this character vector into the data table structure, which is called *dataframe* in R language:

```
text_df <- data_frame(text).
```

At this stage all the text is located in unique table cell in a column called by *text*. Then we split the column *text* into tokens (words) using the *tokenizers* package, splitting the table into one-word-per-row:

```
tidy_text <- text_df %>% unnest_tokens(word, text).
```

Note that the function *unnest_tokens* supports non-standard evaluation through the *tidyeval* framework. Here *tidy_text* is the *dataframe* structure which contains the unique column *word*.

Hereinafter, we use package *dplyr* enabling us Non-SQL queries for *dataframe* structures. In order to reduce the volume of the vocabulary of words contained in the text, they use so called “stop words”, which are words which are filtered out before or after processing of natural language data (text). The package *tidytext* offers us its own *dataframe* *stop_words*, containing column *word*, e.g. a, a’s, able, about, above etc. and *lexicon*, e.g. SMART, which can be used in text mining. Applying function *anti_join*:

```
tidy_text <- tidy_text %>% anti_join(stop_words)
```

we get *tidy_text* containing only meaningful words. The frequency of the words can be calculated with help of *count()* function from the package *dplyr*:

```
tidy_text %>% count(word, sort = TRUE).
```

So we get in *tidy_text* all the words, which are ordered due to their frequency indicated in the additional column *n*. In the case of *Measure for measure* the “top 15 words” are presented in Fig. 11.4.

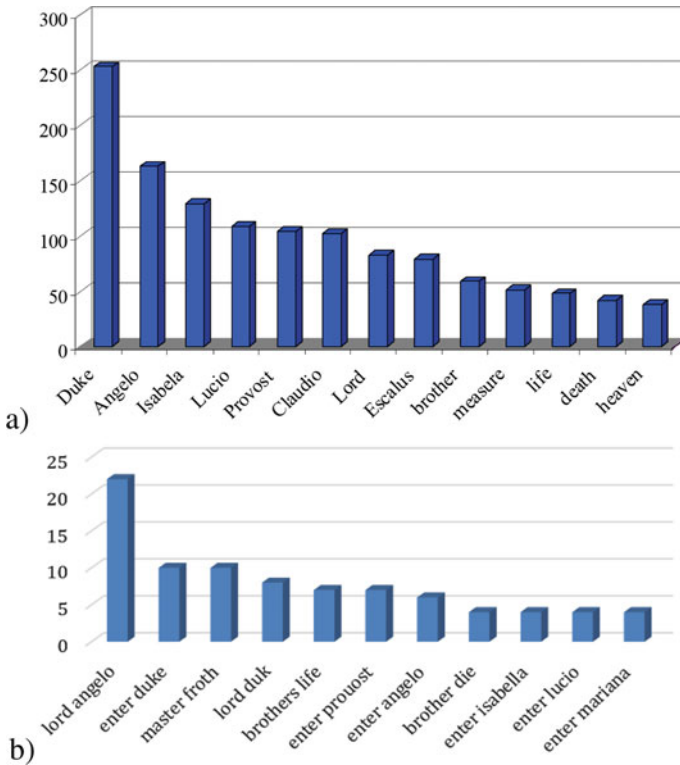


Fig. 11.4 a The most frequent meaningful words (names and notions) of the play “Measure for Measure”; b the most extended bigrams in the play “Measure for Measure”

As we can observe “Duke” is spoken most frequently, even if he is hidden for almost the whole action. Two pairs of heroes are mentioned also many times (100–150) so it underlines their meaning as most important characters of the play.

Note that some stop words are from old English, e.g. “haue”, “thou”, “doe”, “hath”, “thy”, “selfe”, “vpon”, “tis”, “thee”, “giue” etc. So, when analyzing such texts, they should be added. Thus, even in this primary stage of text mining we can acquire the most significant words in the text due to their frequencies. The next stage is dealing with establishment of relations between words. Of course, the most appropriate way is their presentation with help of graphs. Here we offer the usage of approach which is based on construction, processing, analysis and presentation of bigrams. Any bigram is pair of words joined with its frequency in the text. In our case we get all available bigrams with help of:

```
text_bigrams <- text_df %>% unnest_tokens(bigram, text, token = "ngrams",
n = 2)
```

which results in dataframe with a single column *bigram*, which contains all available text bigrams. In order to split words in bigrams we use function *separate()*:

```
bigrams_separated <- text_bigrams %>% separate(bigram, c("word1",
"word2"), sep = " ")
```

and get dataframe with columns *word1* and *word2* presenting two separate words in bigrams.

The simplest questions can be answered when analyzing the **bigrams containing certain words**. For example, we may investigate the participation of heroes in the scenes with the help of analyzing the frequencies of bigrams containing the word "enter". For this aim, we run the query

```
bigram_graph2_1 <- bigram_counts %>%
filter(word1%in%c("enter"))
```

as a result of which we get the bigrams in Fig. 11.5.

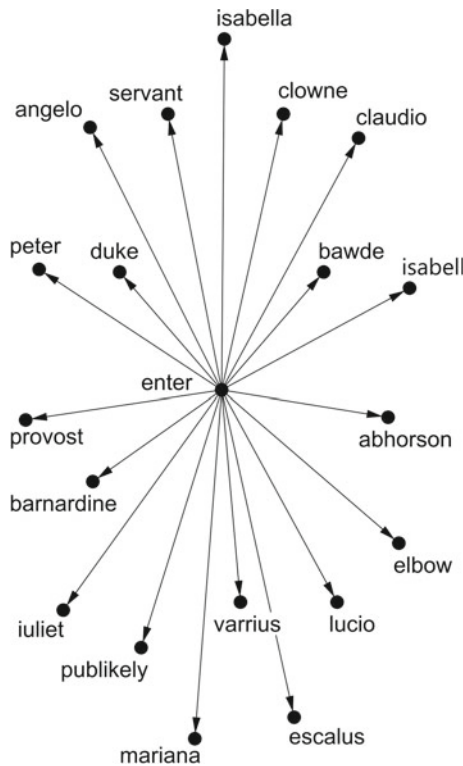


Fig. 11.5 Bigrams, which are constructed for the word "enter", indicating the frequency of participation of heroes in the scenes. On the right: decreasing frequencies of participation of heroes in the scenes, which are obtained from the bigrams

Here we can view the most extended bigrams in *Measure for Measure* (see Fig. 11.5). Pay attention that a lot of works should be done in order that pairs of words were omitted as the words serving for description of composition of play (e.g., “prima act”, “scene 1” etc.) but the word ‘enter’ was left.

The next processing of bigrams is dealing with applying different filters allowing us to get corresponding sets of bigrams in order to evidence existence and strength of certain relations in the text. We can research all relations with single word “*word_to_investigate*”. In order to get corresponding bigrams, we perform the query

```
bigram_counts % > % filter(word1 == "word_to_investigate" | word2 ==
"word_to_investigate")
```

As a result we have bigrams containing the word “*word_to_investigate*” as either *word1* or *word2* (see examples in Fig. 11.6a, b).

We can construct graph presenting bigrams containing two or more words. In case of words “marriage” and “prince” we run:

```
bigram_graph1 <- bigram_counts % > %
filter(word1%in% c("marriage","prince") | word2%in%
c("marriage","prince"));
```

the obtained graph is shown on Fig. 11.7. It is disconnected directed graph.

In order to look for possible paths between two words in the text we can do it in the following recursive way (see Algorithm 1).

Algorithm 1 Finding paths between two words

Algorithm 1. Finding paths between two words

Input: string *word1*, string *word2*, bigrams *D*

Output: *G* - graph, which is constructed from bigrams containing paths between *word1* and *word2*

int *i* = 1

Method:

```
construct_path (graph G, int i) {
while(! check_path_exists(D, word1,word2))
if (i=0) then G = construct_graph_for_all_bigrams_containing_words(word1,word2)
else {
G = construct_graph_for_all_bigrams_containing_words(words(G))
i++
construct_path(G,i)
}
}
return G
}
```

In order to find the path between two words we initialize *word1*, *word2* and bigrams *D* and call the method *construct_path* (*empty_set*, 0). When applying this method in our example we call at *i* = 1:

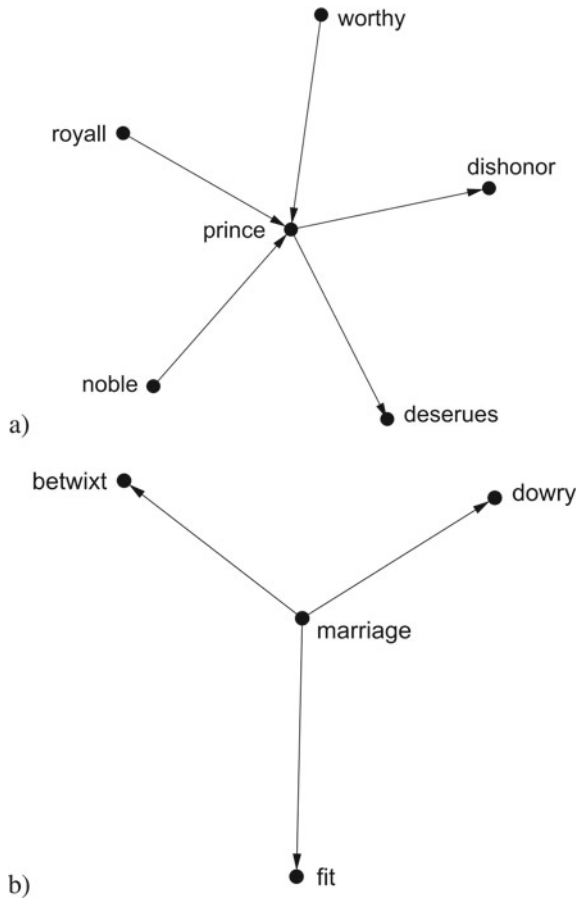


Fig. 11.6 Bigrams constructed for word “prince” on the basis of play ‘Measure for Measure’, **a** prince—noble, royal, worthy etc.; **b** marriage—fit, betwixt (between) and dowry

```

bigram_graph2 < - bigram_counts % > % filter(word1%in%
bigram_graph1$word1 | word1%in% bigram_graph1$word2 | word2%in%
bigram_graph1$word1 | word2%in% bigram_graph1$word2)
  
```

and the get graph on the Fig. 11.6a. At $i = 2$ we run:

```

bigram_graph3 < - bigram_counts % > % filter(word1%in%
bigram_graph2$word1 | word1%in% bigram_graph2$word2 | word2%in%
bigram_graph2$word1 | word2%in% bigram_graph2$word2);
  
```

obtaining the graph in Fig. 11.7b.

At this stage we can develop paths between two words.

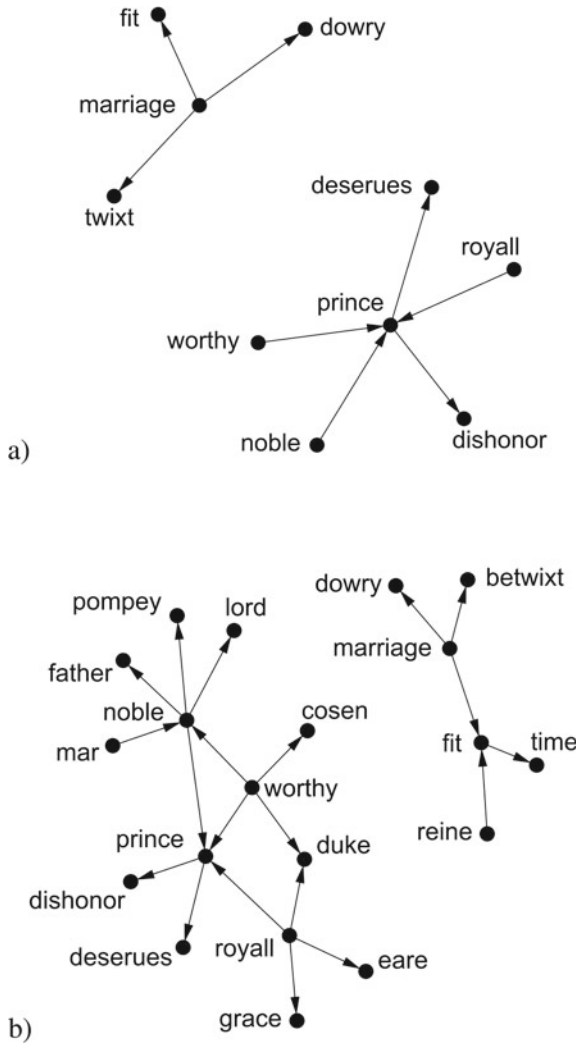


Fig. 11.7 Bigrams constructed for words “marriage” and “prince” for “Measure for Measure”: **a** $i = 1$; **b** $i = 2$

Continuing the previous example related to the word “enter” here we demonstrate the way to get all possible paths of the first order between the words “enter” and “duke”. At the first stage we construct all possible bigrams connected with word1 = “enter” and word2 = “duke” and vice versa

```
bigram_graph2_1 <- bigram_counts % > %
filter(word1%in% c(“enter”,“duke”) | word2%in% c(“enter”,“duke”))
```

The resulting graph is presented in Fig. 11.8.

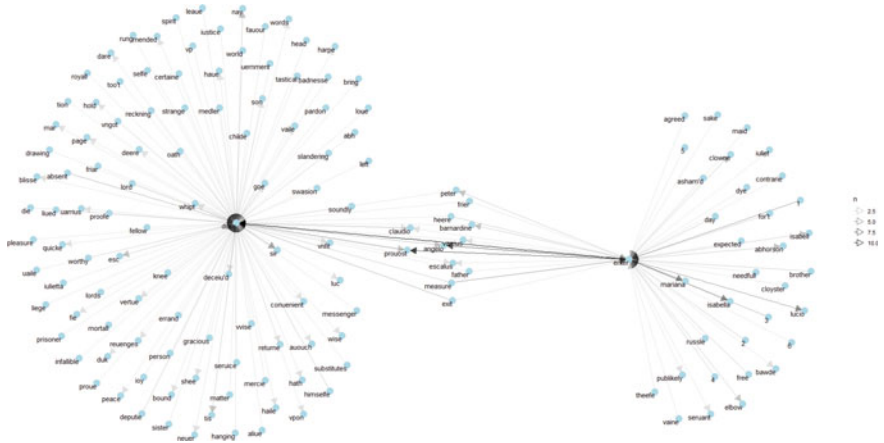


Fig. 11.8 All possible bigrams connected with word1 = “enter” and word2 = “duke” and vice versa; like it gives the visualization panel of the utilized software

During the next stage we search words, which are candidates to be connectors between “enter” and “duke”

```
bigram_graph_enter_2 <- (bigram_graph2_1% > %
filter(word1 ==“enter”))$word2
bigram_graph_1_enter <- (bigram_graph2_1% > %
filter(word2 ==“enter”))$word1
bigram_graph_duke_2 <- (bigram_graph2_1% > %
filter(word1 ==“duke”))$word2
bigram_graph_1_duke <- (bigram_graph2_1% > %
filter(word2 ==“duke”))$word1
```

and apply filtering in order to get possible paths

```
bigram_graph2_2 <- bigram_graph2_1% > %
filter((word1 ==“duke” & word2%in% bigram_graph_enter_2) | (word2
==“duke” & word1%in% bigram_graph_1_enter)|
(word1 ==“enter” & word2%in% bigram_graph_duke_2) | (word2 ==“enter”
& word1%in% bigram_graph_1_duke))
```

The resulting graph, which presents the first order paths between “enter” and “duke”, is displayed in Fig. 11.9.

The next our investigation is dealing with determining the significance of the words from viewpoint of bigram frequencies and relations within graph. Our approach is based on analysis of adjacency matrix which is constructed based on the graph G. The main idea of the offered method is the following. Let graph G contains words/vertices W_1, W_2, \dots, W_N . For any words W_i and W_j we denote bigram frequency as $n_{(i,j)}$.

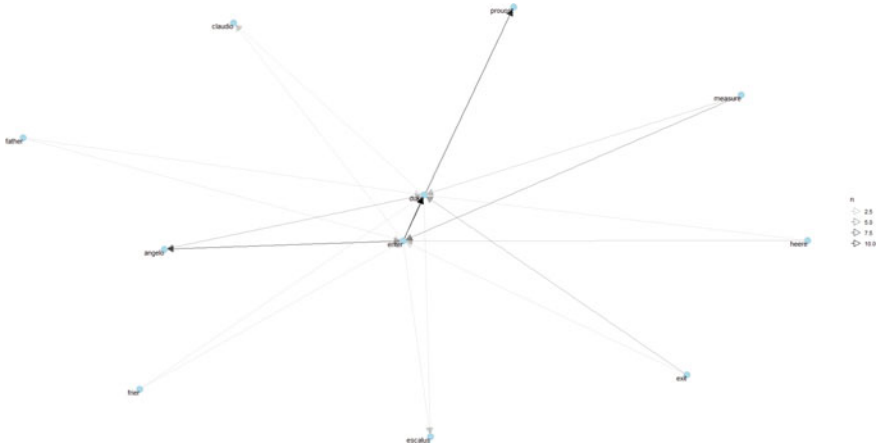


Fig. 11.9 The first order paths between words “enter” and “duke”

Consider adjacency matrix $A = \{n_{\{i,j\}}, i, j = 1, 2, \dots, N\}$. We assume that for arbitrary words W_i and W_j there exist significances w_i and w_j such that

$$\frac{w_i}{w_j} = n_{\{i,j\}}. \quad (1)$$

It implies

$$n_{\{i,j\}} \cdot w_j = w_i. \quad (2)$$

Let $w = (w_1, \dots, w_N)^T$. Hence

$$A \cdot w = N \cdot w. \quad (3)$$

Hence w is an eigenvector of A corresponding to the eigenvalue N . Unfortunately, in practice such eigenvector does not exist in every case. In order to overcome this shortcoming, T. Saati [24] proposed to search the vector w as eigenvector corresponding to the largest eigenvalue. That is background why we use Saati’s idea here, as well.

When applying this approach to our example the adjacency matrix can be constructed in R language as

```
# Make undirected graph so that matrix be symmetric
G <- graph.data.frame(bigram_graph, directed = FALSE)
# add value as a weight attribute
matrix_of_adjacency <- get.adjacency(G, attr = "n", sparse = FALSE)
```

Further we calculate eigenvector corresponding to the largest eigenvalue of the adjacency matrix:

```
# finding eigenvector which corresponds to the largest eigenvalue
ev.matrix_of_adjacency <- eigen(matrix_of_adjacency)
k <- which(abs(ev.matrix_of_adjacency$values) ==
max(abs(ev.matrix_of_adjacency$values)))
# eigenvector which corresponds to the largest eigenvalue
eigenvector <- ev.matrix_of_adjacency$vectors[,k]
```

The absolute values of eigenvalue elements correspond to the significance of the words. For example, in case of graph on Fig. 11.10 we have 216 words (unfortunately, including some words from old English). They can be ordered due to their significance in the following way (see Appendix)

11.5 Discussion and Conclusions

In the present paper, graph-based analyses were performed aiming for receiving a deeper insight into the chosen theatre play i.e. Shakespeare's *Measure for Measure*. The methodology is general and it could be utilized for other plays and/or novels. In fact, it was utilized previously by the authors in [37, 38]. The graphs giving relationships between the play characters highlight the ideas of the Author (Fig. 11.2): (a) power is well organized which is unveiled via symmetry of the graph representing subordination of its officers—from the top down; (b) church is not interfering in the political decisions—being a friar gives an opportunity for the Duke to observe the life of the society from the hidden position. The orders of nuns and monks are shown as independent. The third graph (c) grouping heroes is a picture representing society where the relationships are not described as fully strong. The society is full of divisions and individuals. The only closer feelings occur surprisingly among the outcast living on the outskirts of the society—creating a smallest possible clique.

The graph G5 in Fig. 11.2 represents graphically the happy end. However, the director of Bielsko-Biala performance, made a second version in Cracow where not all marriages were constituted. The idea behind was that not only the cloth and scenography are contemporary but also the social background. There were opinions that the pair Isabella and Duke suffers due to a generation gap and other matrimonies were made by forced. All these comments are totally ahistorical and do not take into account the knowledge about renaissance where all marriages were organized by parents, protectors, chancellors or rulers. Nevertheless, the people were happy due to contemporary beliefs. Moreover, Duke could support such wedlock (Lucio and Prostitute) for many years. In Poland, we have a tradition of the custody, patronage, alms and donations given by royal and ducal families to poor as well as to church. The first president of Polish Red Cross organization was Prince Stefan Sapieha—nobleman permanently involved in charity.

The wide analyses of references confirms that this direction of investigations is in rapid development nowadays, it is especially important is analysis of messages

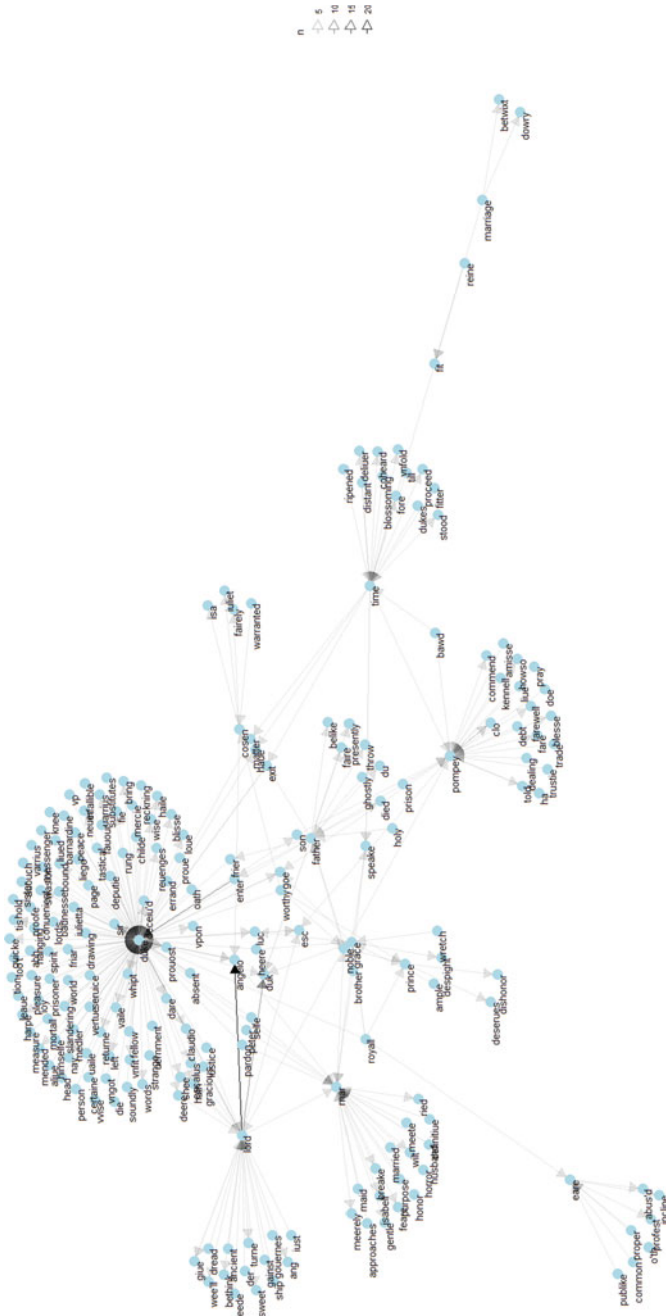


Fig. 11.10 Bigrams constructed for Algorithm 1 at $i = 3$ for “Measure for Measure”: Such an output gives the utilized software

in automatic and autonomous way, even writing automatic, artificial responses is considered, nowadays.

The graphs related to analysis of word connections allow for recognition of the phrases used by the Author as well as importance or repetition of some words. We can obtain the frequencies and collocations. We can measure such frequencies as e.g. speaking about particular heroes or their appearances on the stage. It all gives a deeper insight into creating a plot and story by the genius author.

The proposed approaches are general and overwhelming, their usability can be extended on other plays, which is also confirmed by listed references.

Acknowledgements Authors would like to express their cordial gratitude to the Authorities of Theatre “Teatr Polski” in Bielsko-Biała (especially for Mr. Witold Mazurkiewicz—managing and artistic director) for pictures and stills from the discussed play.

