DOI:10.15774/PPKE.ITK.2014.002

Optimization-based analysis and control of complex networks with nonlinear dynamics

Komplex, nemlineáris dinamikájú hálózatok analízise
és irányítása optimalizálási módszerekkel

János Rudan

A thesis submitted for the degree of Doctor of Philosophy

Pázmány Péter Catholic University
Faculty of Information Technology and Bionics

Supervisors:
Prof. Gábor Szederkényi (PPCU FIT)
Prof. Katalin M. Hangos (HAS SZTAKI)

Budapest, 2014.

DOI:10.15774/PPKE.ITK.2014.002

Non est volentis, neque currentis, sed miserentis Dei

DOI:10.15774/PPKE.ITK.2014.002

Acknowledgements

First of all, I would like to thank *Prof. Katalin M. Hangos* and *Prof. Gábor Szederkényi* for their continuous support, guidance and encouragement during my studies.

I would also like to thank *Zoltán Tuza* for all the work we have done together through our student years.

I wish to express my gratitude to all my colleagues for their advice, and for discussing about my ideas: *Dóra Bihary*, *Bence Borbély*, *Dr. Csercsik Dávid*, *Balázs Jákli*, *Csaba Józsa*, *György Lipták*, *István Reguly*, *Norbert Sárkány*, *Dr. András Horváth*, *Miklós Koller*, *Mihály Radványi, Gábor Tornai* and *Dr. Kristóf Iván.* It was a pleasure for me to work with you. I also would like to thank *Bart Kersbergen*, *Ton van den Boom* and *Prof. Bart De Schutter* for the opportunity of joint research. I am grateful to the Delft University of Technology for hosting me as guest researcher.

I owe sincere thankfulness to *Dr. Judit Nyékyné Gaizler*, *Dr. Péter Szolgay* and *Dr. Tamás Roska*, and I could not have completed this work without the support of the Faculty of Information Technology and Bionics of Pázmány Péter Catholic University.

I am especially grateful to *Csenge* for all her patience and support.

Last but not least, I am very grateful to my mother and father and to my whole family who always supported me in all possible ways.

Part of the work has been supported by the grants K83440 and NF104706 of the Hungarian Research Fund.

DOI:10.15774/PPKE.ITK.2014.002

Abstract

There are several types of dynamical systems that can be described as networks. It is also known, that graph-based system description is a powerful tool to represent networked system structures on different levels of abstraction.

As the focus of the research of networked systems shifted towards the large-scale networks constructed from real-life data, the importance of highly effective computational methods applicable for the analysis and control of these systems increased. Due to the rapid development of computer technology and the underlying computational and analytical methods, optimization methods became an important tool in system theory, applied also for the optimal control of complex, networked systems.

The work summarized in this thesis focuses on the application of centralized, but parallelizable optimization-based methods in the analysis and control of networked systems having nonlinear dynamics. Two classes of networked systems are investigated because they come from basically different approaches of networked system description, while they can be handled with similar mathematical tools.

Firstly, new methods for the structural and dynamical analysis of kinetic reaction networks are proposed. With the help of the introduced algorithms, the search for different alternative realizations of dynamically equivalent or linearly conjugate reaction networks can be completed while considering dynamical and/or structural constraints. Moreover, most of the algorithms have polynomial time complexity enabling us to handle large scale, biologically relevant networks, too. Extensive simulations are completed to evaluate the performance and the correctness of the proposed methods.

Secondly, optimal rescheduling method for the control of railway networks in case of delayed operation is proposed. The presented controller is capable to generate new timetables for the network in order to minimize the sum of the train delays along the prediction horizon. Moreover, with the help of the proposed framework the sensitivity of the railway network can be measured in case of single delays. Additionally, a new model formulation is introduced having an advantageous constraint structure, which gives the opportunity of the deeper analysis of the dependencies between events and control actions in the network. The proposed methods are tested on the model of the Dutch railway network.

DOI:10.15774/PPKE.ITK.2014.002

Összefoglalás

A hálózatos formában történő reprezentáció gyakran hasznos eszköz bizonyos dinamikus rendszerek működésének megértésében és leírásában. Ismert tény, hogy a gráf alapú leírás hatékony eszköz a hálózatos struktúrájú rendszerek különböző absztrakciós szinten történő leírására.

Ahogy a hálózatos struktúrájú rendszerek kutatásának fókusza az adat-alapú, nagy méretű hálózatok vizsgálata felé tolódik, úgy nyernek egyre nagyobb teret az ilyen típusú rendszerek analízisére és irányítására alkalmazható, hatékony számítási módszerek. A számítógépes technológiák és a segítségükkel alkalmazott számítási és analitikai megoldások gyors fejlődésének köszönhetően az optimalizációs módszerek igen fontos eszközzé váltak a rendszerelméletben, melyeket gyakran alkalmaznak a komplex, hálózatos rendszerek optimális irányítására is.