Appendix

The ordering of the significance of the words from graph of Fig. 11.6

colnames(matrix_of_adjacency)[order(eigenvector)]

[1]	"lord"	"angelo"	"duke"	"duk"	"enter"	"mar"	"giue"	"esc"
[9]	"claudio"	"deere"	"gracious"	"iustice"	"escalus"	"hath"	"shee"	"father"
[17]	"noble"	"exit"	"absent"	"ancient"	"der"	"dread"	"gainst"	"gouernes"
[25]	"indeede"	"iust"	"ang"	"bethink"	"ship"	"sweet"	"turne"	"wee'll"
[33]	"cosen"	"prouost"	"deputie"	"himselpe"	"measure"	"nay"	"tis"	
	"pardon"							
[41]	"peter"	"selfe"	"worthy"	"grace"	"frier"	"royall"	"goe"	"heere"
[49]	"luc"	"son"	"matter"	"haue"	"aliue"	"badnesse"	"barnardine"	
	"certaine"							
[57]	"childe"	"deceiu'd"	"die"	"drawing"	"errand"	"fauour"	"fellow"	
	"friar"							
[65]	"hanging"	"head"	"infallible"	"ioy"	"iulietta"	"knee"	"leauē"	"liege"
[73]	"liued"	"lords"	"medler"	"mercie"	"messenger"	"mortal"	"oath"	
	"person"							
[81]	"pleasure"	"prisoner"	"proofe"	"proue"	"reckning"	"rung"	"seruice"	
	"sir"							

[89] "sister" "soundly" "spirit" "strange" "substitutes" "swasion" "tastical"
 "tion"
 [97] "too't" "uaile" "uernment" "varrius" "vngot" "vp" "vwise" "whipt"
 [105] "world" "abh" "auouch" "blisse" "bound" "bring" "conuenient"
 "dare"
 [113] "fie" "haile" "harpe" "hold" "left" "loue" "mended" "neuer"
 [121] "page" "peace" "quicke" "returne" "reuenges" "slandering" "uarrius"
 "vaile"
 [129] "vertue" "vnfit" "vpon" "wise" "words" "husband" "pompey"
 "brother"
 [137] "speake" "approaches" "definitiu" "honor" "horror" "maid" "married"
 "meete"
 [145] "purpose" "breake" "feare" "gentle" "isabell" "meerely" "ried" "wilt"
 [153] "time" "prince" "prison" "ghostly" "holy" "belike" "died" "du"
 [161] "faire" "presently" "throw" "warranted" "fairly" "isa" "iuliet" "ample"
 [169] "wretch" "I" "despight" "eare" "bawd" "amisse" "clo" "dealing"
 [177] "debt" "doe" "kennell" "liue" "pray" "trade" "trustie" "blesse"
 [185] "commend" "fare" "farewell" "ha" "howso" "told" "fit"
 "blossoming"
 [193] "distant" "dukes" "fitter" "ripened" "till" "coheard" "deliuer" "fore"
 [201] "proceed" "stood" "vnfold" "deserues" "dishonor" "common" "o'th"
 "proper"
 [209] "publike" "abus'd" "incline" "profest" "marriage" "reine" "betwixt"
 "dowry"

References

1. Ardanuy, M.C., Sporleder, C.: Structure-based clustering of novels. In: Proceedings of the 3rd Workshop on Computational Linguistics for Literature (CLFL), pp. 31–39 (2014)
2. Bartkowiak, A., Gleichgewicht, B.: O długości sylabicznej wyrazów w tekstach autorów polskich (About the syllable-type length of words in the texts of Polish writers) (in Polish). *Appl. Math.* **6**(3), 309–319 (1962)
3. Chowdhury, T., et al.: Analysis of adapted films and stories based on social network. *IEEE Trans. Comput. Soc. Syst.* **6**(5), 858–869 (2019)
4. Hands-on Activity: Using Graph Theory to Analyze Drama, University of Nebraska-Omaha. https://www.teachengineering.org/activities/view/uno_graphtheory_lesson01_activity2
5. Harary, F.: Structural study of "a severed head". *Psychol. Rep.* **19**(2), 473–474 (1966)
6. Harary, F.: The love-hate structure of Dangerous Corner. *Semiotica* **54**(3–4), 387–393 (1985)
7. Hunter, S.: A novel method of network text analysis. *Open J. Mod. Ling.* **4**, 350–366 (2014)
8. Jung, J.J., You, E., Park, S.-B.: Emotion-based clustering for managing story-based contents: a cinemetric analysis. *Multimed. Tools Appl.* **65**, 29–45 (2013)
9. Labatut, V., Bost, X.: Extraction and analysis of fictional character networks: a survey. *ACM Comput. Surv. (CSUR)* **52**(5), 1–40 (2019)
10. Lee, J., Yeung, Ch.Y.: Extracting networks of people and places from literary texts. In: Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation, pp. 209–218 (2012)
11. Masías, V.H., Baldwin, P., Laengle, S., Vargas, A., Crespo, F.A.: Exploring the prominence of Romeo and Juliet's characters using weighted centrality measures. *Digit. Scholarsh. Hum.* **32**(4), 837–858 (2016)
12. Mateas, M., Stern, A.: Façade: an experiment in building a fully-realized interactive drama. In: Game Developers Conference '2003, pp. 4–8 (2003)

13. Mihalcea, R.: Unsupervised large-vocabulary word sense disambiguation with graph-based algorithms for sequence data labeling. In: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing. ACL, pp. 411–418 (2005)
14. Moretti, F.: Network theory, plot analysis. *New Left Rev.* **68**, 80–102 (2011)
15. Muhuri, S., Chakraborty, S., Chakraborty, S.N.: Extracting social network and character categorization from Bengali literature. *IEEE Trans. Comput. Soc. Syst.* **5**(2), 371–381 (2018)
16. Nan, Ch.-J., Kim, K.-M., Zhang, B.-T.: Social network analysis of TV drama characters via deep concept hierarchies. In: 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 831–836 (2015)
17. Navigli, R.: Word sense disambiguation: a survey. *ACM Comput. Surv.* **41**(2), Article 10, 1–69 (2009)
18. Park, G.-M., Kim, S.-H., Cho, H.-G.: Structural analysis on social network constructed from characters in literature texts. *J. Comput.* **8**(9), 2442–2447 (2013)
19. Park, S.-B., Oh, K.-J., Jo, G.-S.: Social network analysis in a movie using character-net. *Multimed. Tools Appl.* **59**(2), 601–627 (2012)
20. Partan, N., et al.: Evaluation of an automatically-constructed graph-based representation for interactive narrative. In: Proceedings of the 14th International Conference on the Foundations of Digital Games, pp. 1–9 (2019)
21. Roberts, D.L., Isbell, C.L.: A survey and qualitative analysis of recent advances in drama management. *Int. Trans. Syst. Sci. Appl.* **4**(2), 61–75 (2008)
22. Rydberg-Cox, J.: Social networks and the language of Greek tragedy. *J. Chicago Colloq. Digit. Hum. Comput. Sci.* **1**(3), 11 pp (2011)
23. Rzepka, A., Williams, P., Royston, J.: The social network of early English drama: a digital humanities lesson plan. *Emerg. Learn. Des. J.* **5**(1), 4 pp (2017)
24. Saaty, T.L.: *Fundamentals of Decision Making and Priority Theory*. RWS Publications, Pittsburgh, PA (1998)
25. Sack, G.A.: Character networks for narrative generation: structural balance theory and the emergence of proto-narratives. In: Workshop on Computational Models of Narrative 2013, Leibniz-Zentrum fuer Informatik, Dagstuhl Publishing, Dagstuhl, pp. 183–197 (2014)
26. Serino, M., D’Ambrosio, D., Ragozini, G.C.: Bridging social network analysis and field theory through multidimensional data analysis: the case of the theatrical field. *Poetics* 66–80 (2017)
27. Shakespeare, W.: *Measure for Measure* (in Polish). Wydawnictwo Literackie, Kraków (1983)
28. Shakespeare, W.: *Measure for Measure*. https://www.opensourceshakespeare.org/views/plays/play_view.php?WorkID=measure&Act=1&Scene=1&Scope=scene
29. Sparavigna, A.C.: Using time series and graphs in the analysis of Dante’s *Divine Comedy*. *Int. J. Sci.* **3**(12), 33–40 (2014)
30. Sparavigna, A.C., Marazzato, R.: Graph visualization software for networks of characters in plays. *Int. J. Sci.* **3**(2), 69–79 (2014)
31. Sparavigna, A.C., Marazzato, R.: Analysis of a play by means of CHAPLIN, the characters and places interaction network software. *Int. J. Sci.* **4**(3), 60–68 (2015)
32. Thomas, M., Peebles, C.: A graph-theoretic approach to the analysis of contra dances. In: Bridges Conference Proceedings, Jyväskylä, Finland, 9–13 Aug, pp. 285–292 (2016)
33. Wallner, G., Kriglstein, S.: Visualization-based analysis of gameplay data—a review of literature. *Entertain. Comput.* **4**(3), 143–155 (2013)
34. West, D.B.: *Introduction to Graph Theory*. Prentice Hall, Upper Saddle River, New Jersey (2001)
35. Wilson, R.J.: *Introduction to Graph Theory*. Addison Wesley Longman, Harlow, England (1998)
36. Xanthos, A., Pante, I., Rochat, Y., Grandjean, M.: *Visualizing the Dynamics of Character Networks*, DH 2016. Kraków, Poland (2016)
37. Zawisłak, S., Kopeć, J.: Graph-based analysis of Chekhov’s *Uncle Vanya*. *J. Hum. Math.* **9**(2), 157–186 (2019)
38. Zawisłak, S., Kopeć, J.: Theatre, love and graphs. In: Proceedings of APLIMAT, Bratislava, Slovakia, 6–8 Feb, pp. 1100–1111 (2018)

39. Zawiślak, S., Rysiński, J.: *Graph-Based Modelling in Engineering*. Springer, Cham (2017)
40. Zawiślak, S.: Annotated bibliography of the students' papers published in the proceedings—prepared under supervision of Stanisław Zawiślak. In: Zawiślak, S., Rysiński, J. (eds.) *Engineer of the XXI Century. Proceedings of the IX International Conference of Students, PhD Students and Young Scientists*, Bielsko-Biała, pp. 391–400 (2019)

Chapter 12

A Survey of Different Graph Structures Used in Modeling Design, Engineering and Computer Science Problems



Barbara Strug, Grażyna Ślusarczyk, Anna Paszyńska, and Wojciech Palacz

Abstract The paper presents several types of graphs used to model engineering, design, and computer science problems. In the described approach, graphs are used as the representation of the knowledge related to the considered problem, while so-called graph transformations are used to model the process of solving the problem. In the paper, different types of graphs and corresponding graph transformations, as well as their application to model design, engineering, and computer science problems, are presented. The labelled, attributed, directed and undirected variants of standard graphs, composition graphs, hierarchical composition graphs, hypergraphs, hierarchical hypergraphs, layout, and hierarchical layout graphs, as well as multi-hierarchical graphs are considered. The paper presents several applications of the described graph-based approach in architectural and engineering design, computational grids, Finite Element Method computations as well as in computer games.

Keywords Graphs · Graph-based knowledge representation · Graph transformations · Design · FEM

12.1 Introduction

This paper considers different types of graph structures, which can be used as a representation of knowledge, which is then processed using graph transformation rules, in

B. Strug (✉) · G. Ślusarczyk · A. Paszyńska · W. Palacz
Institute of Applied Computer Science, Jagiellonian University, Kraków, Poland
e-mail: barbara.strug@uj.edu.pl

G. Ślusarczyk
e-mail: grazyna.slusarczyk@uj.edu.pl

A. Paszyńska
e-mail: anna.paszynska@uj.edu.pl

W. Palacz
e-mail: wojciech.palacz@uj.edu.pl

order to model design, engineering, and computer science problems. Among many well-known methods of representing knowledge in computer systems, graph structures play an important role. Early solutions to the considered problem resulting from the conceptual process, in which the main components, their functions and interrelationships are established, can be efficiently represented by means of graphs. Then these structures can be easily modified using graph transformations [38, 52]. As graphs allow for expressing geometrical and semantic properties of object components together with relations between object components, in a uniform way, they are frequently used to model knowledge in contemporary computer tools [6, 26–28, 36, 48]. They can be used in design systems to model topological relations between components, in databases to model logical relations between objects as well as to model data and control flow or specification and analysis of software systems.

In our research, different types of graph structures have been used to represent knowledge according to the needs of applications in various domains, like architectural and engineering design, computational grids, FEM computation and computer games. The labelled, attributed, directed and undirected variants of standard graphs, composition graphs (*CP*-graphs), hierarchical composition graphs, hypergraphs, hierarchical hypergraphs, layout and hierarchical layout graphs, and multi-hierarchical graphs were considered.

In our early approach to design, structures of modelled objects were represented by labelled, attributed and directed standard graphs [13], and *CP*-graphs [11, 12, 25]. A *CP*-graph is a labelled and attributed graph, where nodes represent object components and are labelled by names of components they represent. To each node several bonds expressing potential connections between components are assigned. Bonds are connected by edges representing relations between components and labelled by the relation names. Attributes assigned to the graph nodes provide semantic information about components. This type of graph was also used to represent meshes in Finite Element Method [41].

In order to represent problems or artefacts with complex structures, hierarchical graphs [42], hierarchical composition graphs [15, 16, 47] and layered graphs [40] have been used. While composition graphs extend simple graphs by introducing node bonds, which represent fragments of components and allow us to define relations between these fragments, layered graphs being the other extension, are composed of disjunctive subgraphs called the layers and the edges representing relations between layers. Layered graphs are useful in representing complex problems that are semantically heterogeneous, related, for instance, to computer games, grids, and clouds. In hierarchical *CP*-graphs, subcomponents of object components are represented by other *CP*-graphs, which are nested in the nodes representing these components. Graph edges can connect nodes of different levels of hierarchy and having different parent nodes. Hierarchical *CP*-graphs allow for expressing hierarchical dependencies between different parts of designs, like inclusion of rooms in larger spaces.

However, neither layered graphs nor hierarchical *CP*-graphs are sufficient to efficiently model objects considered in some domains as they do not allow for expressing multi-argument relations between parts of different components. Therefore, in our subsequent research, we decided to use hypergraphs and hierarchical hypergraphs

[7] to represent designs [19, 21, 22, 43, 48] and FEM meshes [49]. Such graphs can represent objects with both multi-argument relations and hierarchical dependences between their components. Hypergraphs consist of nodes and hyperedges, which can connect an arbitrary number of nodes. The hyperedges are used to represent both relations and geometrical objects. In our approach hyperedges which represent object components are called object hyperedges, while hyperedges representing relations are called relational hyperedges. Nodes, which are assigned to object hyperedges and can be connected by relational hyperedges, represent fragments of object components. Object hyperedges which are used to represent hierarchical structures of object parts are called hierarchical. In such a hyperedge a hypergraph representing the internal structure of an object component is nested. The hierarchy of these graphs enables us to group elements according to specified objectives, while hyperedges can represent relations between components nested in different nodes representing various parts of objects. In [17] we have used hierarchical hypergraphs to represent building layouts in the conceptual phase of the design process.

Nevertheless, this representation is not uniform, as hyperedges represent both object components and relations between them, while fragments of components are represented by graph nodes. Therefore in [50] we have modified this representation to a more homogeneous one, called layout graphs (*L-graphs*), where graph nodes represent object components, while hyperedges represent multi-argument relations among them. In [52] the hierarchical layout graphs (*HL-graphs*) were used as the internal representation of knowledge about generated designs in the system supporting building design.

In some design tasks there exist many different types of relations between design parts, which cannot be expressed with a single hierarchy. For instance, different hierarchies are generally required for geometrical and functional dependencies. Therefore in [51] a graph-based representation with many hierarchies, which allow to express different types of hierarchical dependencies between design parts in one structure, has been proposed. As many multi-argument relations between components can be represented by hierarchical arrangement of the components in this graph structure, the hypergraph-based model was restricted to contain only binary relations between components, which means that graph nodes are connected by simple edges. Hyperedges representing design components have been replaced by nodes, while hyperedges expressing relations between components have been replaced by directed edges.

When objects are internally represented in a graph-based form, graph methods make it possible to integrate their representations and the process of generating their models [1, 7, 8, 24, 41]. Thus, graph grammars are widely used, especially in architectural design, urban planning [27] and engineering design [3, 5, 9, 10, 28, 39, 46, 53]. In this paper different types of graph rewriting systems, depending on the graph representation used, are described, and their application to model problems in architectural design, engineering design, computer games, computational grids and Finite Element Method are explained.

In [32, 33] the graphical application, called *GraphTool*, which provides a unified environment supporting operations on several types of graphs, like labelled,

attributed, directed and undirected variants of standard graphs, layered graphs, composition graphs, hierarchical composition graphs, hypergraphs, hierarchical hypergraphs, layout and hierarchical layout graphs, and the corresponding graph grammars transforming these structures, is described. The *GraphTool* system supports all steps required to define graph grammars and to control their application in order to obtain graph models of objects/tasks [37]. It provides graphical editors for graphs, graph transformation rules, and control diagrams. *GraphTool*'s ability to handle different types of graphs allows the user to choose the graph type which is the most convenient for a given application area.

12.2 Simple and Composition Graphs

Structures of simple objects can be represented using labelled attributed graphs which can be directed or non-directed. Such a graph is composed of sets of labelled nodes and edges, representing object components and relations between them. Attributes assigned to nodes describe properties of object components represented by these nodes.

Let Σ be a finite alphabet of node and edge labels. Let A be a finite set of attributes.

Definition 1 An **attributed labelled graph** G is a system

$G = (V, E, s, t, lab, atr)$, where:

1. V and E are disjoint finite sets of nodes and edges,
2. $s, t: E \rightarrow V$ are functions assigning to edges source and target nodes, respectively,
3. $lab: V \cup E \rightarrow \Sigma$ is a node and edge labelling function,
4. $atr: V \rightarrow 2^A$ is a node attributing function.

Example 1 When designing a girder, which should cover the span between given supports and transmit a given load to them, a plane simply supported truss can be considered [4]. The graph structure of this truss is presented in Fig. 12.1a. The nodes labelled *Upper chord* and *Lower chord* represent the chords, which are responsible for transmitting the bending moment, whereas the node labelled *Bracing* represents bracing which takes care of the shear force. The attributes assigned to nodes describe the sizes and materials of the corresponding components. A model of the truss represented by the graph from Fig. 12.1a is shown in Fig. 12.1b.

The structure being an extension of a simple graph, which allows us to represent relations between fragments of object components, is called a *composition graph* (*CP-graph*). A *CP-graph* is a labelled and attributed graph, whose nodes are equipped with bonds. Each node represents a part of a design, while bonds assigned to a node represent fragments of a part represented by this node. Bonds are connected by edges representing relations between fragments of parts represented by nodes. As nodes and edges can only represent the topology of a design, attributes are used to represent specific properties. Attributes can be assigned to all elements of a *CP-graph*, i.e. nodes, edges, and bonds.

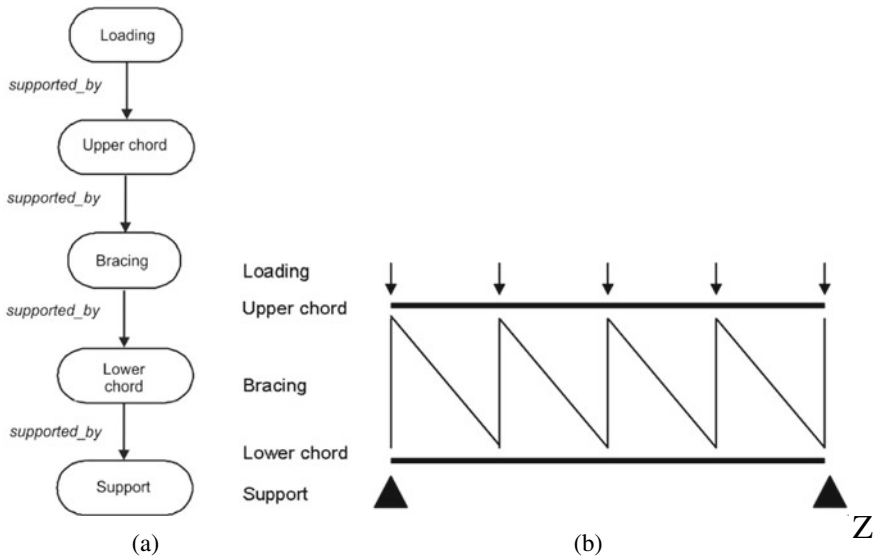


Fig. 12.1 A graph representation of a truss and a truss visualization [4]

The formal definition of a CP-graph is as follows.

Let Σ be a finite alphabet used to label object nodes, bond nodes and edges. Let A be a finite set of attributes.

Definition 2 A composition graph (CP-graph) G is a tuple

$G = (V, B, E, bd, s, t, lab, atr)$, where:

1. V, B, E are pairwise disjoint finite sets, whose elements are respectively called nodes, bonds and edges,
2. $bd: V \rightarrow 2^B$ is a function assigning sets of bonds to nodes, in such a way that $\forall x \in B \exists! y \in V: x \in bd(y)$, i.e., each bond belongs to exactly one node,
3. $s, t: E \rightarrow B$ are functions assigning to edges source and target bonds, respectively, in such a way that $\forall e \in E \exists x, y \in V: s(e) \in bd(x) \wedge t(e) \in bd(y) \wedge x \neq y$,
4. $lab: V \cup B \cup E \rightarrow \Sigma$ is a labelling function,
5. $atr: V \cup B \cup E \rightarrow 2^A$ is an attributing function.

Composition graphs have been used as internal representations of models in designing plane divisions in the Escher’s style [15], tea pots [2], and bridges [44].

Example 2 A CP-graph, which represents a structure of the bridge shown in Fig. 12.2b, is presented in Fig. 12.2a. It is undirected as its edges represent symmetrical relations. Nodes of the CP-graph represent abutments, pylons, beams, and two types of cables, which constitute structural components of the bridge, and are labelled as $ab, bm, pl, cb1$ and $cb2$, respectively. Bonds assigned to nodes are connected by undirected edges representing relations between bridge components. The edge labels fx and hn denote fixed and hinged connections between components of the bridge.

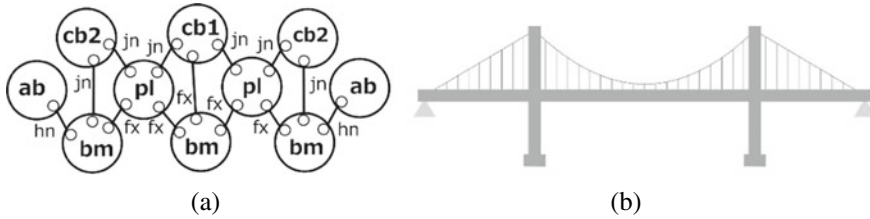


Fig. 12.2 A CP-graph representing a structure of a bridge and a bridge model

The edge label jn denotes the join relation. The node attribution function assigns to all nodes the attribute *length*, which specifies the length of the corresponding bridge fragments. Additionally, to nodes labelled pl and ab the attribute *position*, which specifies the location of pylons and abutments, respectively, is assigned. For the sake of simplicity, the attributes are omitted in Fig. 12.2a.

CP-graphs have also been used to represent meshes in the Finite Element Method (FEM) [41]. The Finite Element Method is used to find an approximate solution of the partial differential equation (PDE) defined over some finite element mesh covering the computational domain.

The approximation of the solution is represented as the linear combination of basis functions spread over nodes of the finite element mesh. As a result of the discretization of the PDE over the defined computational mesh, a global system of linear equations is generated. In order to increase the accuracy of the obtained solution, some adaptation can be used (for example h-adaptation—dividing elements into smaller elements).

Example 3 A CP-graph representing a simple finite element mesh (Fig. 12.3b) is presented in Fig. 12.3a. It is undirected as its edges represent symmetrical relations. Nodes of the CP-graph represent an interior node of a finite rectangular element (a graph node with label I), edge nodes of a finite rectangular element (graph nodes with label e) and vertices of a finite rectangular element (graph nodes with label v). Bonds assigned to nodes are connected by undirected edges representing relations between element nodes. Additionally, some attributes are assigned to nodes of the graph: attributes p , ph and p_v denote the polynomial order of approximation, attribute (x, y) denotes the coordinates of vertices of rectangular elements. Attributes hp , m , s , err are used to store additional information connected with the interior of the element: type of adaptation (attribute hp), element matrix (attribute m), solution (attribute s), and element error (attribute err).

The graph structures representing modelled objects can be dynamically generated by means of graph grammars. A graph grammar can be designed in such a way that it generates only graphs representing a particular class of objects.

A graph grammar consists of a set of graph transformation rules, which describe local modifications of graphs, together with a starting graph. Each rule is composed of the left and right-hand side graphs. A rule can be applied to a given host graph only

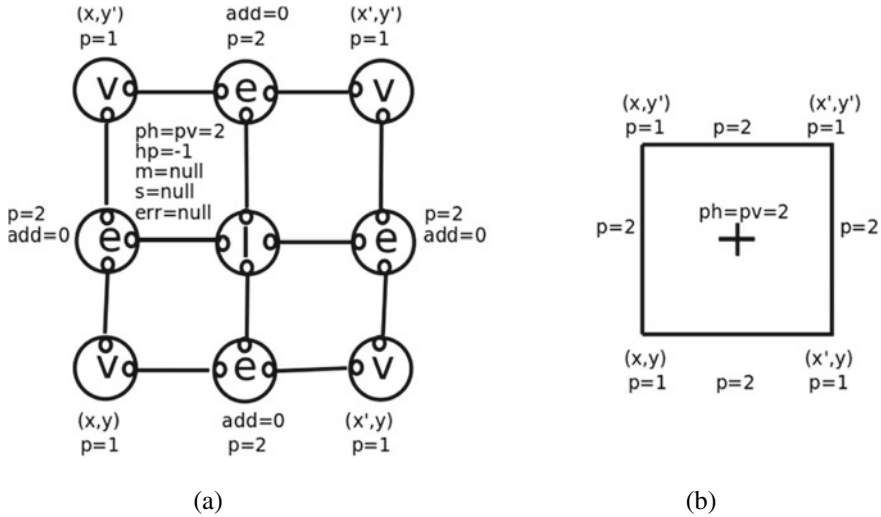


Fig. 12.3 An example of a one element mesh and a CP-graph representing it

if its left-hand side graph is isomorphic with a host subgraph. The found subgraph is then replaced by the right-hand side graph. The sequence of such replacements, which starts from the initial graph, is called a *derivation process*. There are systems where rules have additional application conditions (applicability predicates) or can modify values of attributes assigned to graph elements. A graph grammar can derive many different graphs from the starting graph by applying its rules in a different order. In some grammar types rules are chosen at random. In other grammars there are special mechanisms, e.g. a control diagram, to determine which rules should be considered at a given stage of the derivation process.

CP-graphs representing modelled objects in the above-mentioned fields of application have been generated using composition graph grammars. A CP-graph grammar is composed of a set of productions and an axiom being an initial CP-graph.

A CP-graph grammar rule is of the form $p = (l, r)$, where l and r are CP-graphs of the same type, i.e., having the same number of free bonds. The application of the rule p to a CP-graph G consists in substituting r for a CP-graph being an isomorphic image of l in G , replacing free bonds of the CP-graph being removed with the free bonds of r with the same numbers. After inserting r into a host CP-graph all edges which were coming into (or out of) a bond with a given number in the CP-graph l , are coming into (or out of) bonds of r with the same number.

Let N and T denote sets of non-terminal and terminal node labels, respectively.

Definition 3 A CP-graph grammar GG over N and T is a system

$$GG = (V_N, V_T, P, x), \text{ where:}$$

1. V_N, V_T are nonempty, finite sets of nodes with bonds labelled by elements of N and T , respectively,

2. P is a finite set of productions of the form $p = (l, r)$ satisfying the following conditions:
 - l and r are CP -graphs of the same type with ordered free bonds,
 - l contains at least one node with a label of N ,
3. x is an initial CP -graph containing at least one node with a label of N , and called an axiom of GG .

Example 4 Fourteen selected rules of a CP -graph grammar generating CP -graphs representing structures of bridge designs are shown in Fig. 12.4. These rules allow for adding the successive parts of designed bridges. Four bridge drawings being

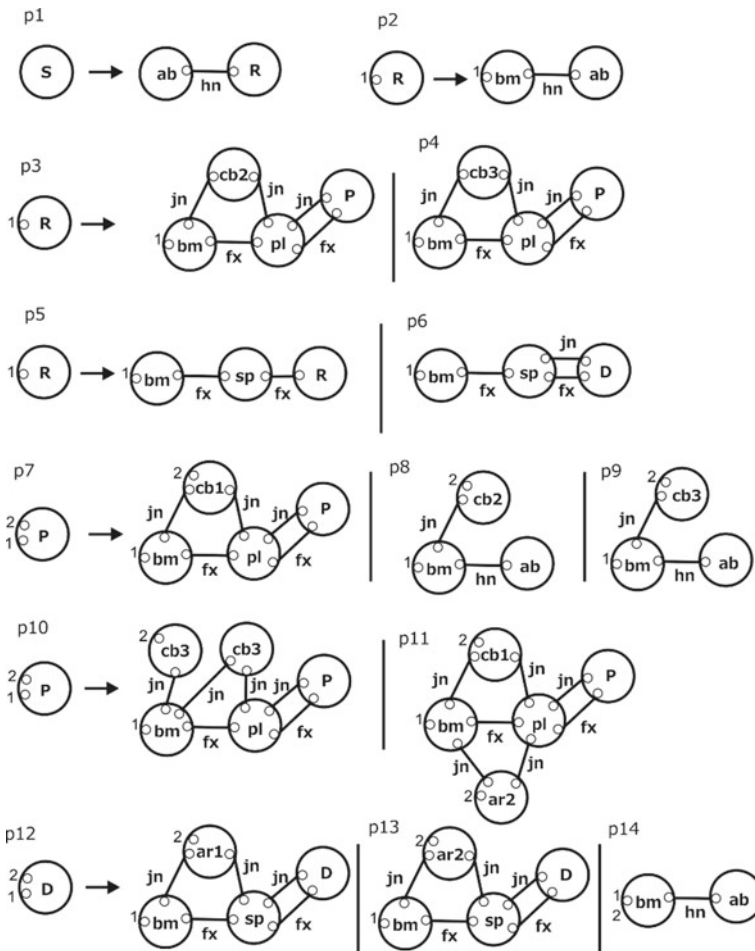


Fig. 12.4 Selected rules of a CP -graph grammar generating bridge designs



Fig. 12.5 Bridge drawings being visualizations of *CP*-graphs obtained using a *CP*-graph grammar from Fig. 12.4

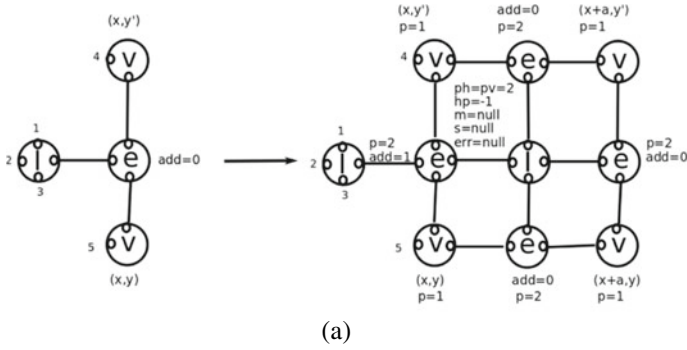
visualizations of *CP*-graphs obtained using this *CP*-graph grammar are illustrated in Fig. 12.5.

Selected rules of a *CP*-graph grammar generating structures of finite element meshes are shown in Fig. 12.6. The first production allows for adding the new rectangular element to an edge of an existing element. The second production allows for performing h-adaptation—dividing a rectangular element into four smaller elements. Figure 12.7 presents the *CP*-graph obtained from the *CP*-graph shown in Fig. 12.3a by applying the production from Fig. 12.6a.

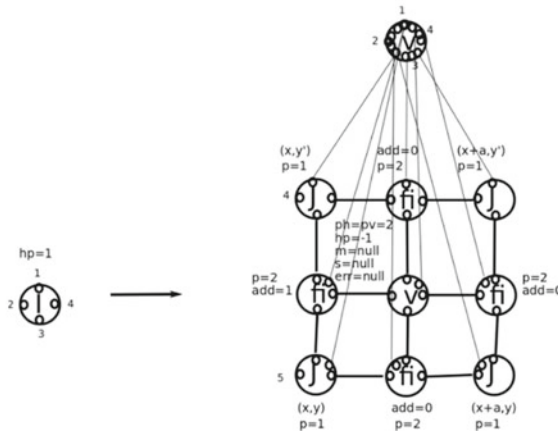
The presented graph grammar approach models finite element mesh generation as well as finite element mesh adaptation. The advantage of the model is the correctness of the generated mesh and the simplicity of the parallelization of the mesh generation and adaptation process (the graph grammar productions can be seen as atomic tasks) [35]. Another advantage is ensuring that the anisotropic adaptation process is performed in such a way that no deadlocks appear [45]. The model was also used to speed up the solver algorithm by adding new productions generating the quasi-optimal ordering in which the computations are performed [34].

12.3 Hierarchical Graphs and Hierarchical Composition Graphs

Objects with complex structures are often internally represented using hierarchical graphs. Hierarchical graphs allow for expressing hierarchical dependencies between different parts of objects. In each graph node representing a component of the given object, subordinate nodes and edges, which describe the inner structure of that component, can be nested. These child nodes can contain their own subordinate nodes and edges, thus creating a recursive hierarchy of components, subcomponents, sub-subcomponents and so on.



(a)



(b)

Fig. 12.6 Selected rules of a CP-graph grammar generating finite element meshes

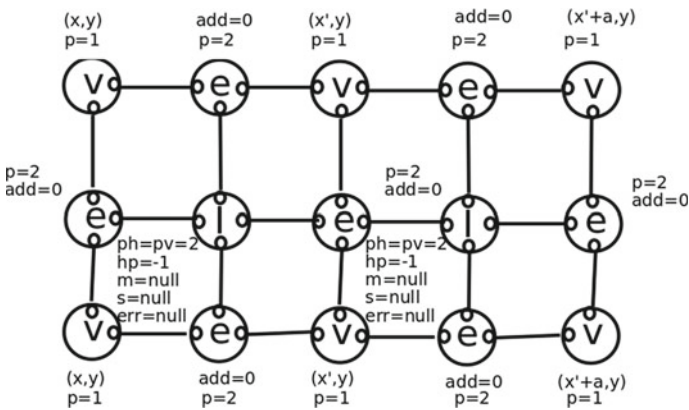


Fig. 12.7 The CP-graph obtained after applying the production from Fig. 12.6a to the CP-graph from Fig. 12.3a

Definition 4 A hierarchical attributed labelled graph G is a system

$$G=(V, E, s, t, lab, atr, par), \text{ where:}$$

1. V, E, s, t, lab and atr are defined as for a non-hierarchical graph (Def. 1),
2. $par: V \cup E \rightarrow V \cup \{\perp\}$ is a parent assigning function (symbol \perp indicates that a given node has no parent), specified in such a way that no edge or node can be its own ancestor.

Hierarchical graphs have been used as an internal representation of designed gardens, furniture [42] and plants [30].

Example 5 Figure 12.8 provides an example of a hierarchical graph which models a structure of a garden. The topmost node represents the whole garden. This node has three child nodes, which represent a path, a pond, and a group of trees. The pond node and the tree group node have child nodes of their own, which correspond to nine bushes growing on the pond’s shore and three individual trees forming this group. An edge connecting the path node with the pond node represents their adjacency.

Another type of hierarchical graphs are hierarchical composition graphs, where in CP -graph nodes other CP -graphs representing structures of subcomponents of components corresponding to these nodes can be nested. Hierarchical graphs allow for expressing relations between subcomponents of the designed object by means of edges connecting nodes having different parent nodes, i.e., being on different levels of hierarchy.

Definition 5 A hierarchical composition graph H is a tuple

$$H=(V, B, E, bd, s, t, lab, atr, ch), \text{ where:}$$

1. V, B, E, bd, s, t, lab and atr are defined as for a CP -graph (Def. 2),
2. $ch: V \rightarrow 2^{V \cup B \cup E}$ is a child nesting function such that:

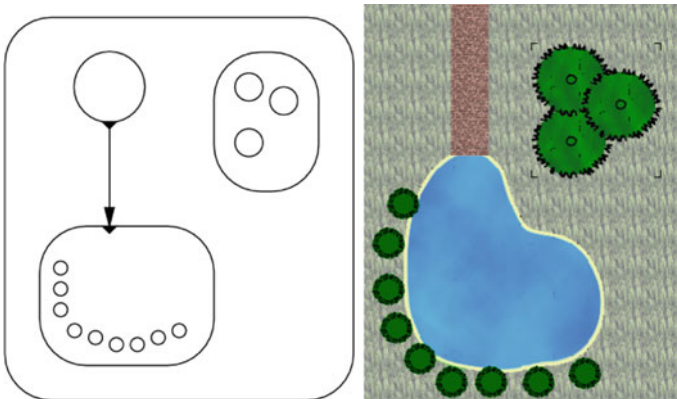


Fig. 12.8 A hierarchical graph representing a garden and its visualization

- $\forall z \in V \cup B \cup E, \forall v, w \in V, z \in ch(v) \wedge z \in ch(w) \Rightarrow v = w$, i.e., an element cannot have two distinct parents,
- $\forall v \in V v \notin ch^+(v)$, i.e., no element can be its own child ($ch^+(v)$ denotes a set of all descendants of the graph node v).

Hierarchical composition graphs have been used as internal representations of garden designs [14], plane divisions in the Escher’s style based on triangular grids [15] and in modelling skeletal structures in the optimization process [5].

Example 6 Figure 12.9a shows an example of a hierarchical composition graph representing a topology of a designed transmission tower (Fig. 12.9b). Nodes labelled T, P and B are hierarchical and they contain child nodes labelled x and K , where x represents a segment of the structure and K represents an endpoint segment to which a transmission line can be attached. Figure 12.9b shows only the top-level structure of a tower.

When structures of design objects are described in terms of hierarchical composition graphs, hierarchical composition graph grammars can serve as efficient tools for generating these structures. A hierarchical CP -graph grammar is composed of a set of rules, where both sides of each rule are hierarchical composition graphs with the same number of free bonds, and an axiom being an initial hierarchical CP -graph.

Definition 6 A hierarchical composition grammar hGG over N and T is a system $hGG = (V_N, V_T, P, x)$, where:

1. V_N, V_T are nonempty, finite sets of nodes with bonds labelled by elements of N and T , respectively,
2. P is a finite set of productions of the form $p = (l, r)$ satisfying the following conditions:

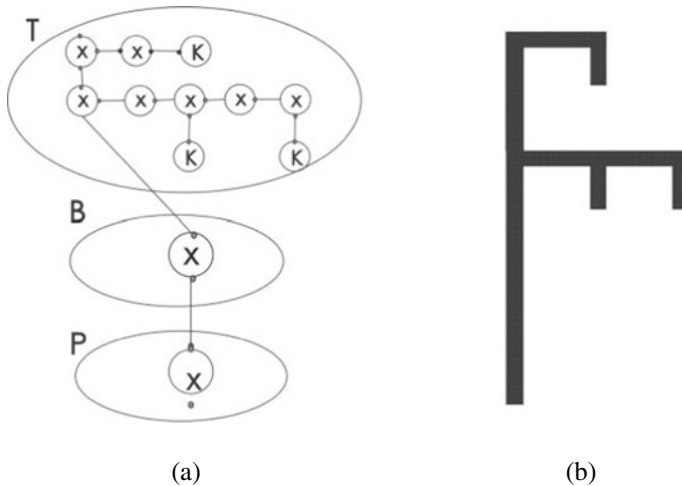


Fig. 12.9 A hierarchical CP -graph and the transmission tower topology represented by it

- l and r are hierarchical CP -graphs of the same type with ordered free bonds,
 - l contains at least one node with a label of N ,
3. x is an initial hierarchical CP -graph containing at least one node with a label of N and called an axiom of hGG .

The application of the production p to a hierarchical composition graph consists in substituting r for a subgraph isomorphic with l and replacing each bond of the subgraph being removed by the external bond of r with the same order number.

The representation of towers in the form of hierarchical composition graphs has been used in an evolutionary optimization process. The initial population of transmission towers in this process has been generated using a hierarchical CP -graph grammar.

Example 7 Selected rules used in the process of generating hierarchical CP -graphs representing transmission towers are depicted in Fig. 12.10. Depending on the sequence of the rules used, different towers can be generated. An example of such a derivation process and the obtained tower are shown in Fig. 12.11.

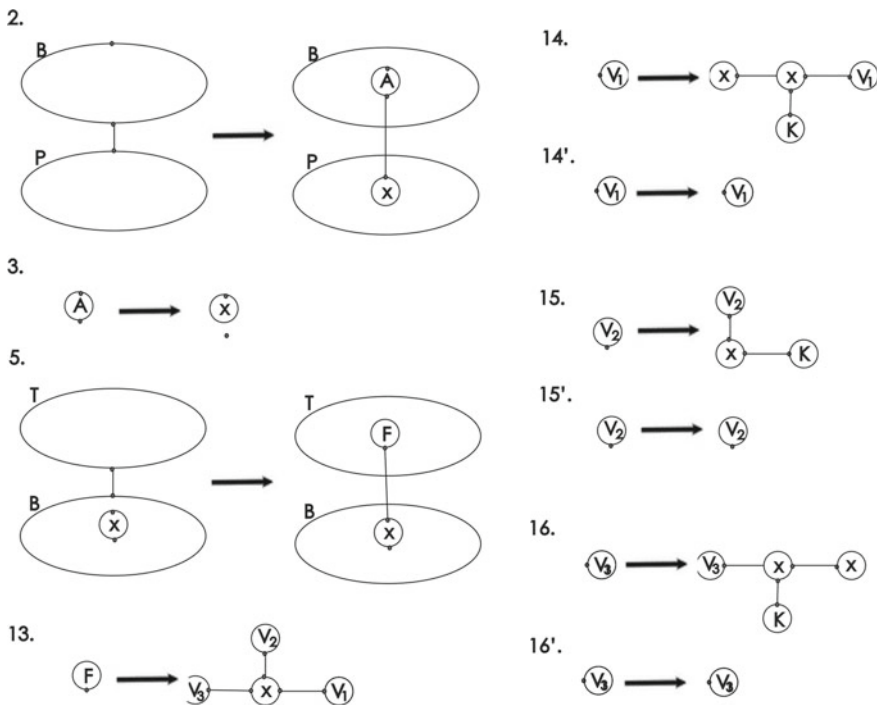


Fig. 12.10 Some rules of the hierarchical CP -graph grammar used to generate transmission towers

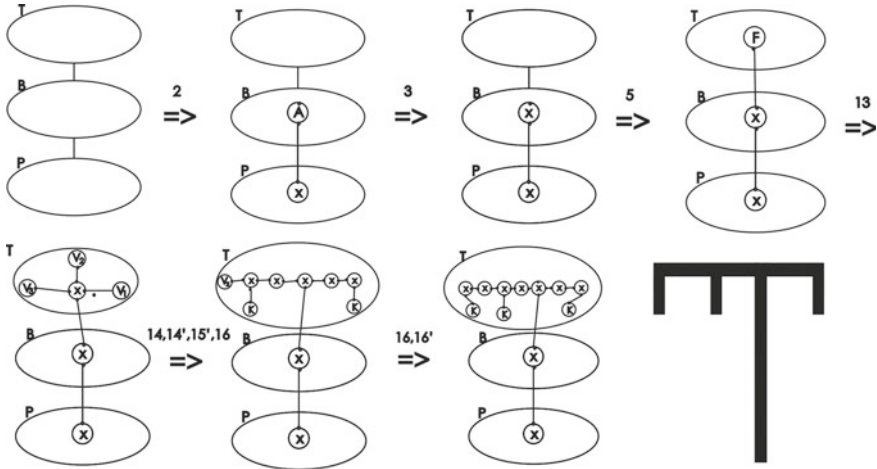


Fig. 12.11 An example of the derivation process

12.4 Layered Graphs

In hierarchical graphs, elements represented by nodes on different levels of hierarchy are of the same type. In some cases, a different approach is more suitable. For example, a computational grid contains different types of elements such as physical nodes (computers, computational elements), virtual nodes (software, operating systems, applications) and storage elements that can be treated both as physical elements (designated hard drives) or virtual ones (attached to other physical elements) [20, 31]. Such a structure requires using graphs which can represent both types of elements.

Moreover, some virtual elements of a grid are responsible for performing computational tasks on the grid, while other elements (services) are only responsible for managing the grid structure data, behaviour and the flow of the tasks. Thus, a representation for a grid should be able to adequately represent all elements, their interconnections and interdependencies. We introduce a notion of a *layer* composed of hierarchical graphs as a formal description of one part of a grid; for example, we can have a physical resources layer, a management layer, or a jobs layer. Each layer consists of one or more graphs representing parts of a grid. For example, at the resources layer a graph may represent a group of servers located at one place. As such parts of a grid are independent of each other, they are represented by disjointed graphs. On the other hand, each such graph consists of elements that can communicate with each other, so they are represented by connected graphs.

Formally, a layer is defined in the following way:

Definition 7 A **layer** L_y is a family of labelled attributed hierarchical graphs

$$L_y = \{G_1, G_2, \dots, G_n\}, \text{ where } n \in \mathbb{N}, \text{ and such that:}$$

1. $G_i = (V_i, E_i)$, is a connected graph, $1 \leq i, j \leq n$,

2. $V_i \cap V_j$ is an empty set for $i \neq j$,
3. $E_i \cap E_j$ is an empty set for $i \neq j$.

We propose to represent each of the grid layers as a layer composed of hierarchical graphs [31]. The graph layers are connected by interlayer edges which represent how the communication between different layers of a grid is carried out. As the layered graph represents the structure (topology) of a grid, semantics is needed to define the meaning, and thus possible uses, of its elements. Such information may include an operation system installed on a given computational element, specific software installed, size of memory or storage available, type of resource represented by a given element etc. This information is usually encoded in attributes assigned to elements of a graph. Each attribute is a function assigning values to attribute names. In a graph representing a given grid each node can have several attributes assigned, each of them having one value.

Let Σ_E be a set of edge labels and A_E be a set of edge attributes.

Definition 8 A **layered graph** GL is a set of layers

$GL = (\{L_{y1}, L_{y2}, \dots, L_{yk}\}, E, I_L, I_A)$, where

1. $E = \{e \mid e = (v_i, v_j); v_i \in V_{G_i}, v_j \in V_{G_j}, G_i \neq G_j, G_i \in L_{y_i}, G_j \in L_{y_j}, i \neq j\}$,
2. $I_L: E \rightarrow \Sigma_E$ is an edge labelling function,
3. $I_A: E \rightarrow 2^{A_E}$ is an edge attributing function.

Example 8 An example of a layered graph representing a simple grid is depicted in Fig. 12.12. To describe the grid we use the three-layered graph, where the resource layer, management layer and computing (physical) layer, labelled RL , ML and CL , respectively, are denoted by dashed lines. Let $\Sigma_V = \{C, CE, RT, ST, CM, index, services, broker, job\ scheduling, monitoring\}$ (where C stands for a computer, CE —for a computational element, CM —for a managing unit, RT —for a router and ST —for storage), be a set of node labels used in a grid representation. Each node label describes the type of a grid element represented by a given node. Let $\Sigma_E = \{isLocatedAt, hasACopyAt, actionPassing, infoPassing, taskPassing\}$ be a set of edge labels. The top layer of this graph, layer RL , represents the main resources/services responsible for task distributing/assigning/allocating and general grid behaviour. The second layer represents the elements responsible for the grid management. Each node labelled CM represents a management system for a part of a grid, for example, for a given subnetwork/computing element. The management elements CM can be hierarchical, as it is shown in the example. Such a hierarchy represents a situation in which data received from the grid services is distributed internally to lower-level managing components and each of them in turn is responsible for some computational units. At the same time each CM element can be responsible for managing one or more computational elements. The labels of edges are not written in this figure for clarity.

Grid elements, depending on their type, have some additional properties. These properties in a graph-based representation are described by attributes. Let attributes of nodes be defined on the basis of node labels. We also assume that attributes are

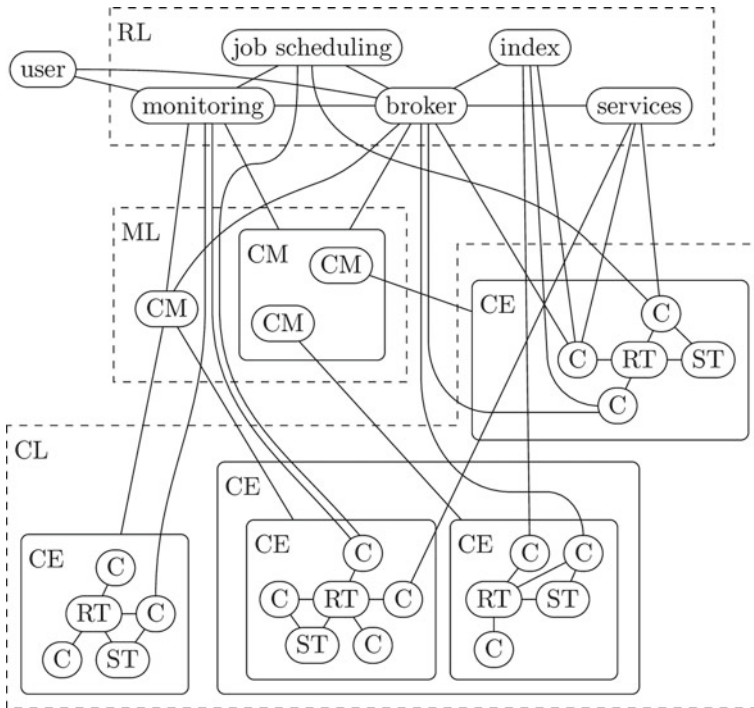
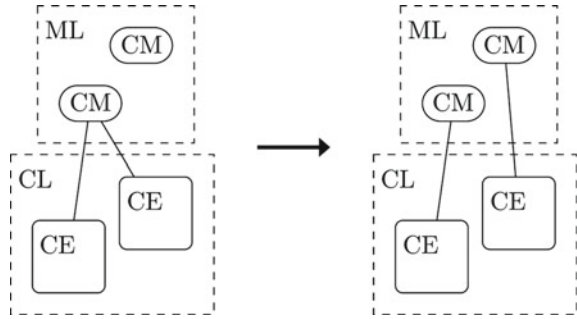


Fig. 12.12 An example of a layered graph representing a computational grid

defined for low-level nodes only. The attributes for their ancestors are computed on the basis of the child attributes. The set of node attributes used is $A_V = \{capacity, RAM, OS, CPU, apps, load, class, size, type\}$. The sets of attributes assigned to a given node are based on the label of this node. For example, to nodes labelled *C* a set of attributes $\{RAM, OS, CPU, apps\}$ is assigned, while to nodes labelled *CM* the attribute *load* is assigned. In order to assign values for node attributes, first a domain for each attribute has to be defined. For example, the domain for the attribute *OS*, $D_{OS} = \{Win, Lin, MacOS, GenUnix\}$. Then the values for the specified node attributes can be established.

The grid represented by a layered graph can be generated by means of a graph grammar. In a grid generation process, there is a need for two categories of productions, which can further be divided into five subtypes. The productions operating on only one layer can be based on traditional productions used in graph grammars. But, as we use a more complex structure, there is a need for productions operating on several layers. Moreover, we need productions that can both add and remove elements from the grid represented by the layered graph. To make sure that the functionality of a grid is preserved, the productions removing elements also have to be able to invoke rules responsible for maintaining the stability of affected elements. The application of a production can also require some actions to be invoked. For

Fig. 12.13 A production for dividing a manager



example, if a production removes a node representing a physical resource all other elements using this resource must be redirected to another resource, of the same type. If such a copy does not exist, it must be generated before the node can be removed from the grid. The productions used to generate a grid can be divided into five main types:

- working on two layers, but without adding edges or nodes (see Fig. 12.13)
- working on two layers, adding only edges
- working on two layers adding nodes and edges
- working on one layer and not requiring additional actions (see Fig. 12.14)
- working on one layer with additional actions required

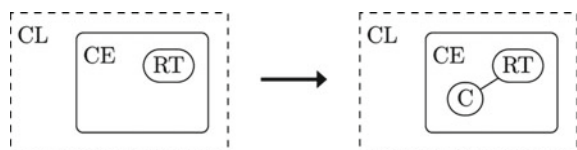
Example 9 Figure 12.13 displays a production, which moves a computing element between manager units. This is a type one production: it does not change the number of nodes and edges in the graph, it only changes the target of a pre-existing edge. Figure 12.14 displays a production which adds a computer. This is a production of type four.

By applying productions of the grid grammar, a grid structure can be generated. This approach also enables us to modify the grid in order to model its changing nature. As the layered graph represents not only the structure but also interconnections it can be used to simulate the working paradigm of the grid.

Layered graphs were also used to model the state of a computer adventure game [23]. This approach allowed for representing all aspects of a gameplay in a single graph structure, which has four layers corresponding to four types of objects used to describe the game and its state at a given moment.

Example 10 Figure 12.15 shows an example model of a game. The model has four

Fig. 12.14 A production for adding a computer to the grid



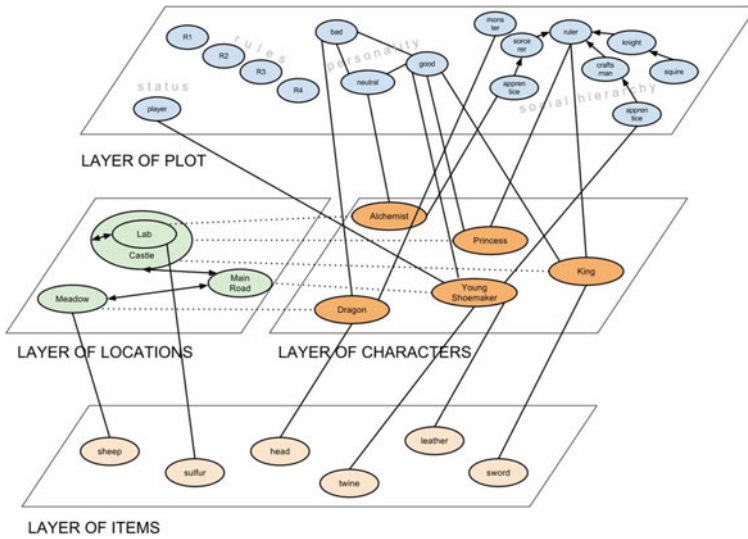
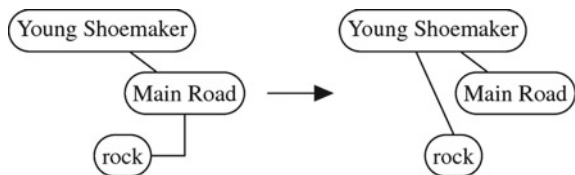


Fig. 12.15 A layered graph model of an adventure game

layers. The *location layer* contains nodes representing locations in the game world and edges specifying traversable paths between these locations. Thus, this layer can be treated as a map of the world. The *character layer* contains nodes representing characters. There is a node corresponding to the main character controlled by the player (in this example “Young Shoemaker”), as well as nodes representing allies and enemies of the hero. Every character node is connected by a single interlayer edge to a location node—these edges move during gameplay when characters walk between locations. The third layer is an *item layer*. Items can lie on the ground in a given location or they can be carried by a given character. In order to represent this fact every item node has a single edge which connects to a location node or a character node. The *plot layer* is dedicated to nodes representing immaterial plot elements. It can contain for example nodes representing character status (healthy, wounded, dead), character personality (good, neutral, bad), completion of quests, etc.

Figure 12.16 shows one of the transformation rules which are used during the play to change the graph model and thus the state of the game. This rule expresses

Fig. 12.16 A rule for picking up the rock



the fact that the “Young shoemaker” picks up a rock from the main road. It changes the target of the edge specifying the current placement of the rock.

12.5 Hypergraphs and Hierarchical Hypergraphs

In cases of design or modelling objects, where the representation of multi-argument relations between object components is needed, hypergraphs are used as the internal representation of objects. Hypergraphs are composed of nodes and two types of hyperedges. Hyperedges of the first type, called *object hyperedges*, are non-directed and correspond to object components. Hyperedges of the second type, called *relational hyperedges*, represent relations among fragments of these components. The fragments of components that can be used as arguments of relations are represented by hypergraph nodes. To each hyperedge of a hypergraph a sequence of different nodes is assigned. Hyperedges are labelled by names of components or relations. Both to hyperedges and nodes sets of attributes, which allow specifying properties of the corresponding object elements, are assigned.

Let Σ be a finite alphabet used to label nodes and hyperedges. Let A be a finite set of attributes. Let $[i]$ denote the interval $\{1, \dots, i\}$ for $i \geq 0$ (with $[0] = \emptyset$).

Definition 9 An **attributed hypergraph** G over Σ and A is a tuple

$G = (E, V, s, t, lb, att, ext)$, where:

1. $E = E_C \cup E_R$, where $E_C \cap E_R = \emptyset$, is a nonempty finite set of hyperedges, where elements of E_C (object hyperedges) represent object components, while elements of E_R (relational hyperedges) represent relations,
2. V is a nonempty finite set of nodes,
3. $s, t: E \rightarrow V^*$ are two mappings assigning sequences of different source and target nodes to hyperedges, respectively, in such a way that $\forall e \in E_C \ s(e) = t(e)$,
4. $lb: E \cup V \rightarrow \Sigma$ is a hyperedge and node labelling function,
5. $att: E \cup V \rightarrow 2^A$ is a hyperedge and node attributing function,
6. $ext: [n] \rightarrow V$ is a mapping specifying a sequence of hypergraph external nodes.

Hypergraphs have been used as internal representations in designing floor layouts [17], garden arrangements [18] and as a representation of finite element meshes [49].

Example 11 A hypergraph, which represents a floor layout shown in Fig. 12.17b, is depicted in Fig. 12.17a. Object hyperedges (denoted as rectangles) represent areas of different types, like *room*, *kitchen*, *hall* and *bathroom*. Relational hyperedges (denoted as ovals) represent relations between these areas, namely *adjacency* and *accessibility*. Nodes are depicted as small black circles.

A hypergraph representing a simple triangular finite element mesh is presented in Fig. 12.18. Nodes of the hypergraph represent vertices of a triangular element and have the label v . The hyperedge with label I denotes the interior node of a

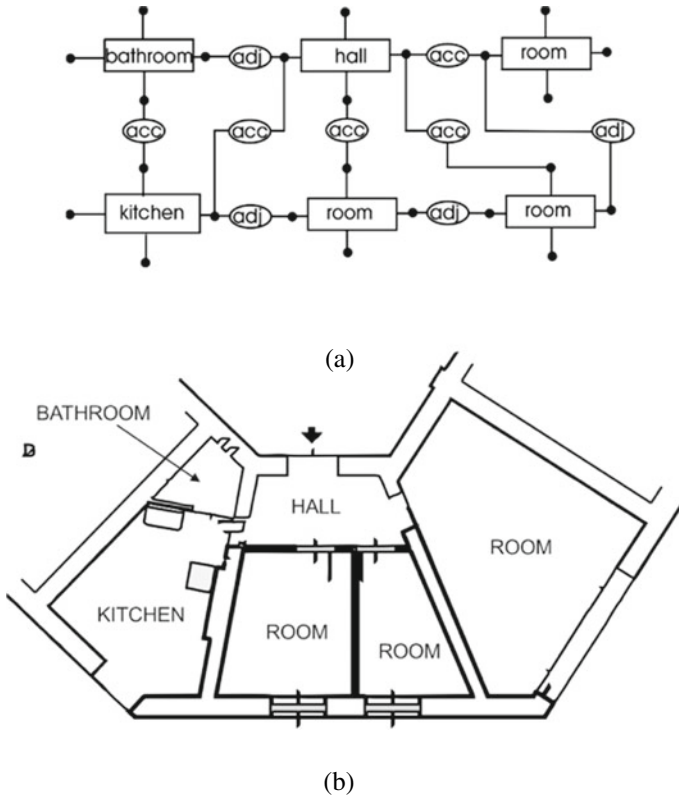


Fig. 12.17 A floor layout and its hypergraph representation

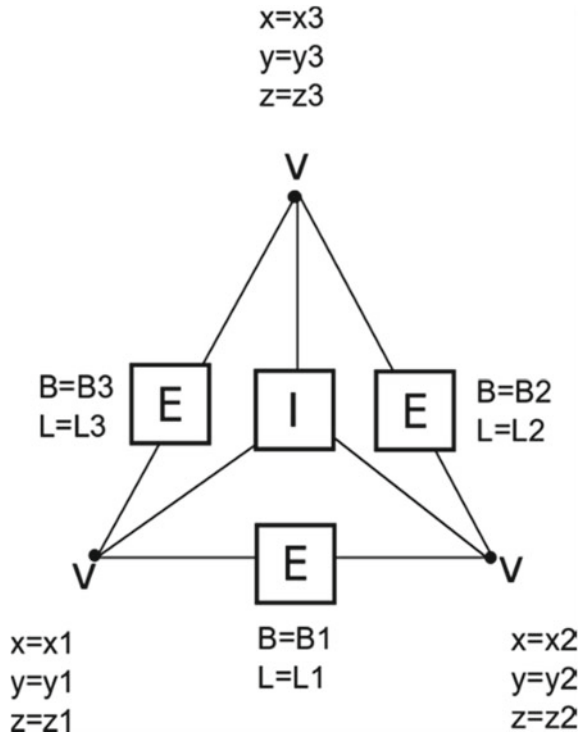
finite triangular element and the hyperedges with label E denote edge nodes of this element. Additionally, some attributes are assigned to nodes and hyperedges of the hypergraph: attributes x, y, z denote the coordinates of vertices of the triangular elements. Attribute L denotes the length of an edge of the triangular element and attribute B equals true if the edge is a boundary edge and false in the other case.

In the case of designing objects with complex structures, when the possibility of expressing the top-down way of the design solution development is needed, hierarchical hypergraphs are used. Object hyperedges which are used to represent hierarchical structures of design parts are called *hierarchical*. In such a hyperedge a hypergraph representing the internal structure of a design component is nested.

Hierarchical hyperedges may be used to hide certain details of a designed object that are not needed at a given stage of design or to group objects having some common features (geometrical or functional).

Let AT denote a set of hypergraph hyperedges and nodes called together *hypergraph atoms*.

Fig. 12.18 A hypergraph representing a one element triangular mesh

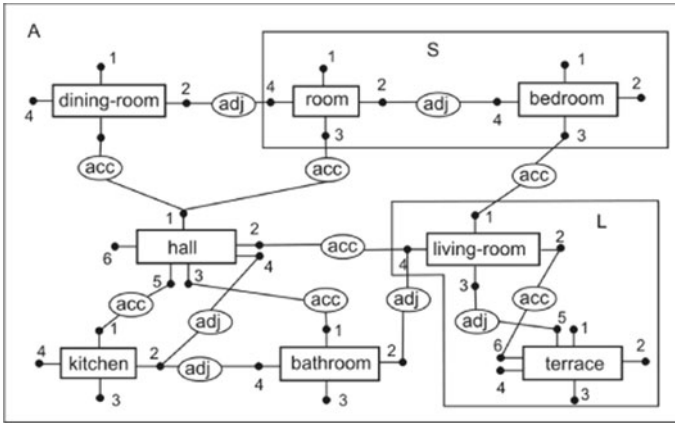


Definition 10 An **attributed hierarchical hypergraph** G over Σ and A is a tuple $G = (E, V, t, s, lb, att, ext, ch)$, where:

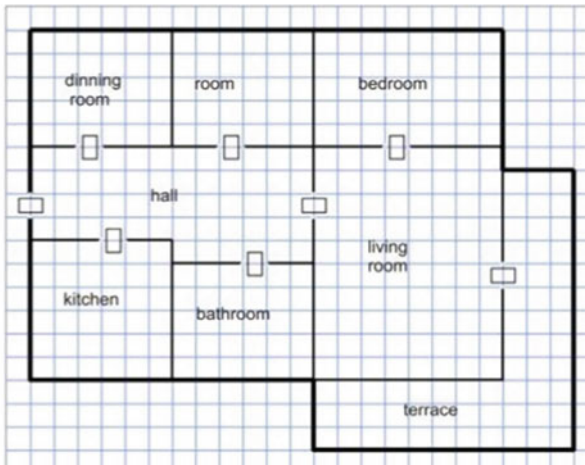
1. E, V, t, s, lb, att and ext are defined as for an attributed hypergraph (Def. 7),
2. $ch: E_C \rightarrow 2^{AT}$ is a child nesting function which places hierarchical hypergraphs in object hyperedges, and such that one atom cannot be nested in two different hyperedges, a hyperedge cannot be its own child, and source and target nodes of a nested hyperedge e are nested in the same hyperedge as e .

Hierarchical hypergraphs have been used as internal representations of building floor layouts [43] and transmission truss towers [48].

Example 12 An example of a hierarchical hypergraph representing a floor layout of the designed apartment shown in Fig. 12.19b is depicted in Fig. 12.19a. The object hyperedges represent components of the apartment. The hyperedges labelled *dining room, room, bedroom, hall, living room, kitchen, bathroom* and *terrace* represent different spaces of the layout. Relational hyperedges are labelled *acc* and *adj* and represent accessibility and adjacency relations, respectively. The nodes assigned to object hyperedges representing rooms denote walls of the rooms or sides in case of open rooms (like a terrace) and are numbered in a clock-wise manner according to the polygon sides which they represent, starting from the most top left one. The



(a)



(b)

Fig. 12.19 A floor layout diagram and a corresponding hierarchical hypergraph

attributes assigned to object hyperedges include *area*, which specifies the area of a space represented by a given hyperedge. The hypergraph contains four hierarchical object hyperedges. The first one (labelled *A*) represents the whole apartment and has three other hierarchical object hyperedges (labelled *S*, *L* and *R*) nested inside. The second hierarchical object hyperedge is labelled *S* and represents the sleeping area of the apartment. There are two non-hierarchical object hyperedges representing rooms of this area and the relational hyperedge representing adjacency between these rooms nested in it. The third hierarchical object hyperedge is labelled *L* and represents the

living area. It has two nested non-hierarchical object hyperedges representing the living room and the terrace and two relational hyperedges representing two relations between them: the accessibility through the east wall of the living room and the adjacency through its south wall. The fourth hierarchical object hyperedge is labelled R and represents the remaining area. It has four nested non-hierarchical object hyperedges representing the dining room, the hall, the kitchen and the bathroom, three relational hyperedges representing the accessibility of the dining room, the kitchen and the bathroom from the hall, and two relational hyperedges representing the adjacency between the hall and the kitchen and between the kitchen and the bathroom. The hierarchical object hyperedges representing areas of the apartment do not have any nodes assigned, as during the design process, when hypergraphs representing components of these areas and relations among them were nested in hyperedges, the nodes assigned to these hyperedges were substituted by the corresponding external nodes of the child hypergraphs. Thus, the role of nodes representing the sides of the areas has been taken over by the nodes attached to nested object hyperedges representing rooms of these areas.

A hierarchical hypergraph which represents the structure of the truss tower from Fig. 12.20b is presented in Fig. 12.20a. This hypergraph contains three hierarchical hyperedges labelled S_{gb} , S_{gm} , S_{gt} and representing three segments of the truss tower. Hyperedge S_{gb} contains two object hyperedges representing truss panels, while hyperedge S_{gm} contains four of them. Hyperedge S_{gt} contains seven object hyperedges representing horizontal trusses and four object hyperedges representing insulators. Hypergraph nodes represent truss nodes which connect trusses and insulators. Twenty-two relational hyperedges represent connections among trusses and

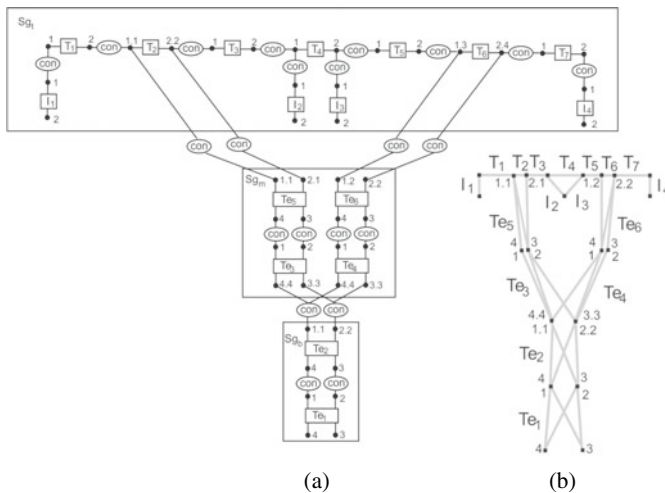


Fig. 12.20 A diagram of a transmission truss tower and a corresponding hierarchical hypergraph

between trusses and insulators. Attributes *width*, *height*, *position*, *type* and *material* are assigned to object hyperedges, while the attribute *order* which specifies the sequence of truss nodes, is assigned to hypergraph nodes.

Hypergraphs representing structures of design solutions are generated by hypergraph grammars. A hypergraph grammar is composed of a set of hypergraph edges with terminal and non-terminal labels, a set of hypergraph nodes, a set of productions and an axiom being an initial hypergraph. Each grammar production is of the form $p = (l, r)$, where l and r are attributed hypergraphs with the same number of ordered external nodes.

Let N and T denote sets of non-terminal and terminal labels, respectively. Let lb be a hyperedge labelling function.

Definition 11 A hypergraph grammar G over N and T is a system

$G = (V, E, P, X)$, where:

1. V is a finite set of nodes,
2. $E = E_N \cup E_T$ is a finite set of hyperedges, where $lb: E_N \rightarrow N$ assigns non-terminal labels to hyperedges of E_N , while $lb: E_T \rightarrow T$ assigns terminal labels to hyperedges of E_T ,
3. P is a finite set of productions of the form $p = (l, r)$ satisfying the following conditions:
 - l and r are attributed hypergraphs composed of nodes of V and hyperedges $E_N \cup E_T$ with the same number of ordered external nodes
 - l contains at least one hyperedge of E_N ,
4. X is an initial attributed hypergraph containing at least one hyperedge of E_N and called an axiom of G .

The application of the production p to a hypergraph H composed of nodes and hyperedges of $E_N \cup E_T$ consists in substituting r for a hypergraph isomorphic with l and replacing external nodes of the hypergraph being removed isomorphic with nodes of ext_l by the corresponding external nodes of ext_r .

Example 13 Some rules of the hypergraph grammar generating structures of floor layouts are presented in Fig. 12.21. The first production divides the floor area into three functional units, a hall, a kitchen and an antechamber. The second and third production allow choosing a layout with one or two bedrooms in the first sleeping area. The fourth production adds a living-room connected to a terrace. The productions $p5$ and $p6$ allow for obtaining different layouts composed of two rooms, a shower and a small hall. The former rule gives a possibility to locate the shower between the rooms, while the latter one allows the shower to be adjacent to the one room only, but to have two entrances, one from the *hall1* and the second from the *room*. To all hyperedges representing spaces and rooms attributes *area* and *room_number*, which specify the sizes of rooms and spaces, and the number of rooms in a given space, are assigned. Therefore, predicates of applicability for productions, which define possibility of their application provided sufficient sizes of the considered

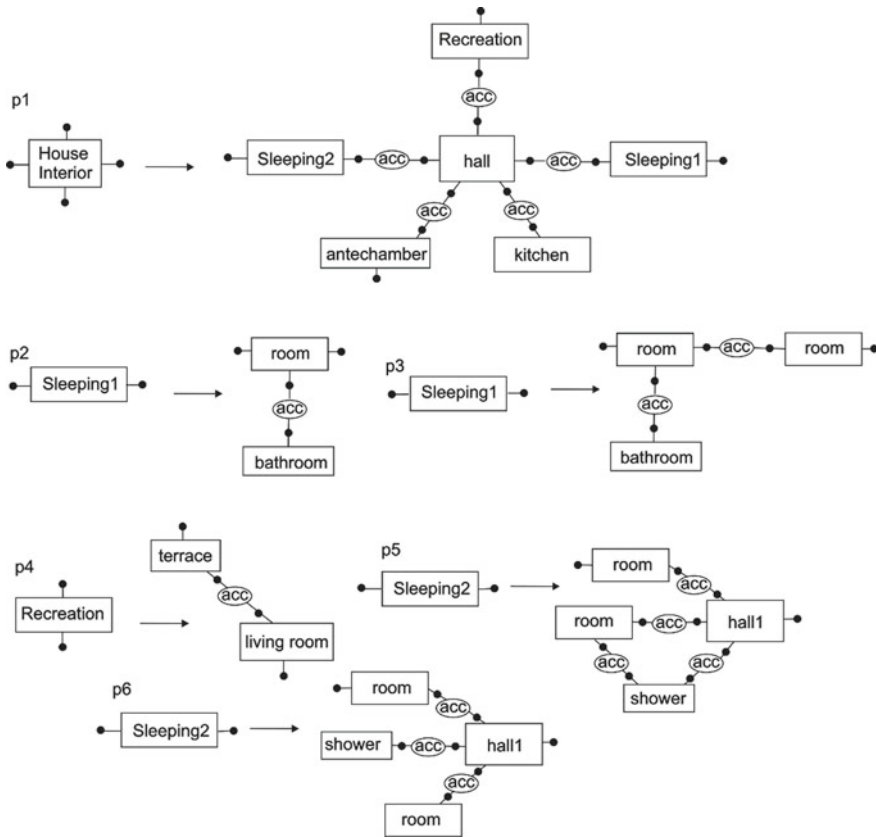
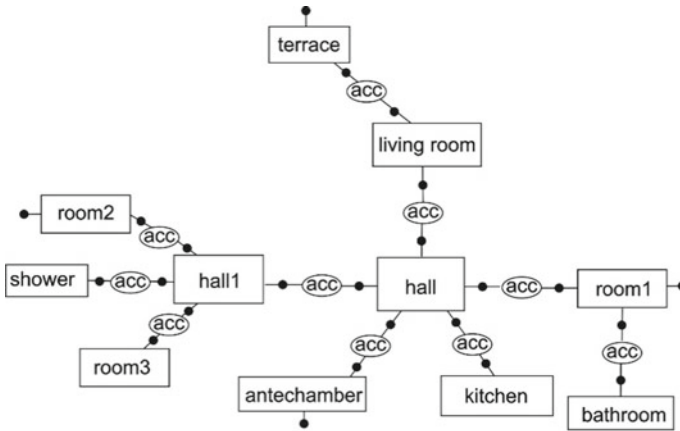


Fig. 12.21 Some rules of a hypergraph grammar generating structures of floor layouts

areas, can be specified. A hypergraph generated using this grammar is shown in Fig. 12.22a, while one of the possible floor layouts corresponding to this hypergraph is shown in Fig. 12.22b.

Example 14 Figure 12.23 presents one of the rules of a hypergraph grammar performing h-adaptation—dividing a triangular element into two smaller elements along the longest edge. The values of attributes of nodes and hyperedges of the right-hand side graph are the same as the values of attributes of the corresponding nodes and hyperedges of the left-hand side graph. The values of attributes of nodes and hyperedges of the right-hand side graph, which do not have the corresponding nodes and hyperedges in the left-hand side graph, are calculated. For the presented production, a so-called applicability predicate was defined. The predicate of applicability specifies conditions under which the rule can be used. The production from Fig. 12.23 can be applied only if the applicability predicate is fulfilled—the interior should be broken ($R1 == T$), if the edge is on the boundary of the mesh ($B1 == T$) and if the boundary edge is the longest one ($L1 \geq L2, L1 \geq L3$). Similar productions



(a)



(b)

Fig. 12.22 A hypergraph generated by the hypergraph grammar from Fig. 12.21 and its visualization

are defined for other cases—if the longest edge is not on the boundary of the mesh, etc. Figure 12.24 shows an example of a hypergraph and the same hypergraph after the application of the production from Fig. 12.23.

When structures of objects are described in terms of hierarchical hypergraphs, hierarchical hypergraph grammars serve as efficient tools for generating these structures. A hierarchical hypergraph grammar is composed of a set of hypergraph edges with terminal and non-terminal labels, a set of hypergraph nodes, a set of productions and an axiom being an initial hypergraph. Each grammar production is of the form p

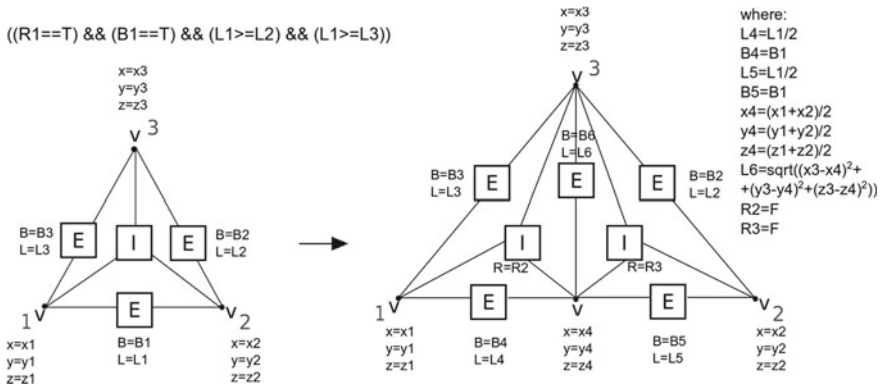


Fig. 12.23 One of the rules of a hypergraph grammar performing h-adaptation—dividing a triangular element into two smaller elements along the longest edge

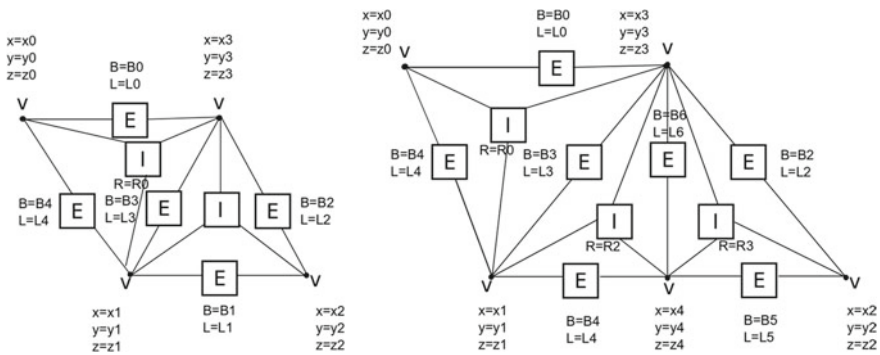


Fig. 12.24 An example of a starting hypergraph and the same hypergraph after the application of the production from Fig. 12.23

$= (l, r)$, where l and r are attributed hierarchical hypergraphs with the same number of external nodes equipped with ordering relations.

Let N and T denote sets of non-terminal and terminal labels, respectively. Let $A = V \cup E$ be a set of atoms of H , where H denotes a family of attributed hierarchical hypergraphs over $N \cup T$. Let lb be a hyperedge labelling function.

Definition 12 A hierarchical hypergraph grammar hG over N and T is a system $hG = (V, E, P, X)$, where:

1. V is a finite set of nodes,
2. $E = E_N \cup E_T$ is a finite set of hyperedges, with a child nesting partial function $ch: E \rightarrow P(A)$, where $lb: E_N \rightarrow N$ assigns non-terminal labels to hyperedges of E_N , while $lb: E_T \rightarrow T$ assigns terminal labels to hyperedges of E_T ,
3. P is a finite set of productions of the form $p = (l, r)$ satisfying the following conditions:

- l and r are attributed hierarchical hypergraphs composed of nodes of V and hyperedges of $E_N \cup E_T$ with the same number of ordered external nodes
 - l contains at least one hyperedge of E_N ,
4. X is an initial attributed hierarchical hypergraph containing at least one hyperedge of E_N and called an axiom of hG .

Example 15 Some productions of a hierarchical hypergraph grammar generating structures of floor layouts are presented in Fig. 12.25. The numbers assigned to the nodes of the left and right-hand side hypergraphs of productions indicate successive external nodes of these hypergraphs. One of the possible hierarchical hypergraphs obtained using this grammar is shown in Fig. 12.26a, while the corresponding floor layout is illustrated in Fig. 12.26b.

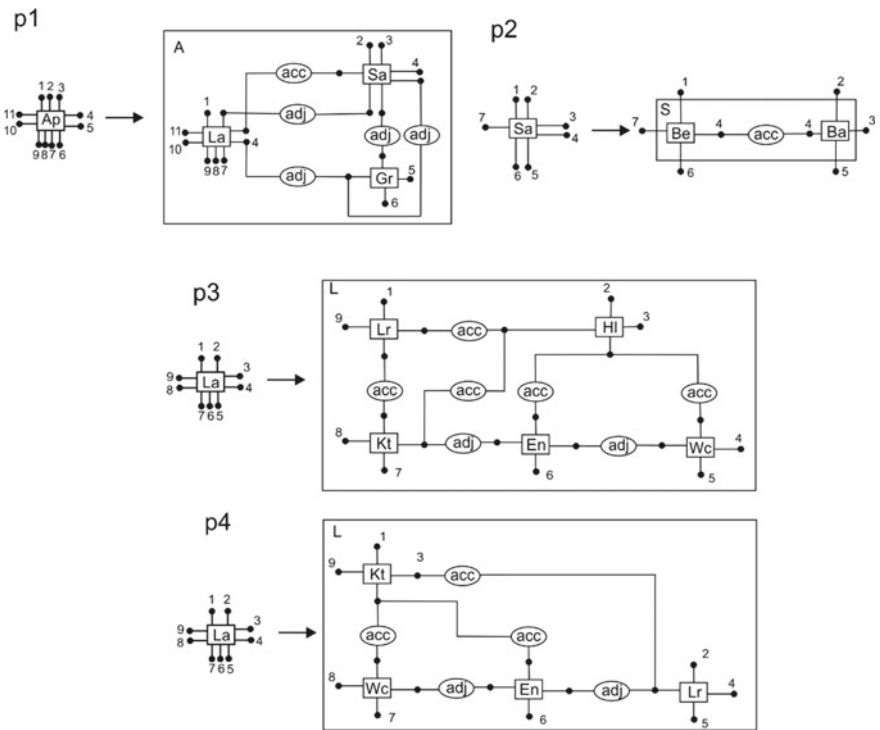
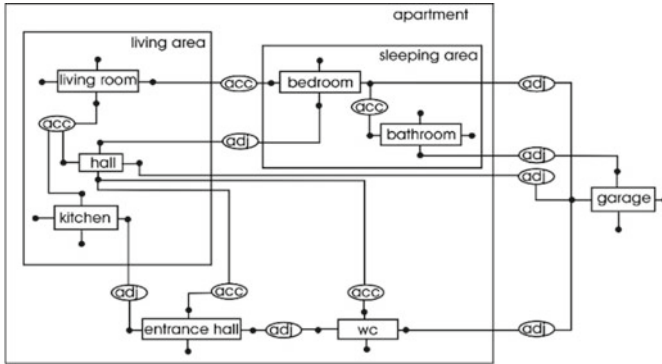
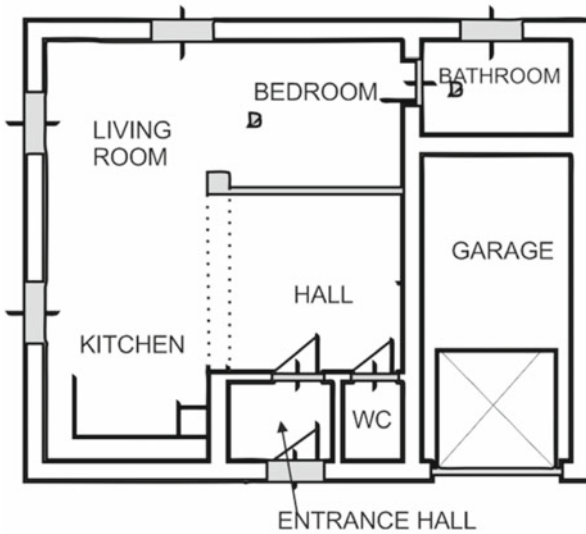


Fig. 12.25 Some productions of a hierarchical hypergraph grammar generating structures of floor layouts



(a)



(b)

Fig. 12.26 A hierarchical hypergraph representing a floor layout and its visualization

12.6 Other Types of Graphs

As in a hypergraph representation of modelled objects, hyperedges represent both object components and relations between them, this representation has been simplified to the form of a layout graph (*L-graph*) [50]. In a *layout graph*, graph nodes represent object components and are labelled by names of these components, while hyperedges represent multi-argument relations among them and are labelled by names of these relations. To each hyperedge of an *L-graph* a sequence of different target and

source nodes is assigned. Nodes and hyperedges of a layout graph are attributed. Attributes represent characteristic features of objects and relations.

In [50] *L*-graphs have been used to represent garden designs. In [52] the hierarchical layout graphs (*HL*-graphs) were used as the internal representation of knowledge about generated designs in the system supporting building design. In *HL*-graphs, hierarchical layout graphs can be nested in graph nodes. *HL*-graph hierarchy is obtained by nesting in selected nodes *HL*-graphs representing internal arrangements of design components corresponding to these nodes. Additionally, there is a possibility of specifying relations between components represented by nodes having different parent nodes and nested on different levels of hierarchy. Such a hierarchical structure allows for hiding low-level data that are unnecessary at the moment and present the designer only a detailed view of the selected object parts.

In [52] a hierarchical layout graph grammar, called an *HLG*-grammar, which is used to generate *HL*-graphs representing structures of designed objects, is defined.

In examples described in previous sections spatial relations between some elements are modelled using graph hierarchy. If the node representing a given design component is nested in some other graph node, it means that this component is physically a part of a super-component represented by that parent node. Computing grid models are the only exception: some types of grid components are immaterial software services, so for them being a child of some other graph node does not mean *is a part of* but *is running on*. In this case graph nodes still form a single hierarchy, but hierarchical dependency means something non-physical.

There are design tasks where there exist many different types of relations between design parts, which cannot be expressed with a single hierarchy. Usually different hierarchies are required for geometrical and functional dependencies. For example, university buildings have a spatial hierarchy which assigns rooms to floors, as well as an administrative hierarchy which assigns rooms to organizational units (departments, chairs, etc.). Therefore in [51] *multi-hierarchical graphs*, which have several independent hierarchies, specified by several nesting functions, have been proposed to model structure, functionality and organizational aspects of a building.

12.7 Conclusions

In this paper, several types of graphs used to model engineering, design, and computer science problems were described. In the presented approach, graphs were used as the representation of the knowledge related to the considered problem. Additionally, graph transformations were used to model the process of solving the problem. According to the needs of applications in various domains, different types of graph structures have been used. Therefore, the labelled, attributed, directed and undirected variants of standard graphs, composition graphs, hierarchical composition graphs, hypergraphs, hierarchical hypergraphs, layout, and hierarchical layout graphs, as well as multi-hierarchical graphs are considered. Depending on the type of graphs used, graph transformation rules also take different forms. The paper also describes

the application of the presented graph-based approach to model the design process of bridges, transmission towers, grids, floor layouts, and finite element mesh generation and adaptation process.

References

1. Barendregt, H., van Eekelen, M., Glauert, J., Kennaway, J., Plasmeijer, M., Sleep, M.: Term graph rewriting. *LNCS* **259**, 141–158 (1987)
2. Borkowski, A., Grabska, E., Hliniak, G.: Function-structure computer-aided design model. *Mach. Graph. Vis.* **8**, 367–381 (1999)
3. Borkowski, A., Szuba, J.: Graph transformation in architectural design. *Comput. Assisted Mech. Eng. Sci.* **3**, 109–119 (2001)
4. Borkowski, A., Grabska, E., Szuba, J.: On graph-based knowledge representation in design. *Comput. Civ. Eng.* 1–10 (2002)
5. Borkowski, A., Grabska, E., Nikodem, P., Strug, B.: Searching for innovative structural layouts by means of graph grammars and evolutionary optimization. In: *Proceedings of 2nd International Structural Engineering and Construction Conference, Rome, Italy*, pp. 475–480 (2003)
6. Bretto, A., Gillibert, L.: Hypergraph-based image representation. *LNCS* **3434**, 1–11 (2005)
7. Drewes, F., Hoffmann, B., Plump, D.: Hierarchical graph transformation. *J. Comput. Syst. Sci.* **64**, 249–283 (2002)
8. Eloy, S., Duarte, J.P.: A transformation grammar for housing rehabilitation. *Nexus Netw. J.* **13**, 49–71 (2011)
9. Flasiński, M.: Use of graph grammars for the description of mechanical parts. *Comput. Aided Des.* **27**, 403–433 (1995)
10. Göttler, H., Günther, J., Nieskens, G.: Use graph grammars to design CAD-systems. *LNCS* **532**, 396–409 (1990)
11. Grabska, E.: Graphs and designing. *LNCS* **776**, 188–203 (1993)
12. Grabska, E., Borkowski, A.: Assisting creativity by composite representation. In: *Artificial Intelligence in Design'96*. Kluwer Academic Publishers, Dordrecht, Netherlands, pp. 743–759 (1996)
13. Grabska, E., Hliniak, G.: Graphic prints design using graph grammars. *Mach. Graph. Vis.* **7**, 345–353 (1998)
14. Grabska, E., Palacz, W., Szyngiera, P.: Hierarchical graphs in creative design. *Mach. Graph. Vis.* **9**, 115–122 (2000)
15. Grabska, E., Ślusarczyk, G., Niewiadomska, A.: Hierarchical graph grammars in graphic prints design and generation. *Machine Graphics and Vision* **10**(4), 519–536 (2001)
16. Grabska, E., Ślusarczyk, G., Papiernik, K.: Interpretation of objects represented by hierarchical graphs. In: *Proceedings of 3rd Conference on Computer Recognition Systems, Wrocław, Poland*, pp. 287–293 (2003)
17. Grabska, E., Łachwa, A., Ślusarczyk, G., Grzesiak-Kopeć, K., Lembas, J.: Hierarchical layout hypergraph operations and diagrammatic reasoning. *Mach. Graph. Vis.* **16**, 23–38 (2007)
18. Grabska, E., Strug, B., Ślusarczyk, G., Grabski, W.: Grammar-based distributed design. In: *Computational Intelligence: Methods and Applications, EXIT, Warszawa*, pp. 493–502 (2008)
19. Grabska, E., Ślusarczyk, G.: Knowledge and reasoning in design systems. *Autom. Constr.* **22**, 927–934 (2011)
20. Grabska, E., Palacz, W., Strug, B., Ślusarczyk, G.: A graph-based generation of virtual grids. *LNCS* **7203**, 451–460 (2011)
21. Grabska, E., Łachwa, A., Ślusarczyk, G.: New visual languages supporting design of multi-storey buildings. *Adv. Eng. Inform.* **26**, 681–690 (2012)

22. Grabska, E., Ślusarczyk, G., Gajek, Sz.: Knowledge representation for human-computer interaction in a system supporting conceptual design. *Fundam. Inform.* **124**, 91–110 (2013)
23. Grabska-Gradzińska, I., Porębski, B., Palacz, W., Grabska, E.: Graph-based data structures of computer games. In: Proceedings of 6th Annual International Conference on Computer Games, Multimedia and Allied Technology (CGAT 2013)
24. Heitor, T.V., Duarte, J.P., Pinto, R.M.: Combining grammars and space syntax: formulating, generating and evaluating designs. *Int. J. Archit. Comput.* **2**, 492–515 (2004)
25. Hliniak, G., Strug, B.: Graph grammars and evolutionary methods in graphic design. *Mach. Graph. Vis.* **9**, 5–13 (2000)
26. Kneidl, A., Borrmann, A., Hartmann, D.: Generating sparse navigation graphs for microscopic pedestrian simulation models. In: Proceedings of 18th EG-ICE International Workshop, Twente, Netherlands (2011)
27. König, R., Schneider, S.: Hierarchical structuring of layout problems in an interactive evolutionary layout system. *AIEDAM: Artif. Intell. Eng. Des. Anal. Manuf.* **26**, 129–142 (2012)
28. Kraft, B., Meyer, O., Nagl, M.: Graph technology support for conceptual design in civil engineering. In: Proceedings of the International Workshop on EG-ICE, Darmstadt, Germany, pp. 1–35 (2002)
29. Minas, M.: Concepts and realization of a diagram editor generator based on hypergraph transformation. *Sci. Comput. Program.* **44**, 157–180 (2002)
30. Nowak, D., Palacz, W., Strug, B.: On using graph grammars and artificial evolution to simulate and visualize the growth process of plants. *Comput. Imaging Vis.* **32**, 668–673 (2006)
31. Palacz, W., Ryszka, I., Grabska, E.: Graphs with layers—a visual tool for conceptual design and graph generation. In: Proceedings of 21st EG-ICE International Workshop, Cardiff, UK (2014)
32. Palacz, W., Paszyńska, A., Świdowska, I., Ślusarczyk, G., Strug, B., Grabska, E.: GraphTool: a visual support for generating graph models of artefacts. In: Proceedings of The International Conference on Methods & Tools for CAE—Concepts and Applications, Bielsko-Biała, Poland, pp. 81–86 (2017)
33. Palacz, W., Paszyńska, A., Świdowska, I., Ślusarczyk, G., Strug, B., Grabska, E.: GraphTool: case studies. In: Proceedings of the International Conference on Methods & Tools for CAE—Concepts and Applications, Bielsko-Biała, Poland, pp. 75–80 (2017)
34. Paszyńska, A.: Graph-grammar greedy algorithm for reutilization of partial LU factorization over 3D tetrahedral grids. *J. Comput. Sci.* **18**, 143–152 (2017)
35. Paszyński, M., Schaefer, R.: Graph grammar-driven parallel partial differential equation solver. *Concurr. Comput.: Pract. Exp.* **22**, 1063–1097 (2010)
36. Qureshi, R.J., Ramel, J., Cardot, H.: Graph based shapes representation and recognition. *LNCS* **4538**, 49–60 (2007)
37. Ryszka, I., Grabska, E.: GraphTool—a new system of graph generation. In: Proceedings of 7th International Conference on Advanced Engineering Computing and Applications in Sciences, ADVCOMP, Porto, Portugal, pp. 79–83 (2013)
38. Rozenberg, G.: Handbook of Graph Grammars and Computing by Graph Transformations, vol. 2: Applications, Languages and Tools. World Scientific, London (1999)
39. Schürr, A., Winter, A., Zündorf, A.: Graph grammar engineering with PROGRES. *LNCS* **989**, 219–234 (1995)
40. Strug, B., Ryszka, I., Grabska, E., Ślusarczyk, G.: Generating a virtual computational grid by graph transformations. In: Remote Instrumentation for eScience and Related Aspects, Springer, pp. 209–226 (2012)
41. Strug, B., Paszyńska, A., Paszyński, M., Grabska, E.: Using a graph grammar system in the finite element method. *Int. J. Appl. Math. Comput. Sci.* **23**, 839–853 (2013)
42. Strug, B.: Graph-based knowledge representation in computer-aided design: models and their applications. AGH Monograph (2014)
43. Strug, B., Grabska, E., Ślusarczyk, G.: Supporting the design process with hypergraph genetic operators. *Adv. Inform.* **28**, 11–27 (2014)

44. Strug, B., Ślusarczyk, G., Grabska, E.: A graph-based generative method for supporting bridge design. In: Proceedings of 24th EG-ICE international workshop, Nottingham, UK, pp. 294–302 (2017)
45. Szymczak, A., Paszyńska, A., Paszyński, M., Pardo, D.: Preventing deadlock during anisotropic 2D mesh adaptation in hp-adaptive FEM. *J. Comput. Sci.* **4**, 170–179 (2013)
46. Ślusarczyk, G.: Hierarchical hypergraph transformations in engineering design. *J. Appl. Comput. Sci.* **11**, 67–82 (2003)
47. Ślusarczyk, G.: Heuristic methods and hierarchical graph grammars in design. In: Proceedings of Visual and Spatial Reasoning in Design III, Key Centre of Design Computing and Cognition, University of Sydney, pp. 45–66 (2004)
48. Ślusarczyk, G.: Visual language and graph-based structures in conceptual design. *Adv. Eng. Inform.* **26**, 267–279 (2012)
49. Ślusarczyk, G., Paszyńska, A.: Hypergraph grammars in hp-adaptive finite element method. *Procedia Comput. Sci.* **18**, 1545–1554 (2012)
50. Ślusarczyk, G., Piętak, P.: Maintaining style of garden designs by using graph-based constraints. *Autom. Constr.* **36**, 79–94 (2013)
51. Ślusarczyk, G., Łachwa, A., Palacz, W., Strug, B., Paszyńska, A., Grabska, E.: An extended hierarchical graph-based building model for design and engineering problems. *Autom. Constr.* **74**, 95–102 (2017)
52. Ślusarczyk, G.: Graph-based representation of design properties in creating building floorplans. *Comput. Aided Des.* **95**, 24–39 (2018)
53. Vilgertshofer, S., Borrmann, A.: A graph transformation based method for the semi-automatic generation of parametric models of shield tunnels. In: Proceedings of 23rd EG-ICE International Workshop, Kraków, Poland (2016)

Chapter 13

On the Problem of Visualization of Big Graphs for Infrastructure Engineering



V. Martsenyuk, M. Karpinski, S. Zawiślak, A. Vlasyuk, A. Shaikhanova,
and O. Martsenyuk

Abstract The work is devoted to the development of an approach allowing us to visualize complex infrastructure networks. We offer the way of construction of the graph based on the formal description of the infrastructure on macro- and micro-topology. The results of testing various layout algorithms on the plane of the graph representing the complex infrastructure network are presented. The method of automatic graph layout is offered, which allows to efficiently perform the layout of a graph of a large-scale infrastructure network. The method of clustering the graph of the infrastructure network, taking into account the strength of the branches and the number of clusters unknown in advance, is proposed. The method is implemented in R language with help of igraph package for visualization. Different layout schemes of nodes are compared: circle, star, tree, grid, force-directed. Using in hierarchical clustering, we illustrate the arrangement of the clusters produced by the corresponding analysis. An example of graph layout for the topology of the Western States Power Grid of the United States for 4990 nodes is considered.

V. Martsenyuk (✉) · M. Karpinski · S. Zawiślak
University of Bielsko-Biała, Bielsko-Biała, Poland
e-mail: vmartsenyuk@ath.bielsko.pl

M. Karpinski
e-mail: mkarpinski@ath.bielsko.pl

S. Zawiślak
e-mail: szawislak@ath.bielsko.pl

A. Vlasyuk
The National University of Ostroh Academy, Ostroh, Ukraine
e-mail: anatolij.vlasyuk@oa.edu.ua

A. Shaikhanova
Shakarim State University of Semey, Semey, Kazakhstan

O. Martsenyuk
Ternopil City Council, Ternopil, Ukraine
e-mail: marceniuk@ukr.net

Keywords Layout of the graph on the plane · Visualization of the infrastructure network · Clustering of the infrastructure network · R · igraph

13.1 Introduction

One of the key factors in the socio-economic development of society is the creation, maintenance and development of infrastructure, which provides and supports various processes of human activity. Various types of infrastructures are distinguished and investigated—social, transport, financial, information, etc.

Currently, the problem of optimal design of complex geographically distributed systems of engineering infrastructure is one of the most urgent in the construction industry. This problem includes the formulation, formalization and typification of design solutions, procedures and design processes in the integrated technology of GIS (geographic information systems) and CAD (computer-aided design systems). An important aspect in the creation of an integrated GIS and CAD technology based on a common mathematical and informational basis, is their versatility. However, at present, the creation of universal integrated GIS and CAD is difficult, since specialized modeling and calculation programs are not sufficiently open and their integration based on modern principles and technologies is impossible. On the other hand, the creation of such a technology requires a sufficiently universal mathematical apparatus that will allow at the design stage to take into account the spatial location of various engineering networks, their multilevel and interaction.

A graph is a fundamental data structure that allows you to model a variety of objects and the relationships between them. With an increase in the amount of information, the role of graphs in science is difficult to overestimate: they are used in physics to model interactions between bodies, in chemistry to establish bonds between molecules and atoms; biologists use graphs to model the behavior of bacteria, sociologists study relationships in society and identify patterns of behavior, in programming, graphs are the central data structure.

The visualization of graphs has been actively developing since the early 1980s, as it is a convenient tool for analyzing complex data. Confirmation of the relevance of the topic is the holding of annual international symposia and conferences (the largest ones are InfoVis and Graph Drawing). The task of visualization is to present the graph (draw, build styling) in a simple and understandable way, emphasizing its structure and topological properties.

Research in the field of management of infrastructure network models are of great importance. In the course of solving this problem, the need arose for the automatic representation of the infrastructure network as a graph reflecting not only the topological state of the network, but also the distance of the network elements from each other.

As it turned out, this task has its own value and can be used for the following purposes:

- visualization of an unknown infrastructure network, when there is data only about the network model, but not its graphical representation;
- network clustering for emergency control purposes;
- finding trajectories for the problem of analyzing the static stability.

The concept of “hypernetwork” was first introduced in [18] as a mathematical model for describing complex network structures, primarily communication networks. Hypernetworks, including hierarchical, non-stationary and structured ones, in contrast to graphs, describe systems that have more than two generating sets, taking into account the implementation of one structure in another. In the present work, in contrast to [17, 18], hypernetworks are used to optimize the design process of the network infrastructure of utility networks, which has a multi-level structure.

It should be noted that the use of the hypernetwork model as an alternative to the existing methods and technologies for creating GIS expands the functionality of the designed GIS by using the capabilities of the hypernetwork approach for the analysis and organization of multi-level systems.

The graph layouts considered as part of the work are mainly used to visualize various relationships, such as social networks, hierarchical representation of biological organisms, computer network structures, etc. The paper presents the method of application of graph layout algorithms for the automatic representation of a large-scale infrastructure network, which takes into account the infrastructural connectivity of network nodes. Based on the graph layout performed, its clustering is performed using the proposed modified k-means algorithm in the form of tree-like presentation.

Graph visualization algorithms have been developed since the beginning of the 1960s [1]. The area of graph drawing began to develop most actively with the appearance of methods based on physical analogies. The classic works are the Eades spring algorithm [2] and the Kamada and Kawai algorithm [3]. The forced model was proposed by Fruchterman and Reingold [4]. Later, numerous modifications and optimizations of these algorithms were offered [5, 6].

The task of large-scale graph visualization has been actively studied in the last years, when processing of large amounts of information became available. Most of the existing algorithms are designed to draw a limited class of graphs or not applicable for large-scale graphs [7]. The multidimensional scaling method has been used to visualize graphs since 1980 [8]. It was widely used after the work of Gansner et al. [9], and now it is one of the most popular methods for drawing graphs [10].

There are several ways to layout graphs on a plane. The most popular methods are based on the physical analogies of forced (force-directed) and spring methods. For constructing layouts, a special model is constructed in which the vertices and edges of the graph correspond to real physical interacting objects. An energy function is introduced for this system so that configurations with a lower energy level correspond to better layout. In this case, the task of finding the best layout of the graph is reduced to finding the minimum energy of the system. However, such methods have a number of limitations, and in the classical formulation are not applicable to drawing graphs of the complex infrastructure based on the big data.

Here we present modifications of these methods to layout the graphs of complex infrastructure systems. Summarizing, the contribution of our work is as follows: the forced algorithm is formulated in a generalized form in which it can be applied to a wider class of problems, including drawing weighted and disconnected graphs for the big infrastructure networks; spring method (multidimensional scaling method) is adapted for the big infrastructure networks; a prototype system was created for interactive construction and visualization of layouts using the described algorithms and the *igraph* package; visualizations of several complex infrastructure networks were built, an analysis of the changes and development of these networks was carried out.

13.2 Formal Description of the Infrastructure Data

The notion of infrastructure considered in this work in a formalized form is an identified as a set of points and their connecting edges that ensure the delivery of something between points, as well as a set of supporting subsystems (power, water, transport, phone networks, etc.).

Infrastructure, being the most important driver of the effectiveness of the national economy, at the same time requires enormous costs for maintenance, repair and development. In this context, it is extremely important to increase the efficiency of managing the processes of maintaining and using infrastructure based on modern modeling methods and information technologies. A formalized description of the infrastructure is used to solve a wide range of management tasks. In doing so, various tasks require different types of data.

1. *Factographic data* on infrastructure (description of infrastructure facilities, data on their condition, work performed at these facilities, etc.). These data are stored in centralized databases and are used to solve the problems of maintaining and maintaining infrastructure, managing the transportation process, etc.
2. *Topological data* on the infrastructure (graph-scheme of the network, a schematic plan of the nodes, the scheme of the contact network, etc.). These data can be stored both in databases and in local files and are used in simulation and optimization problems and visualization.
3. *Geo-data* on infrastructure (geographic coordinates of infrastructure objects) The indicated data can be stored both in databases and in local files and used in tasks of GIS visualization, GIS analysis, etc.).

Traditionally, different types of data about objects are stored in various local systems and are not interconnected, which excludes their sharing, and also dramatically reduces the quality of data due to duplication and inconsistency. The organization of a unified information space for the infrastructure on the basis of a unified identification of infrastructure objects and integrated storage and maintenance of factual, topological and geo-data on objects is solved by creating an integrated model of the infrastructure.

Currently, models of the infrastructure are deprived of universality, their structure is dictated solely by a list of tasks to be solved. These models can be divided into two groups—detailed and enlarged, microtopological and macrotopological.

1. Macrotopology is an enlarged description of infrastructure objects. At the macro level, only key objects are reflected—stations, stages, large bridges, tunnels, etc. The macro level covers large structural elements—roads, directions or a network of roads.
2. Microtopology—a detailed description of infrastructure within the boundaries of relatively small structural elements, such as stations. At this level, a large list of elements is reflected - paths, turnouts, level crossings, signals, stops, etc. The implementation of only one scale within the framework of an integrated model of the infrastructure significantly limits the capabilities of the system. For example, a serious limitation of the level of macrotopology in modeling and infographic problems is the lack of such an infrastructure element as the path. The node on this scale is a point; there is no track development. Thus, when using the topology of one level, it is impossible to solve the problems of modeling and optimization of infrastructure work, visualization of indicators related to the path.

For the reasons given we conclude that the approach of hypernetwork fits the problems of infrastructure engineering since it corresponds to a multilevel hierarchy of the big data used.

13.3 Mathematical Description and the Solution Methods

We give the necessary definitions. *Graph* G consists of the set of vertices V and the set of edges $E \subset V \times V$. The number of vertices in the graph is denoted by $n = \#(V)$, the number of edges $m = \#(E)$. The vertices of the graph $V = \{v_1, v_2, \dots, v_n\}$ can also be denoted by the indices $V = \{1, 2, \dots, n\}$. For the vertex v , its weight w_v is introduced; similarly, for the edge (u, v) , the weight is denoted by w_{uv} . Here we consider ordinary (undirected, without loops and multiple edges) graphs.

Taking into account the multilevel process of designing engineering networks, due to the *nesting* of networks, can be done by dividing into primary and secondary networks - just like in communication networks. In this case, the topology of the primary network is described by the graph $G_{PN} = (X, V; P)$, in which the vertices X correspond to the nodes of the existing network, and the edges V correspond to the branches.

Each edge V of the primary network PN is represented in the form of traces for laying lines of the developed network. The secondary network $G_{SN} = (Y \subseteq X, R; W)$ corresponds to the structure of the developed network, in which the sets of vertices Y are divided into two disjoint subsets, that is, $Y = I \cup J, I \cap J = \emptyset$. The first subset $I = \{1, \dots, m\}$ contains the vertices corresponding to the points where the sources of the target production can be placed. The second subset $J = \{1, \dots, n\}$ contains the vertices corresponding to consumers. The edges R of the graph of the

secondary network SN correspond to physical lines (pipes, cables, etc.) that directly move the target product from the source to the consumers. The interaction of these subsystems is determined by a two-level hypernet [18].

The hypernet $G_{HN} = (X, V, R, P, F, W)$ includes the following objects:

- $X = (x_1, x_2, \dots, x_n)$ is a set of nodes (vertices);
- $V = (v_1, v_2, \dots, v_g)$ is a set of traces;
- $R = (r_1, r_2, \dots, r_m)$ is a set of physical lines (pipes, cables, etc.);
- P is a mapping assigning to each element $v \in V$ the set of vertices $P(v) \subseteq X$.

Thus, the mapping P defines the graph of the primary network $G_{PN} = (X, V; P)$;

- F is a mapping that assigns to each element $r \in R$ the set of traces $F(r)$ forming a simple route in the graph $G_{PN} = (X, V; P)$.

The mapping F defines a hypergraph $G_{FN} = (V, R; F)$. The set of all routes $F(r)$ that maps to each edge $r \in R$ of the graph G_{SN} a unique route in the graph G_{PN} is called an *embedding* of the graph G_{SN} into G_{PN} .

- W is a mapping assigning to each element $r \in R$ the set of vertices $W(r) \subseteq P(F(r))$ in the graph G_{PN} , where $P(F(r)) = Y$ is a set of vertices in G_{PN} incident to traces $F(r) \subseteq V$. The mapping W defines the secondary network graph $G_{SN} = (Y, R; W)$.

When considering models of infrastructure hypernets the problem of graph layout is of importance, especially taking into account big data of infrastructure topology. The *layout* L of the graph $G = (V, E)$ is called the map $L : V \rightarrow R^k$ of the vertices of the graph into the set of points in the k -dimensional space. To *draw* a graph means to indicate the coordinates of the vertices and to draw straight lines between the vertices. The position of the vertex v_i will be denoted by x_{v_i} or x_i , its coordinates are $x_i^1, x_i^2, \dots, x_i^k$. The *distance* between the vertices v and u is called the Euclidean distance between the corresponding points, namely, $|x_v - x_u|$.

Any infrastructural network can be represented as a weighted graph. In case of power network, the graph node is the bus of a power station or substation. The edge of the graph is a branch (element) of an electrical circuit with resistance.

For each edge of the graph, its weight is set equal to or proportional to the impedance of the corresponding branch. Each node of the graph has coordinates in the space of the graph. The coordinates of the space of the graph are given in the same units as the weights of the edges of the graph. Thus, the coordinates of the nodes of the graph determine the electrical distance of the electrical substations from each other.

It is required to arrange the nodes of the graph so that they form spatial clusters in such a way that within the cluster there are a lot of mutual or strong connections, and long lines form connections between the clusters. For the layout of the graph on the plane were analyzed various algorithms. As a test example, a sufficiently large power network of the real power system was chosen.

Since it is important to take into account the distance of network nodes when building a network graph, algorithms based on forces (force directed) were investigated [11, 12]. Such algorithms for the layout of an undirected graph on a plane are based on representing the vertices of the graph as repulsive particles, and the faces of the graph as springs tightening the corresponding nodes. For an infrastructural network, the force of knotting a node by the edge is proportional to the conductivity of the branch. Finding a balance through the optimization process, we obtain a solution in which strongly connected parts of the network are concentrated into clusters.

The minimization problem can be written in two ways:

- forced algorithm:

$$\min_{x_i} \sum_{(i,j) \in E} f_a w_{ij} |x_i - x_j|^a + \sum_{i \neq j} f_r w_i w_j |x_i - x_j|^r \quad (13.1)$$

where $a, r, f_a, f_r \in R$;

- spring-like algorithm:

$$\min_{x_i} \sum_{1 \leq i < j \leq n} w_{ij} (|x_i - x_j| - l_{ij})^2 \quad (13.2)$$

where n is the number of vertices of the graph; w_{ij} is the weight of the edge, which determines the force of contraction of the corresponding vertices; l_{ij} is optimal edge length.

Solving the last optimization problem using the gradient descent method, it is quite easy to get to the local minimum. In this case, the layout of the graph can be far from optimal. In addition, when moving only one vertex at each iteration, computational complexity cannot be expressed through the number of vertices and edges. The system can go into an oscillatory mode and not converge to the optimum.

To exclude local minima, an annealing simulation algorithm is used. The application of the plan to change the temperature proposed in the works of Fruchterman and Reinhold [13], allows to solve this problem in a reasonable time. However, for a graph of a sufficiently large electrical network, this algorithm gives unsatisfactory results.

As it can be seen, the resulting graph does not reflect the distance of the network nodes, depending on their mutual conductivity. The GEM algorithm proposed by Frick et al. [14], using the concept of local temperature, the attraction of vertices to their center of gravity and the detection of vibrations, quickly converges, but without taking into account the resistances of the branches in the forces of attraction, the graph layout is also unrealistic does not reflect the electrical distance of network nodes). The algorithms Force Atlas and Force Atlas 2 [15], implemented in the Gephi package, provide the most adequate results for the representation of the power network. These algorithms are also based on a model of repulsive particles (vertices) connected by springs (edges).

If we take the formulas for the forces of attraction and repulsion as in their physical equivalent (springs and charges), then the force of attraction F_a between the vertices will be determined by the formula $F_a = -kd$, where d is the length of the face connecting the vertices, and the repulsive force $F_r = k/d^2$, where d is the distance between the vertices. The coefficient k is determined depending on the size of the graph.

For the algorithm of Fruchterman and Reinhold, the forces of attraction and repulsion are equal:

$$F_a = d^2/k; \quad F_r = -k^2/d. \quad (13.3)$$

In the ForceAtlas2 algorithm, the forces of attraction and repulsion are equal to:

$$F_a = d/k; \quad F_r = -k^2/d. \quad (13.4)$$

As it was shown earlier, the degree of consideration for the distance between vertices has the greatest impact on algorithms based on the use of forces.

If you define the degree in the formulas of distance, as a and r :

$$F_a = d/k; \quad F_r = -k^2/d^r, \quad (13.5)$$

then visually the clusters become more pronounced with smaller values $a - r$.

For the graph representing the infrastructural network, the optimal value is

$$a - r = 2 \quad (13.6)$$

In addition, it is important to consider the weight of the bond determined by the conductivity of the branch:

$$F_a = w(e) \cdot d = k \cdot Y \cdot d, \quad (13.7)$$

where Y is the branch conductivity; k is the normalization factor (for the considered network it is 1000).

As in the Force Atlas 2 algorithm, the repulsive force is modified in such a way that it decreases for the strongly connected nodes, and for the most weakly connected. As a result, there are separate distant nodes. To correct this deficiency, after working on the Force Atlas 2 algorithm, we pull up loosely coupled nodes and freeze the rest of the graph. The layout of the graph of the power network after pulling up the weak links is shown in Fig. 13.1



Fig. 13.1 Changing vertices size and appearance in dependence of “strength” of vertices

13.4 The Simulation by Means of igraph

The data for numerical experiment comes from [16]. Namely, they describe power grid of the western states of the USA. The topology is an undirected, unweighted graph, which consists of 4941 vertices, representing power objects, and 6594 edges, connecting them. GML model for representing graph is used. For the research we apply RStudio IDE and igraph package, which offers the implementation of a set of graph-based algorithms.

Different techniques of the node layout were used. We started from the initial modification of appearance of the nodes (i.e., size, labels, colors) (Fig. 13.1).

Different layout techniques (random, circle, star, tree, grid, force-directed) are presented in Fig. 13.2. We pay attention that the force-directed technique is non-deterministic. It means that we get different layout nodes depending on the initial seed values (Fig. 13.3). Special attention should be paid to hierarchical clustering of the network. Namely, we can illustrate the arrangement of the clusters produced by the corresponding analysis (k-means technique) with help of tree layout in Fig. 13.2.

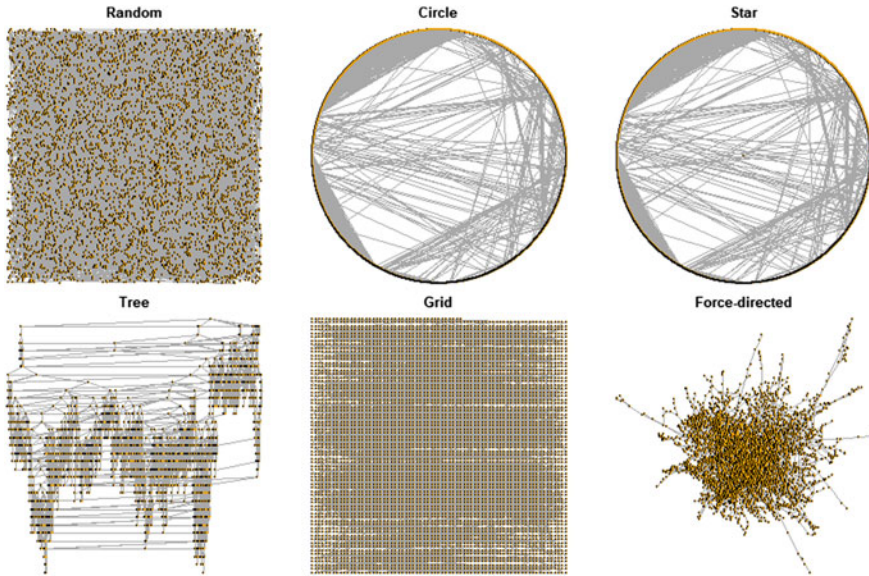


Fig. 13.2 Comparison of application of different layout techniques for the power network: random, circle, star, tree, grid, force-directed

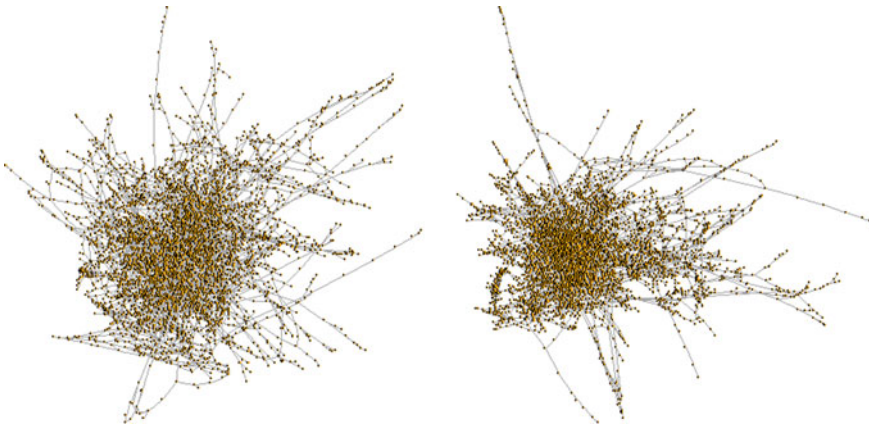


Fig. 13.3 Application of force-directed algorithm for the data of power network. Results for different seeds: seed = 777 (on left), seed = 666 (on right)

13.5 Conclusions

The developed method of layout on the plane of the graph of the infrastructural network was tested by the authors on various real infrastructural system schemes (see numerical example).

The presented methodology for graph visualization can be used when carrying out engineering calculations of infrastructural system by design and engineering organizations, when there is only an infrastructure model and its graphical representation is required for analyzing the calculation results.

The technique can be used for clustering of infrastructural hypernets of a large dimension (more than 1000 nodes).

The clustering of an infrastructural scheme using this technique allows you to quickly identify weak links in the network. At the same time, it should be noted that the use of graph layout algorithms based on modeling of attracting and repelling particles does not preclude situations where the more distant vertices, taking into account the topology, fall into one cluster. This solution can serve as a good initial approximation for the search for trajectories of weighting in the problem of analysis of the stability of the infrastructural system.

It is known that digital terrain models (DTM) are used as a mathematical basis for creating GIS and CAD systems. The most common method for constructing a DTM is the transition to a discrete analogue of the terrain based on the grid technology. Further, on the basis of existing optimization methods, paths are found on the map grid for the optimal spatial placement of engineering networks for various purposes.

To solve design problems for the organization and management of utility networks for various purposes, the methods of graph theory are mainly used. At the same time, the design results at one of the levels are not used as input data for subsequent levels. Thus, the design process does not take into account the relationships and interdependencies between the levels. In other words, the graphs are used for each of the levels separately. As a result, the proposed configuration of the designed network may not be optimal, and in some cases during operation may lead to unstable operation of the designed network. In this regard, in the work [17], as a mathematical basis for creating a specialized GIS, a hyper-network technology for optimizing the construction of the network infrastructure of modern cities was proposed.

Acknowledgements The work was co-funded by the European Union's Erasmus + Programme for Education under KA2 grant (project no. 2020-1-PL01-KA203-082197 "Innovations for Big Data in a Real World").

The authors are thankful to the reviewers for the valuable remarks that allowed to improve the article significantly.

References

1. Tutte, W.T.: How to draw a graph. *Proc. Lond. Math. Soc.* **13**, 743–768 (1963)
2. Eades, P.: A heuristic for graph drawing. *Congr. Numer.* **42**, 149–160 (1984)
3. Kamada, T., Kawai, S.: An algorithm for drawing general undirected graphs. *Inform. Process. Lett.* **31**, 7–15 (1989)
4. Fruchterman, T., Reingold, E.: Graph drawing by force-directed placement. *Softw. Pract. Exp.* **21**(11), 1129–1164 (1991)
5. Harel, D., Koren, Y.: A fast multi-scale method for drawing large graphs. *J. Graph Algorithms Appl.* **6**(3), 282–285 (2002)

6. Noack, A.: An energy model for visual graph clustering. In: Proceedings 11th International Symposium on Graph Drawing, pp. 425–436 (2003)
7. Diehl, S., Gorg, C.: Graphs, they are changing. In: Proceedings 10th International Symposium on Graph Drawing, pp. 23–30 (2002)
8. Kruskal, J.B., Seery, J.B.: Designing network diagrams. In: Proceedings of the First General Conference on Social Graphics, pp. 22–50 (1980)
9. Gansner, E.R., Koren, Y., North, S.C.: Graph drawing by stress majorization. In: Proceedings 12th Int. Symposium on Graph Drawing, pp. 239–250 (2004)
10. Pich, C.: Applications of multidimensional scaling to graph drawing. Ph.D. thesis, Universitaet Konstanz, Germany (2009)
11. Egorov, D., Bezgodov, A.: Improved force-directed method of graph layout generation with adaptive step length. *Procedia Computer Sci.* **66**, 689–696 (2015). <https://doi.org/10.1016/j.procs.2015.11.078>
12. Dogrusoz, U., Giral, E., Cetintas, A., Civril, A., Demir, E.: A layout algorithm for undirected compound graphs. *Inf. Sci.* **179**(7), 980–994 (2009). <https://doi.org/10.1016/j.ins.2008.11.017>
13. Fruchterman, T.M.J., Reingold, E.M.: Graph drawing by force-directed placement. *Software Practice Experience* **21**(11), 1129–1164 (1991)
14. Frick A., Ludwig A., Mehdau H.: A fast adaptive layout algorithm for undirected graphs (extended abstract and system demonstration). In: International Symposium on Graph Drawing. Springer, Berlin, pp. 388–403 (1994)
15. Kobourov, S.G.: Force-directed drawing algorithms. In: Handbook of Graph Drawing and Visualization, pp. 383–408. CRC Press, Boca Raton (2013)
16. Watts, D.J., Strogatz, S.H.: Collective dynamics of ‘small-world’ networks. *Nature* **393**(6684), 440–442 (1998)
17. Toktoshov, G.Y.: On creation of geographic information systems based on hypernetworks for the organization of the engineering infrastructure of modern cities. *Bull. Tomsk Polytechnic Univ. Geo Assets Eng.* **327**(1) (2016)
18. Popkov, V.K.: Hypernetworks and structured models of complex systems. In: Mathematical and Imitational Models of Complex Systems. Systemic Modeling, Novosibirsk, pp. 26–48 (1981)

Part IV
Miscellaneous

Chapter 14

Professor Józef Wojnarowski—IFTToMMist and Pioneer of Graphs' Application in Mechanics in Poland



S. Zawiślak and J. Kopeć

Abstract The chapter is dedicated to the achievements of Professor Józef Wojnarowski, polish IFTToMM activist, who had originated graph-based modelling of mechanical systems in Poland in the 70's of the twentieth century. He simultaneously, attracted a group of co-workers as well as other Polish scientists to this idea. All together they wrote a bunch of papers and a few books related to graph theory application. He also initiated and organized two international conferences “Graphs and Mechanics”. So, He could be recognized as a very important scientists whose output is fully in accordance with the goals of the present book and the IFTToMM mission.

Keywords Pioneer · IFTToMM activist · Versatile types of graphs · Different mechanisms and machines

14.1 Introduction

As usual, the title of the chapter should not be too long, but should be relevant to the content. What does it communicate? Graph theory is a branch of discrete mathematics. On the other hand, discrete mathematics is a field of mathematics where the discrete objects are considered, where additionally, lack of continuity is an immanent characteristic. On contrary, continuity is one of the fundamental notions for other fields of mathematics as e.g. analysis (including derivatives and integrals) as well as differential and integral equations.

S. Zawiślak (✉) · J. Kopeć
University of Bielsko-Biala, Bielsko-Biala, Poland
e-mail: szawislak@ath.bielsko.pl

J. Kopeć
e-mail: jkopec@ath.bielsko.pl

Fig. 14.1 Professor Józef Wojnarowski (*Resource own family collection*)



Graph theory has a wide spectrum of application, however its utilization for modeling of mechanical systems is relatively less known.

Professor Józef Wojnarowski (Fig. 14.1) is an outstanding Polish emeritus Professor who still works in the field of mechanics and simultaneously who is an indefatigable activist of IFToMM since its origins [1] and other scientific societies in Poland as well as all over the world. He was a supervisor of the J. Kopeć's doctor dissertation and a supervisor of the master thesis of S. Zawiślak—the authors of the present chapter. So, they can be considered as His scientific sons and—among other goals—they would like to pay a tribute to their mentor.

Professor Józef Wojnarowski started application of graphs for modeling of mechanical systems in Poland via his conference publication from 1971 [2], and then by a journal paper from 1973 [3]. Therefore we can just celebrate 50-th anniversary of this direction of investigations in Poland! Simultaneously and independently, Professor Krzysztof Arczewski (Warsaw TU) had been working on this topic—publishing the paper in 1972 [4] and defending the adequate doctor thesis in 1974 [5]. The important difference between these two scientist—in relations to graphs' application—consists in fact that K. Arczewski published the related papers, almost in every case, as a single author [6–12]. On contrary, J. Wojnarowski have involved, inspired and attracted a numerous group of co-workers creating a so-called 'science school' in Gliwice and a science sub-school in Bielsko-Biała. For many years, He had been working at the Silesian University of Technology in Gliwice. He was a

supervisor of approx. 20 doctorates from which many dissertations were dedicated to graph theory modeling of versatile mechanical systems applying different types of graphs and hypergraphs. However, the authors of the present chapter represent the mentioned science sub-school in Bielsko-Biała. Professor worked simultaneously in Bielsko-Biała University for 20 years.

Moreover, up until now in Poland, there is an *upper scientific title* i.e. Doctor of Science (D.Sc.). Several co-workers of Józef Wojnarowski received this title based on the habilitation theses dedicated to graph theory usage and application e.g. Andrzej Nowak, Jerzy Świder [13], Andrzej Buchacz, Jerzy Margielewicz [14] and Stanisław Zawiślak [15]. Professor Józef Wojnarowski organized in Gliwice two International Conferences entitled “Graphs and Mechanics” in 1993 and 1999, respectively. Both mentioned pioneering researches took part in these two conferences as well as all Poles involved in applied graph investigations e.g. [16]. Many representatives from abroad e.g. Canada, Germany, Hungary, Iran, Romania, U.K. etc. presented their talks. We can conclude that He would like to integrate the whole group of scientist interested in these problems. The conferences were successful, moreover the chosen, high quality papers were then published [17–21] in the MMT journal (Machine and Mechanism Theory) edited under patronage of IFTToMM. To sum up, we can stated that the “Polish School” of graphs’ application can be even observed due to activities of the Hero of the chapter.

Versatile types of graphs were utilized by Him and His team e.g. plain, directed, signal flow as well as bond-graphs however application of hypergraphs was unique and remains till today.

Like it was mentioned, the field of modeling of mechanical systems by means of graphs sometimes (for a moment) seems difficult to understand because of the fact that the real physical objects are continuous! The mystery consists in some mental activities, among which discretization is the essential one. Discretization is a decomposition of an object into some discrete particles/pieces/nodes etc. In Wikipedia, we can read the following phrase: “**discretization** is the process of transferring continuous functions, models, variables, and equations into discrete counterparts”. It is the background for such commonly known numerical methods as: FEM (finite element method), BEM (boundary element method) as well as (obviously) numerical integration. So, in case of the initially mentioned methods, a considered object is covered by the nodes (called sometimes a cloud of nodes) and a mesh which connects these nodes. If we consider these meshing on a plane; then we can consider nodes as graph vertices and the connections as graph edges. Methods like FEM and BEM are world-widely accepted by engineers, therefore the graph-based modeling can be recognized as obviously possible and acceptable. Józef Wojnarowski’s output has shown that the discussed graph-based approaches are reliable, effective, correct and useful. Once more, the discretization is the first step, however the problem arises how can we derive formulas and/or methods related to the mechanical field. Some detailed explanation will be given in the present chapter.

Moreover, in the chapter, we will roughly describe the idea of modeling of a chosen mechanical system, some details of Józef Wojnarowski’s curriculum vitae as well as some analyses and description of His scientific publications.

Additionally, we will report about some papers related to the discussed topic to show background, trends, solved problems and a spectrum of possible utilization of graphs.

The last notion used in the title should also be explained i.e. ‘IFTToMMist’. The word has been created by Professor Marco Ceccarelli—former President of IFTToMM (The International Federation of the Promotion of Mechanism and Machine Science). In what follows, only abbreviation of the full name of this organization will be solely utilized. The discussed word means far more that only an IFTToMM member, namely a man who is called in this way should be an IFTToMM active officer, deeply and fully involved in scientific, promotive and administrative activities of this organization. Professor is just like the new word sounds.

14.2 Graph as a Model of Mechanical System

Graph $G(V, E)$ is a pair of sets, V —nonempty set of vertices and E —set of edges. An edge is a pair of elements belonging to V . We consider both sets as finite. It is a simple graph, when we additionally assume that there is not any edge from a particular vertex to itself (a graph without loops).

Modelling is an immanent activity of the theoretical analysis of every system. In an initial step, modelling consists usually in simplification and posing of some assumptions. In versatile approaches e.g. FEM, BEM, reverse engineering, flow analysis etc. simplification consists in e.g. formulation of reductive assumptions (prescribed DoF, linearity etc.) as well as digitalization i.e. replacing continuous body/element by means of particles, modes, bodies, points which are connected in a special way. Aiming for easiness in understanding, the process for will be illustrated via a chosen mechanism e.g. planetary gear. In Fig. 14.2, the general scheme of modeling process is shown. MS means mechanical system. For the purpose of analysis, it should be simplified and turn into the discrete one. We are performing this for the chosen exemplary artifact i.e. planetary gear. So, we focus our attention only on kinematics. For a short time, we neglect other mechanically important factors as vibrations, fatigue, quality, effectiveness etc. So we consider discrete parts: sun wheel—1, planetary wheel—2, ring (crown wheel)—3 and carrier (arm)—h (Fig. 14.3).

Path in Fig. 14.2: “16-1-2-3-4” leads to MS testing. In fact, in every case simplification and discretization is needed. We can calculate the ratio e.g. via Willis approach. Then we can verify the correctness via testing. The same result can be achieved tracing the path “16-1-5-6-7-8-9”. The abbreviation C-DKT means **cross-domain knowledge transfer** which means in our case: gear parts are modelled via graph vertices, relations between system parts are modelled via graph edges. Moreover, we consider types of edges in the following way: engaged/mesh pairs (1, 2) and (2, 3) (in Fig. 3b) are shown as dotted lines in Fig. 3c. Edge (2, 3) is drawn by means of double dotted line because element 3 (ring) is braked. Turning pair (2, h) i.e. (planet, carrier) is represented via continuous line in Fig. 3c. The pairs (1, 3), (1, h) and (3,

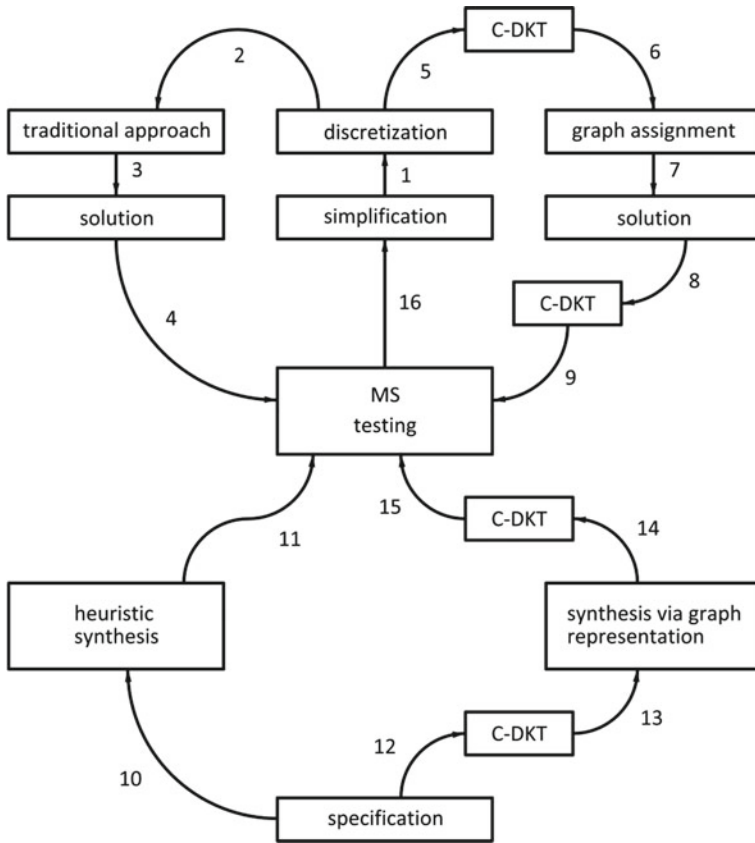


Fig. 14.2 Scheme of mechanical system modelling; MS—mechanical system and C-DKT—cross-domain knowledge transfer

h) create a triangle (in Fig. 14.3c). In general it could be a polygon. For some visual reasons and compactness, it is presented as shadowed area.

Here, we describe one exemplary method of MS modeling using a mixed graph i.e. by means of the modified Hsu’s graph [22]. Other methods can be found in other listed publications. After graph creation, we can solve the re-formulated problem in the domain of graphs. Then the solution is translated back into the mechanical field—see: path “7-8-9” in Fig. 14.2. The reason for graph usage could be disputable but it is well established alternative approach. Moreover, the dedicated software can do/aid the activities in an automatic and an algorithmic way. Algebraic equations are generated in the systematic manner. The bottom part of the scheme shown in Fig. 14.2 is dedicated to the idea of synthesis of mechanical systems and mechanisms, in particular. Based on a specification, an experienced engineer can propose the MS—see path “10-11”. Thereafter, a prototype can be tested. Path “12-13-14-15” in Fig. 14.2 represents the graph-based approach. After C-DKT, the transformed

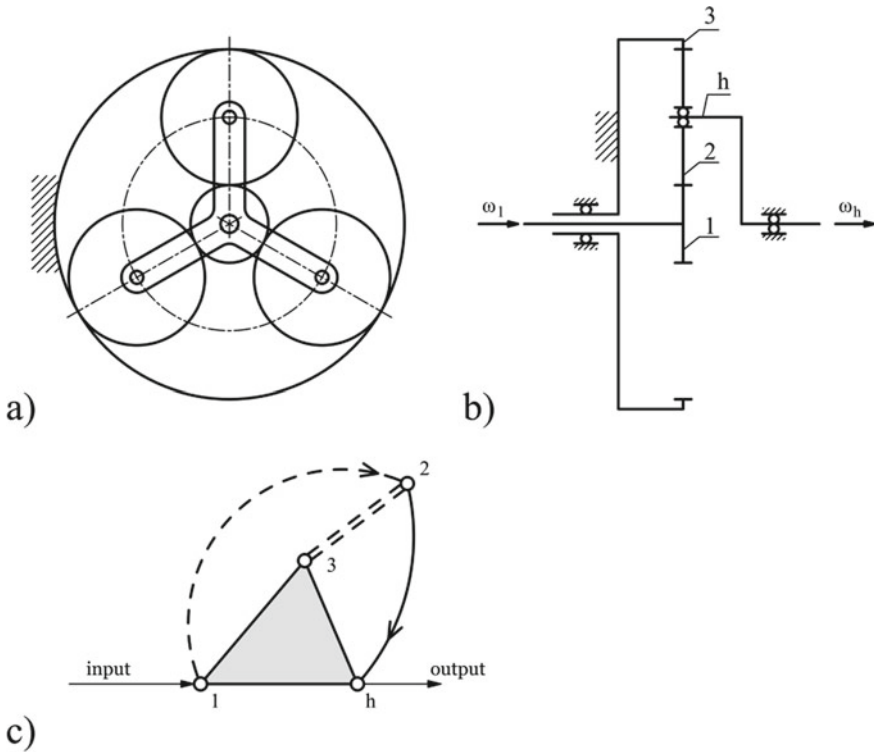


Fig. 14.3 Mixed graph model of planetary gear: 1—sun wheel, 2—planet (wheel), 3—crown (ring) wheel; h—carrier (arm)

problem can be solved as e.g. generation of graphs' family. Then, after turning the obtained graphs into mechanical systems [15], the family of variants (of the considered MS) is generated automatically. So, it can aid the engineer works effectively and essentially, giving—in same cases—full range of MS possible design forms for the assumed specification.

Further possibilities of utilization of the proposed way of modelling (after Hsu [22]) and the resultant calculation methods can be found in the authors' works [23–25].

These general ideas of graph-based modeling have been developed for many years. The methods were described in the best mechanical world-wide journals, especially *Mechanism and Machine Theory* (edited under IFToMM supervision) as well as *ASME Journal on Mechanical Design* and other [23–25]. On Polish market, there is *Journal of Theoretical and Applied Mechanics*, which will be mentioned—once more—underneath.

There is only a few books dedicated to the discussed topics e.g. in Polish [27, 28] where especially the idea of structural numbers is widely explained, in English, the

following books are available [29–31]. Therefore the idea of further propagation of the discussed methodology is worth to embodiment just through the present book.

14.3 Professor Józef Wojnarowski's Curriculum Vitae—Chosen Datums

14.3.1 *Introductory Data*

Józef Bolesław Wojnarowski was born on 3rd April 1933 in Stary Sącz. This is a very old Polish town placed in south region of the country. The region was granted in 1257 to Kinga—wife of Polish Prince Bolesław, by a ruler itself. Princess established a nun cloister in 1280. So, the town has a very old tradition, the monastery still exists, and additionally town was important in entire Polish history.

The birth town, the region as well as the family usually have an important influence on inhabitants, so it was the case. His brother was a playwright, an author of dramas for puppet theatre as well as a co-worker of the Polish State Radio preparing cultural radio broadcasts. In 1949, Professor finished first level secondary school in His hometown. Then He received a diploma of a field engineering (technician) from the ‘Śląskie Techniczne Zakłady Naukowe’ in Katowice (1952). The name of the institution is difficult to translate, in fact, it was the vocational secondary school of extremely high level of professional knowledge, equipped in modern laboratories, giving an ideal balance between theoretical and practical skills.

Katowice is a capital of Silesia—an industrial region, which due the second world war consequences, belongs entirely to Poland since 1945. The region was (and still is) full of industrial plants, mines (in decreasing number) as well as scientific institutes and versatile universities. Once more, both the town and the region have usually a strong, positive influence on their residents. Obviously, the personal features of a man allow for individual development. So, Professor is a very intelligent, active, full of energy scientists wanting to realize the established goals.

In general, ‘if you do not plan the future, in fact—you plan a defeat, a disaster or at least a boredom’. However, He belongs to individuals who can formulate tasks, goals as well as tools and methods for performance/achievement of them. Another feature of His character is arrangement of time! He studied (till 1958) and obtained Ph.D. title (1964) at the Silesian Technical University in Gliwice, Faculty of Mechanical Engineering (Fig. 14.4). Temporary, this institution was the third/forth the biggest technical University in Poland after Warsaw, Wrocław and Cracow (where, in fact, there are two distinct technical institutions). At the moment, the Silesian TU is considered by the Polish Ministry of Higher Education as belonging to the “top 10” Polish universities of all kinds!

So, the base for scientific career has been solid and powerful; i.e. very good education, strong family (Fig. 14.5), outstanding personal features of character and inspiring surrounding of leading Polish scientists.



Fig. 14.4 Silesian TU; **a** Mechanical Engineering Faculty; **b** main entrance; 75 anniversary of establishing the University i.e. 1945–2020; **c** innovation center; **d** logo and confirmation about belonging in the ‘top 10’ universities (so called “research”-one) All photos besides these made by T. Geisler were made by Stanisław Zawiślak

The results of His investigations were published in over 500 journal papers, conference proceedings papers, dissertations, academic lecture notes, collections of worksheets (exercises) as well as monographs and books. He is an author of five patents.

14.3.2 *International Experience and Activities*

Professor Józef Wojnarowski has an enormous international experience, taking part in tens of international conferences all over the world, paid numerous visits at versatile universities due to various invitations, projects and/or co-operation agreements. He has served as *invited professor* or held lectureships at the following institutions:



Fig. 14.5 Celebration of professor Wojnarowski's 80th birthday (*Photos Tomasz Geisler*)

Peter the Great St. Petersburg Polytechnic University, abbreviated as SPbPU/also, formerly known as “Saint Petersburg State Technical University”/(1970/1971, in the 70's of the twentieth century the name of the town was *Leningrad*), University of Alberta in Edmonton, Canada (1981), University of Niš, Serbia (1985), Omsk State Technical University /OmSTU/, Russia (1989), Monash University in Melbourne, Australia (1994), Open International University of Human Development “Ukraine”, Kiev (2002) as well as the Lvov Technical University—due to 160th anniversary of establishing. Moreover, He took part in the following international congresses and conferences e.g. IFTToMM Congresses: Zakopane (Poland 1969), Montreal (Canada 1979), Milano (Italy 1995), Oulu (Finland 1999), Tianjin (China 2004), Besançon (France 2007), Guanajuato (Mexico 2011) and the last one—jubilee (number 15th) in Cracow (Poland 2019). Another commonly known series of conferences is GAMM organized in Germany and neighbor countries. He attended e.g. editions—in: Zurich (Switzerland 2001), Augsburg (Germany 2002), Padua (Italy 2003), Dresden (Germany 2004), Luxemburg (2005) and many others as well as in some other series of conferences. Especially close scientific-didactic relationships were maintained by Professor with the Technical University of Doneck in Ukraine. Among other, He cooperated with Valentin Onishchenko, Ph.D., who received finally D.Sc. degree (so called: habilitation) as well as published common papers in high class journals. Some years later, in 1995, Professor obtained the

highest academic distinction i.e. *Doctor Honoris Causa* from Doneck TU. The same prestigious degree He received in 2007 from Lodz Technical University. To sum up, Professor J. Wojnarowski has been not only a participant of the mentioned events but also frequently a member of the scientific committees—preparing tens of deep, adequate and inspiring reviews. Moreover He always took part in discussions, asking questions, giving pieces of advice, suggesting further references and possibilities of generalization or continuation. All together caused that He was and still is a recognizable and fully active member of the academic societies and circles.

14.3.3 Professor Józef Wojnarowski as IFToMMist

J. Wojnarowski was an outstanding IFToMM member for decades. He was present at the initiation conference on MMS in Zakopane, Poland in 1969 when the IFToMM had been created [32].¹ One of the founding fathers—from the Polish side—was Professor Jan Oderfeld.

For many years Professor Wojnarowski was a president of IFToMM-Poland branch, i.e. since 2002 [33]. Since the beginning, this domestic organization was affiliated to the Polish Academy of Science. There were polish IFToMM officers e.g. IFToMM President Prof. Adam Morecki and Secretary General A. Morecki and Prof. Teresa Zielińska (recent two terms), member of Executive Council Prof. Krzysztof Kędzior. Prof. Józef Wojnarowski was the president of MO (Member Organization) for four consecutive terms, in recent years.

IFToMM is the organization which acts since 1969. During the Second World Congress on Theory of Mechanisms and Machines which took place just in Zakopane (Poland), two scientists, Ivan Artobolevski (USSR) and Erskine Crossley (USA), proposed to establish an International Federation of scientists and engineers to facilitate worldwide collaboration. The proposal was accepted and the Organization joins almost 50 national committees all over the world.

During the period of His domestic presidency, He initiated meetings of the Polish Committee on MMS (so called IFToMM-Poland branch) in versatile Polish scientific institutions. The plenary talks were always accompanied by scientific sessions and visits in the laboratories. It allowed the members of the body for comparison of level, laboratory equipment, procedures, certificates, study curricula etc. Upon His initiative, the Committee submitted opinions upon governmental acts on higher education in Poland as well as ranges of subjects related to MMS at polish technical universities.

The body (IFToMM-Poland) had visited e.g. Military Naval Academy in Gdynia, Military TU in Warsaw, Warsaw TU, Silesian TU in Gliwice, Wrocław TU and Białystok TU (accompanied with an excursion to Vilnius in Lithuania) and many others. In Fig. 14.6, we can see the members of the meeting in Warsaw TU where

¹ In the mentioned file, one can see the photo from Zakopane from 1969. Professor Wojnarowski is sitting against the rear wall, fist on the left in upper row.

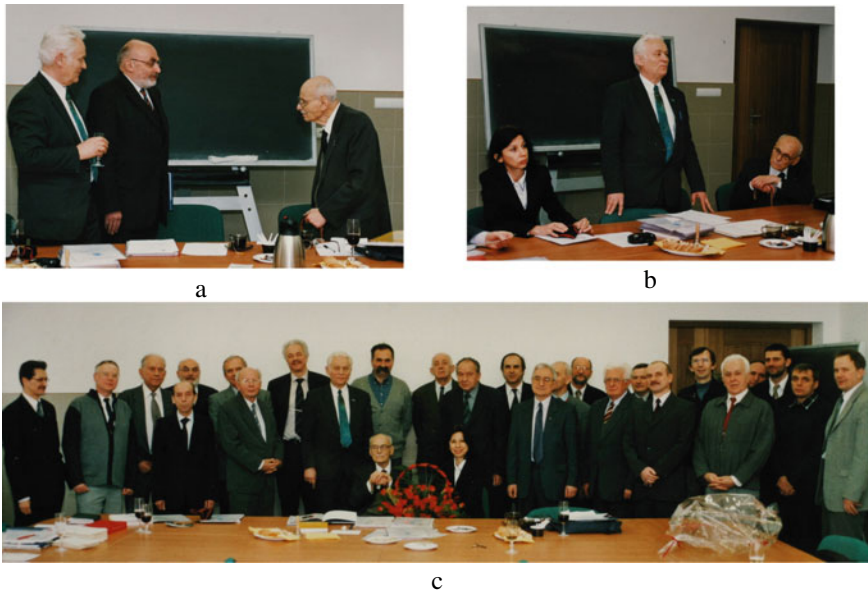


Fig. 14.6 Meeting of IFTToMM-Poland Committee in Warsaw in 2006; **a** Professors (from left) J. Wojnarowski, M. Trombski and J. Oderfeld, **b** from left T. Zielińska, J. Wojnarowski and J. Oderfeld, **c** all participants of the Meeting (*Photos Tomasz Geisler*)

Professor Jan Oderfeld (founding father of IFTToMM [1]) was a special Guest, who was the initiator of Polish Scientific Group on TMM in the 50's of the previous century. Later in 2008, being 100 y.o., Professor Jan Oderfeld received Doctor Honoris Causa degree from Warsaw TU. Many IFTToMM officers attended this ceremony e.g. Professor Marco Ceccarelli from Rome, Italy.

Professor J. Wojnarowski was also an initiator of unveiling of the plaque commemorating the 40-ties anniversary of establishing of IFTToMM. The ceremony took place in Zakopane in 2010. The plaque was placed at the entrance to the hotel 'Hyrny'. He supported and took part in all Polish conferences on MMS, which started to be organized biennially since 50-ies of the previous century. He (with His co-workers) even organized some editions. The conference has been converted lately in an international one. The conference papers had been printed in high level journals. The event circulated around all the institutions involved in MMS science in Poland. In many cases, the invited speakers were the IFTToMM officers, sometimes from neighbor countries sometimes representing governing bodies/councils (e.g. professors Andres Kecskeméthy and Marco Ceccarelli—current and former IFTToMM Presidents).

Recently, also with his attendance and support, due to some internal legislation changes, the IFTToMM-Poland group has been registered by the court as an independent society.

He, as a MO President, took part in the IFTToMM General Assemblies during World Congresses [34]: in Tianjin, China (2004), France (2007), Mexico (2011) and

Taipei, Taiwan (2015) [35]. He was an initiator of returning of IFToMM congresses to Poland—which idea was turned into practice due to His lobbying, talks, discussions and propagation of the concept. The organization of the 15-th Congress was granted to the AHG-Technical University in Cracow. The event took place in July 2019, to celebrate the 50-th anniversary of establishing of IFToMM in 1969 in Zakopane, Poland. In Fig. 14.7, we can see Professor J. Wojnarowski during this event, the most



a)



b)



c)



d)

Fig. 14.7 Professor Józef Wojnarowski at the jubilee 15-th IFToMM Congress in Cracow in 2019; **a** with Prof. Marco Ceccarelli; **b** giving a speech; **c** with (from the left) Prof. Tadeusz Uhl—Organizer of the Congress; Prof. Andres Kecskeméthy newly-elected IFToMM President (since January 2020) and Prof. Marco Ceccarelli IFToMM President (of that time); **d** with University Professor Stanisław Zawiślak—former Secretary of IFToMM-Poland (MO); *Photo Stan Zawiślak*

sound is the picture (c) where there are professors (from left): T. Uhl—Organizer of the Congress, A. Kecskemethy—current IFTToMM President, M. Ceccarelli—former president and the hero of our chapter.

At present—despite retirement—He is a member of some IFTToMM bodies i.e.: the Permanent Commission for History of MMS and the Technical Committee for Robotics and Mechatronics, being a propagator of mechatronics and an author of a lecture notes dedicated to this relatively new field. In the past cadences, He was a member of other bodies e.g. PC for Standardization and Terminology.

14.3.4 Other Achievements

Furthermore, we can mention that Professor Józef Wojnarowski was extremely active in development of young scientists' generations—like it was mentioned, being e.g. a supervisor of the doctor theses (19) as well as inspiring and giving opinions for international fellowships. These activities cover also issuing of reviews of doctor theses (55), habilitation theses (38)/so called D.Sc. title—known in some countries/all over of Polish technical universities as well as 28 motions for the professorship—governmental level. It means that He was recognized by the academic community and authorities as a fair, competent, experienced and proper scientist to evaluate so important scientific works. He wrote also several papers' reviews for high-class international journals and tens for conference papers e.g. related to IFTToMM congresses. He was additionally a member of bodies related to the Polish governmental institutions as well as related to versatile scientific societies. Within the years 1997–1999, he was a member of the ministerial committee for scientific titles in Poland. Among others, He—since 1963—has been a member of prestigious Polish Society of Theoretical and Applied Mechanics. The activities of the society overlap in some areas of the IFTToMM mission! In 1998, the authorities granted Him the title of Honor Member of the Society. The Society issues the Journal of Theoretical and Applied Mechanics which is indexed/abstracted in the Web of Science, Scopus and other scientific databases. He is a member of the Advisory (Scientific) Board of the Journal. Since 1998, He had been a member of the Committee for Machine Building of the Polish Academy of Science. Other chosen memberships are: EUROMECH—European Mechanics Society (since 1999), Academy of Engineering in Poland (since 1996, vice-president 2002–2006). In 2002, He became a member of the Ukrainian Academy of Engineering Sciences and also in 2003 He joined the European Academy of Science, Art and Literature.

14.4 Development of Graph Theory Modelling in Mechanics

In 1971 Professor Józef Wojnarowski initiated in Poland an application of graph theory in mechanical engineering [2], like it was mentioned before. In the next paper [3], He formulated theoretical bases for graph-based modelling. Underneath, dedicated reference review is given. Like it was mentioned, His publication list is very, very long. Here we focus our attention on a few of them, only i.e.: (i) strictly related to graph application, published in high class journals and (ii) representing some other directions of His activities. Moreover, there is a list of works of His co-workers/including current authors of the chapter/ (iii) as well as (iv) the chosen important publications of international background to give a glimpse of the field related to the present book.

The direction of investigations (graphs and mechanics) was initiated in the 50's of the previous century [36] in Canada, USA, Germany. Then, further groups joined the community—especially from China, Taiwan, India, Iran, Hungary, Spain, Romania, Israel, Russia, Brazil and finally from Egypt and Turkey. The review according to all directions is done in Zawiślak's D.Sc.-dissertation [15] and in relation to gears—in a very comprehensive (all-embracing) review-type paper [37].

Professor Józef Wojnarowski continued the works in this direction (graph-based modelling) together with the wide group of co-workers, publishing papers by Himself or within a group of co-authors. The book entitled “Application of graphs in analysis of vibrations of mechanical systems”—written by Him in Polish in 1991—was the first such monography on the Polish market [28]. It was edited under patronage of the Polish Academy of Science, by the prestigious domestic scientific publishing house. Several books (lecture notes, exercises) were written by his co-workers, as well. Lately, the book in MMS Series (edited as a series by prof. Marco Ceccarelli) was printed in Springer [31]. The present book is a continuation of the aforementioned one. The publication—edited by Zawiślak and Rysiński—had a great success, i.e.: it was evaluated in top 25% most downloaded (taking into account single chapters) by Springer [31] (in 2017). In this practical, comprehensive book, Professor Józef Wojnarowski published the historic paper [38] about famous Polish mathematician Kazimierz Kuratowski who proved theorem on planar graphs. The problem was open for tens of years, having incomplete approaches.

One of the pioneering papers on modelling of systems via graphs was published in 1955 [36]. The problems considered were e.g.:

- general ideas of modelling, philosophical background and knowledge transformations [39, 40], developed widely by Offer Shai [41–45];
- modelling of civil engineering structures [29];
- mechanisms analysis, modelling and synthesis [46, 47],
- multibody dynamics [48–51];
- enumeration and synthesis of structures [52–54],
- modelling of planetary gears [22–25, 54],
- other related problems, including isomorphism of structures [47, 56–60],

- truss calculations [61, 62].

The papers dedicated to graphs' application published by the Józef Wojnarowski's circle were dedicated to different graph types and different mechanical systems:

- (i) simple graphs [21, 63–66]; additionally—review type paper [67] which is frequently cited;
- (ii) signal flow graphs [68, 69];
- (iii) hypergraphs [17, 70–74];
- (iv) bond-graphs [75, 76];
- (v) modified Hsu's graphs [67];
- (vi) contour graphs [23–25, 54, 76].

So, we can conclude that the 'academic/science school' on graphs is fully recognizable and strongly confirmed. Other Professor Wojnarowski's directions of investigation were as follows: (a) theoretical and applied mechanics e.g. lecture notes [78–81]; (b) biomechanics [14, 82, 83]; (c) vibration analysis [84, 85]; (d) gear design and analysis [86]; (e) history of science [38, 87] and some others [88, 89]. All together approx. 500 publications were published by Him in different journals, conference proceedings, special issues, university lecture notes etc. All these confirms the Hero of the present text gave an enormous, important, wide and innovative contribution to graph-based modelling in mechanics.

14.5 Conclusion

Graph theory has versatile application, which is described in other chapters as well as in the present one. What more, this field of knowledge is nowadays under rapid and extremely wide development due to the fact that networks can be modelled as graphs. Professor Józef Wojnarowski has been pioneer of this direction of investigations in Poland which resulted in essential contribution of Polish scientists to the discussed scientific discipline.

Bond graphs are most widely and most commonly used due to commercial software allowing for their practical and computer-aided utilization, but a plain graphs are utilized, as well. Both types were widely utilized by the Hero of the present considerations.

Moreover, in recent years several papers were dedicated to synthesis of gear boxes and mechanisms via graph-based approach where families of variants were obtained—especially in year 2020 [90–94]. It seems that graphs in these solutions are irreplaceable if we focus our attention of algorithmic and systematic manner of such tasks.

Finally, Good Health to the Hero and further achievements!

References

1. Official IFToMM Web-page: <http://iftomm.net/>; access 10/2020
2. Wojnarowski, J.: Analysis of discrete linear mechanical systems of limited degree of freedom by means of the graph method, In: Proceedings of Polish-Czechoslovak Conference on Machine Dynamics, vol. 2, pp. 567–581 (1971)
3. Wojnarowski, J.: Graph as a language of the systems' structure. Scientific Papers of Silesian University of Technology, Series: Mechanics, Z., vol. 52, issue 389, pp. 3–21 (1973) (in Polish)
4. Arczewski, K.: Topological analysis of mechanical vibrating linear systems by means of structural numbers. Arch. Mach. Build. **19**(4), 589–605 (1972)
5. Arczewski, K.: Analysis and Synthesis of Vibrating Mechanical Systems Utilizing the Method of Structural Numbers. Doctor Thesis. Warsaw University. Technology. (in Polish) (1974)
6. Arczewski, K.: Topological analysis of a class of lumped vibrational systems by the method of structural numbers. J. Sound. Vibr. **97**(1), 75–86 (1984)
7. Arczewski, K.: Application of graph theory to the determination of potential energy of systems consisting of rigid bodies and springs. J. Franklin Inst. **324**(3), 369–386 (1987)
8. Arczewski, K.: Application of graph theory to the mathematical modelling of a class of rigid bodies systems. J. Franklin Inst. **327**(2), 209–223 (1990)
9. Arczewski, K.: Graph theoretical approach—I. Determination of kinetic energy for a class of particle systems. J. Franklin Inst. **329**(3), 469–481 (1992)
10. Arczewski, K.: Graph theoretical approach—II. Determination of generalized forces for a class of systems consisting of particles and springs. J. Franklin Inst. **329**(3), 483–491 (1992)
11. Arczewski, K.: Graph theoretical approach—III. Equations of motion of a class of constrained particle systems. J. Franklin Inst. **329**(3), 493–510 (1992)
12. Arczewski, K., Dul, F.: Determination of angular velocities within a multibody system by means of graphs. Z. Angew. Math. Mech. (ZAMM) **75** (SI), S105–S106 (1995)
13. Świder, J.: Matrix hybrid graphs in descriptions of complex vibrating mechanical systems. In: Scientific Papers of the Silesian University of Technology, Series: Mechanics, vol. 106, 221p (in Polish) (1991)
14. Wojnarowski, J., Margielewicz, J., Żochowski, L.: Modelling of human walking via bond multi-graphs. In: Proceedings of the 17th Conference of Biomechanics, Acta of Bioengineering and Biomechanics, vol. 3, issue 2, pp. 663–670 (2001)
15. Zawiślak, S.: The graph-based methodology as an artificial intelligence aid for mechanical engineering design. Habilitation-thesis, Publishing House of the University of Bielsko-Biała, Bielsko-Biała, 284p (2010)
16. Wojnarowski, J., Mirota, K.: Modelling of blood circulating system by means of graph matrix representation, graphs and mechanics. In: Proceeding of the Second International Conference, Gliwice, pp. 53–54 (1999)
17. Buchacz, A.: Modelling, synthesis and analysis of bar systems characterized by a cascade structure represented by graphs. Mech. Mach. Theor. **30**(7), 969–986 (1995)
18. Orlikowski, C.: Symbolic analysis of bond graphs by application of the Coates rule. Mech. Mach. Theor. **30**(7), 1019–1026 (1995)
19. Pikoń, A., Wojnarowski, J.: Application of structural numbers to generating the characteristics of mechanical systems. Mech. Mach. Theor. **30**(7), 1027–1037 (1995)
20. Świder, J.: Matrix hybrid graphs as models of complex vibrating mechanical systems. Mech. Mach. Theor. **30**(7), 1073–1098 (1995)
21. Wojnarowski, J., Zawiślak, S.: Modelling of mechanical system by means of matroids. Mech. Mach. Theor. **36**(6), 717–724 (2001)
22. Hsu, C.H.: Systematic enumeration of epicyclic gear mechanisms for automobiles. JSME Int. J. Ser. C Mech. Syst. Mach. Elem. Manuf. **42**(1), 225–233 (1999)
23. DREWNIĄK, J., ZAWIŚLAK, S.: Graph methods in kinematical analysis of multi-speed epicyclic gears. Int. J. Appl. Mech. Eng. **17**(3), 791–798 (2012)

24. Drewniak, J., Kopeć, J., Zawiślak, S.: Graph models of automobile gears-kinematics. *Int. J. Appl. Mech. Eng.* **19**(3), 563–573 (2014)
25. Drewniak, J., Zawiślak, S., Kopeć, J.: Analysis of complex planetary gears by means of versatile graph based approaches. In: *New Approaches to Gear Design and Production*, pp. 349–364. Springer, Cham (2020)
26. Drewniak, J., Kopeć, J., Zawiślak, S.: Kinematical and efficiency analysis of planetary gear trains by means of various graph-based approaches. In: *Theory and Practice of Gearing and Transmissions*, pp. 263–284. Springer, Cham (2016)
27. Bellert, S., Woźniacki, H.: Analysis and synthesis of electrical system by means of the method of structural numbers. WNT, Warszawa (in Polish) (1968)
28. Wojnarowski, J.: Application of graphs in analysis of vibrations of mechanical systems. Polish Academy of Science, PWN Warszawa (in Polish) (1991)
29. Kaveh, A.: *Structural mechanics: graph and matrix methods*. Macmillan Int. High. Educ. (1992)
30. Tsai, L.-W.: *Mechanism design: enumeration of kinematical structures according to function*. CRC Press, Boca Raton, USA (2000)
31. Zawiślak, S., Rysiński, J. (eds.): *Graph-Based Modelling in Engineering*, 247p. Springer (2017)
32. Ceccarelli, M., Lopez-Cajun, S.: A brief History through a Photo Gallery. http://iftomm.net/images/Documents/About/IFTToMM_History-min.pdf; poster about IFTToMM; access 10 Dec 2020
33. Wojnarowski, J.: The significance and role of IFTToMM Poland in the creative development of mechanism and machine science. In: *Technology Developments: The Role of Mechanism and Machine Science and IFTToMM*, pp. 367–382. Springer, Dordrecht (2011)
34. Ceccarelli, M.: The IFTToMM World Congresses and IFTToMM Presidents, http://iftomm.net/images/Documents/About/09_AllWCposterHistory_of_IFToMM-WC.pdf; access 10 Dec 2020
35. Wojnarowski, J.: Report on the 14th IFTToMM 2015 World Congress (International Federation for the Promotion of Mechanism and Machine Science IFTToMM) which was held on 25–30 Oct 2015, Taipei, Taiwan. *Int. J. Appl. Mech. Eng.* **20**(4), I–VI (2015)
36. Trent, H.M.: Isomorphisms between oriented linear graphs and lumped physical systems. *J. Acoust. Soc. Am.* **27**(3), 500–527 (1955)
37. Xue, H.L., Liu, G., Yang, X.H.: A review of graph theory application research in gears. In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*. vol. 230, issue10, pp. 1697–1714 (2016)
38. Wojnarowski, J., Zawiślak, S.: Kazimierz Kuratowski—biography and genesis of the theorem on planar graphs. In: *Graph-Based Modelling in Engineering*, pp. 233–246. Springer, Cham (2017)
39. Zawiślak, S.: Artificial intelligence aided design of gears based on graph-theoretical models. In: *IFTToMM Twelfth World Congress in Mechanism and Machine Science*, Besancon, France. <https://www.yumpu.com/en/document/read/22539646/artificial-intelligence-aided-design-of-gears-based-on-iftomm> (2007)
40. Zawiślak, S., Szypuła, Ł., Myśliwiec, M., Jagosz, A.: Some applications of graph transformations in modeling of mechanical systems. *Proc. GT-VMT* **3**, 332–345 (2008)
41. Shai, O., Preiss, K.: Graph theory representations of engineering systems and their embedded knowledge. *Artif. Intell. Eng.* **13**(3), 273–284 (1999)
42. Shai, O.: The duality relation between mechanisms and trusses. *Mech. Mach. Theor.* **36**(3), 343–369 (2001)
43. Shai, O.: Utilization of the dualism between determinate trusses and mechanisms. *Mech. Mach. Theor.* **37**(11), 1307–1323 (2002)
44. Shai, O.: Transforming engineering problems through graph representations. *Adv. Eng. Inform.* **17**(2), 77–93 (2003)
45. Shai, O., Mohr, Y.: Towards transferring engineering knowledge through graph representations: transferring Willis method to mechanisms and trusses. *Eng. Comput.* **20**(1), 2–10 (2004)

46. Ding, H., Feng, Z., Yang, W., Kecskeméthy, A.: Structure synthesis of 6-DOF forging manipulators. *Mech. Mach. Theor.* **111**, 135–151 (2017)
47. Ding, H., Hou, F., Kecskeméthy, A., Huang, Z.: Synthesis of a complete set of contracted graphs for planar non-fractionated simple-jointed kinematic chains with all possible DOFs. *Mech. Mach. Theor.* **46**(11), 1588–1600 (2011)
48. McPhee, J.J., Milad, G.I., Andrews, G.C.: Wittenburg's formulation of multibody dynamics equations from a graph-theoretic perspective. *Mech. Mach. Theor.* **31**(2), 201–213 (1996)
49. McPhee, J.J.: On the use of linear graph theory in multibody system dynamics. *Nonlinear Dyn.* **9**(1–2), 73–90 (1996)
50. McPhee, J.J.: A unified graph—theoretic approach to formulating multibody dynamics equations in absolute or joint coordinates. *J. Franklin Inst.* **334**(3), 431–445 (1997)
51. Oshinowo, O.M., McPhee, J.J.: Object-oriented implementation of a graph-theoretic formulation for planar multibody dynamics. *Int. J. Numer. Meth. Eng.* **40**(22), 4097–4118 (1997)
52. Lu, Y., Ye, N.: Type synthesis of parallel mechanisms by utilizing sub-mechanisms and digital topological graphs. *Mech. Mach. Theor.* **109**, 39–50 (2017)
53. Yan, H.S., Chiu, Y.T.: An algorithm for the construction of generalized kinematic chains. *Mech. Mach. Theor.* **62**, 75–98 (2013)
54. Yan, H.S., Chiu, Y.T.: An improved algorithm for the construction of generalized kinematic chains. *Mech. Mach. Theor.* **78**, 229–247 (2014)
55. Drewniak, J., et al.: Modified method of the kinematic analysis of planar linkage mechanism for non-stationary motion modes. In: *New Advances in Mechanisms, Mechanical Transmissions and Robotics*, pp. 15–23. Springer, Cham (2017)
56. Deptuła, A., Partyka, M.A.: Application of dependence graphs and game trees for decision decomposition for machine systems. *J. Autom. Mobile Robot. Intel. Syst.* **5**, 17–26 (2011)
57. Huang, P., Ding, H., Yang, W., Kecskeméthy, A.: An automatic method for the connectivity calculation in planar closed kinematic chains. *Mech. Mach. Theor.* **109**, 195–219 (2017)
58. Kecskeméthy, A., Krupp, T., Hiller, M.: Symbolic processing of multiloop mechanism dynamics using closed-form kinematics solutions. *Multibody Syst. Dyn.* **1**(1), 23–45 (1997)
59. Shanmukhasundaram, V.R., Rao, Y.V.D., Regalla, S.P.: Algorithms for detection of degenerate structure in epicyclic gear trains using graph theory. *J. Braz. Soc. Mech. Sci. Eng.* **41**(496) (2019)
60. Xu, X., Sun, H., Liu, Y., Dong, P.: Matrix-based operation method for detecting structural isomorphism of planetary gear train structures. *J. Mech. Des.* **142**(6) (2020)
61. Zawiślak, S., Jagosz, A.: Application of evolutionary algorithm and graph-based method of stress calculation to truss optimization, scientific publications. *Electron. Warsaw Univ. Technol.* **165**, 253–262 (2008)
62. Zawiślak, S., Matusiak, D.: Multi-objective evolutionary design of trusses using graph-based method for stress calculation. *Trans. Univ. Košice, Slovakia* **3**, 181–184 (2009)
63. Wojnarowski, J.: Graph representation of mechanical systems. *Mech. Mach. Theor.* **30**(7), 1099–1112 (1975)
64. Wojnarowski, J.: The graph method of determining the loads in complex gear trains. *Mech. Mach. Theor.* **11**(2), 103–121 (1976)
65. Wojnarowski, J.: Application of graphs in the analysis of vibrating 3-dimensional systems. *Int. J. Appl. Mech. Eng.* **21**(3), 761–766 (2016)
66. Zawiślak, S., Wojnarowski, J.: Matroid model of mechanical systems. In: *Book of Abstracts. Annual Scientific Conference GAMM 2001, ETH Zürich*, p. 165 (2001)
67. Wojnarowski, J., Kopeć, J., Zawiślak, S.: Gears and graphs. *J. Theor. Appl. Mech.* **44**(1), 139–162 (2006)
68. Wojnarowski, J., Lidwin, A.: The application of signal flow graphs—the kinematic analysis of planetary gear trains. *Mech. Mach. Theor.* **10**(1), 17–31 (1975)
69. Wojnarowski, J., Świder, J.: Method of independent contours for direct transformation of mechanical system into its signal flow graph. *J. Theor. Appl. Mech.* **16**(4), 507–515 (1978)

70. Buchacz, A.: Analysis of beam hypergraphs by means of exact and approximate methods as models of transverse vibrating subsystems in the synthesis of mechanical and mechatronic systems. *Arch. Mech. Eng.* **57**(4), 431–442 (2010)
71. Buchacz, A.: Transformed hypergraphs in synthesis of vibrating beam-systems. *PAMM* **10**(1), 361–362 (2010)
72. Buchacz, A.: Introduction to synthesis of the torsional vibrating discrete-continuous mechatronic systems by means of the hypergraphs and structural numbers method. *J. Vibroengineering* **14**(2), 514–519 (2012)
73. Buchacz, A., Wojnarowski, J.: The modelling of vibrating bar systems with nonlinear changeable sections of robots by means of hypergraphs and structural numbers. *J. Franklin Inst.* **332**(4), 443–467 (1995)
74. Buchacz, A., Machura, A., Pasek, M.: Hypergraphs in investigation of trajectory of robot's manipulator with links as thin-walled bars. *Autom. Constr.* **7**(5), 363–383 (1998)
75. Nowak, A., Czapla, K., Kaczmarek, K.: Modelling of vibrations of machines models by use of the hybrid bond graphs. *J. Theor. Appl. Mech.* **41**(4), 903–918 (2003)
76. Wojnarowski, J., Margielewicz, J., Gaska, D.: Identification of chaotic attractors of the overhead travelling crane model. In: *IFTToMM World Congress on Mechanism and Machine Science*, pp. 3047–3056. Springer, Cham (2019)
77. Wojnarowski, J., et al.: Application of contour equations to kinematic analysis of complex and compound planetary gears. In: *IFTToMM World Congress on Mechanism and Machine Science*, pp. 987–996. Springer, Cham (2019)
78. Nowak, A.: *Graphs. Theory and Problems*. Publishing House of the Silesian University of Technology, Gliwice, 148p (in Polish) (2006)
79. Wojnarowski, J.: *Laboratory of TMM*. Publishing House of the Silesian University of Technology, Lecture Notes No 684, Gliwice (in Polish) (1976)
80. Wojnarowski, J., Bogucki, Z.: *Mechanics. Exercises and Problems*. Publishing House of the Silesian University of Technology, Lecture notes No 1006, Gliwice (1983) (in Polish)
81. Wojnarowski, J., Buchacz, A., Nowak, A., Świder, J.: Modelling of vibration of mechanical systems by means of graphs and structural numbers method. Publishing House of the Silesian University of Technology, Lecture Notes No 1266, Gliwice (1986) (in Polish)
82. Wojnarowski, J., Mirola, K.: Experimental criteria for hemodynamical evaluation of the artificial mechanical heart valves. In: *Proceedings of the 17th Conference of Biomechanics, Acta of Bioengineering and Biomechanics*, vol. 3, issue 2, pp. 655–662 (2001)
83. Wojnarowski, J., Mirola, K.: On hydrodynamic characteristics of aortic mechanical artificial heart valves. In: *Proceeding of the 11th World Congress in Mechanism and Machine Science*, vol. 1, pp. 101–104. Tianjin University, China (2004)
84. Wojnarowski, J., Buchacz, A.: Application of graphs and structural numbers for determination of the characteristic equation and the frequency spectrum. *J. Theor. Appl. Mech.* **13**(4), 545–560 (1975)
85. Wojnarowski, J., Adamiec-Wójcik, I.: Free vibrations of a frame of a band saw using the rigid finite element method. *Mech. Mach. Theory* **40**(2), 241–258 (2005)
86. Wojnarowski, J., Onishchenko, V.: Tooth wear effects on spur gear dynamics. *Mech. Mach. Theor.* **38**(2), 161–178 (2003)
87. Wojnarowski, J.: 460th Anniversary of De Revolutionibus Orbium Coelestium. *Int. J. Appl. Mech. Eng.* **8**(3), 359–363 (2003)
88. Wojnarowski, J.: *Introduction to Mechatronics*. State Vocational University of Applied Sciences, Nowy Sącz (in Polish) (2012)
89. Wojnarowski, J., Borowik, B.: Applying the Zigbee technology for the enhancing the remote objects control. In: *PAMM: Proceedings in Applied Mathematics and Mechanics*, pp. 679–682. WILEY-VCH Verlag, Berlin (2009)
90. Ding, H., Cai, C.: Patent analysis and structural synthesis of epicyclic gear trains used in automatic transmissions. *Appl. Sci.* **10**(1), 82 (2020)
91. Du, M., Yang, L.: A basis for the computer-aided design of the topological structure of planetary gear trains. *Mech. Mach. Theor.* **145**, 103690 (2020)

92. Han, L., Liu, G., Yang, X., Han, B.: A computational synthesis approach of mechanical conceptual design based on graph theory and polynomial operation. *Chin. J. Mech. Eng.* **33**(1), 2 (2020)
93. Ho, T.-T., Hwang, S.-J.: Configuration synthesis of novel hybrid transmission systems using a combination of a Ravigneaux gear train and a simple planetary gear train. *Energies* **13**(9), 2333 (2020)
94. Yin, C., Tang, D., Deng, Z.: Research on configurations of multi-axis speed-differential mechanisms based on 2K-H gear train. *Mech. Mach. Theor.* **148**, 103783 (2020)
95. Drewniak, J., Zawiślak, S.: Kinematical and dynamical analysis of closed kinematic chains using graphs and contour equations. In: *PAMM: Proceedings in Applied Mathematics and Mechanics*, pp. 547–548. WILEY-VCH Verlag, Berlin (2009)
96. Drewniak, J., Zawiślak, S.: Linear-graph and contour-graph-based models of planetary gears. *J. Theor. Appl. Mech.* **48**, 415–433 (2010)
97. Drewniak, J., Zawiślak, S.: Comparison of graph-based methods of kinematical analysis of planetary gears. *Acta Mech. et Automatica* **4**, 14–18 (2010)
98. Drewniak, J., Kopeć, J., Zawiślak, S.: Analysis of automatic automotive gear boxes by means of versatile graph-based methods. In: *Advances in Mechanisms Design*, pp. 81–86. Springer, Dordrecht (2012)
99. Drewniak, J., Zawiślak, S.: Graph-based models of compound planetary gear boxes. In: *Solid State Phenomena*, vol. 199, pp. 143–148. Trans Tech Publications Ltd. (2013)
100. Drewniak, J., Zawiślak, S., Wieczorek, A.: Modelling multi-way planetary gears by means of contour graphs. *Solid State Phenom.* **220**, 126–131 (2015)