A jelen dolgozatban összefoglalt munka fókuszában a nagyméretű, nemlineáris dinamikával rendelkező hálózatos struktúrájú rendszerek centralizált, de párhuzamosítható optimalizálási módszereken alapuló analízise és irányítása áll. Két hálózatokon alapú modellosztályt vizsgáltam, amelyek leírásai alapvetően más megfontolásokból származnak, ám mégis hasonló matematikai módszerekkel kezelhetők.

Egyrészről új módszereket adtam kinetikus reakcióhálózatok strukturális és dinamikus tulajdonságainak analízisére. A bemutatott módszerek segítségével dinamikusan ekvivalens ill. lineárisan konjugált alternatív reakcióhálózatok határozhatók meg dinamikus és/vagy strukturális korlátozások figyelembe vétele mellett. A javasolt algoritmusok legtöbbje polinomiális időbeli komplexitással rendelkezik, ami lehetővé teszi azt, hogy nagy méretű, biológiailag releváns hálózatokat kezeljünk segítségükkel. A bemutatott módszerek helyességét és teljesítményét kiterjedt szimulációkkal vizsgáltam.

Másrészről optimális újraütemezésen alapuló irányítási módszert javasoltam késéses esetek kezelésére vasúti hálózatokban. A bemutatott szabályzó új menetrendeket állít elő olyan módon, hogy a predikciós horizont mentén minimális legyen a vonatok késésének összege. Emellett a módszer lehetőséget ad a vasúti hálózat érzékenységének vizsgálatára vonatok egyedi késése esetén. Új, előnyös korlátozás-struktúrát mutató problémaalakot javasoltam, melynek segítségével a hálózatban történő események és a kontrollváltozók közötti kapcsolatok könnyen vizsgálhatók. A bemutatott módszereket a holland vasúti hálózat modelljén szimuláltam.

DOI:10.15774/PPKE.ITK.2014.002

Contents

DOI:10.15774/PPKE.ITK.2014.002

DOI:10.15774/PPKE.ITK.2014.002

Chapter 1

Introduction

Dynamical models have central role in several fields of science and technology. With them we can describe the operation of production processes, power generation systems, transportation networks and individual vehicles, agent-based systems etc. Besides models applied in traditional engineering fields, numerous biological processes and phenomena can also be understood and explained by creating their dynamical model. Dynamical modeling becomes necessary if the state of the investigated system, namely the quantities describing the properties of the system, evolves in time and/or space [96].

As a complex system we mean large-scale dynamical systems with complex structure containing nonlinearities. An important requirement in the modeling of complex systems is simplification: the model should most importantly describe only those phenomena and processes which have significant effect on the dynamics of the system. Using a dynamical model, the future behavior of the system can be predicted with respect to a given initial state, thus the analysis and simulation of the system is possible. These are crucial steps towards the controlling of the given system which is essential to achieve desired operation. The understanding and targeted manipulation of these kind of models are the main topics of system and control theory.

In order to describe a large scale, complex system having many components proper structuring of the model should be applied [15]. A modular and clean model formulation helps to capture and understand the most important properties of the investigated system. To accomplish this, a widely used and straightforward way is to separate the dynamics of the individual parts/elements and the connections between them leading to a description with networked structure. A deeply investigated and well known example of this phenomena is the theory of linear networks [42]. A more detailed review of the networked systems can be found in Sec. 1.1.

Nonlinearities in a system can be incorporated into the system model through several ways. Using continuous models, the simplest case is to introduce smooth nonlinearities optionally extended with discrete variables to describe switching-type events. In case of applying discrete models, the theory of discrete event systems [102, 7] is able to handle many

DOI:10.15774/PPKE.ITK.2014.002

important phenomena.

From these theories, it is known, that we can classify dynamical models based on different points of view [91]. By determining the model class corresponding to the investigated system, we are able to determine the set of applicable methods and techniques for the analysis and control of the given system.

Because of the complex nature of many practically relevant control problems, the methods applicable for nonlinear systems with a networked structure are especially important, but they present challenges at the same time. This thesis focuses on the optimization-based solutions of the control problems related to complex systems having networked structure.

1.1 Networks in system modeling

There are several types of dynamical systems that can be described as networks. Supply chains, transportation and public transport networks, in-cell reaction systems, genetic regulatory networks are some of the widely investigated systems having networked structure. Basically, networks consist of two main elements: a set of nodes and a set of links connecting the nodes to each other. Both the nodes and the links can be static or dynamic elements, depending on the properties of the system described by the network [15].

Usually, to describe the structure of networks in a mathematical framework, graph theory is used where nodes and links in the network are corresponding to vertices and edges in the graph, respectively. The way, how graphs can describe dynamical systems can have multiple interpretation: e.g. in case of transportation networks a topographical layout of vertices and edges can be handful, representing junctions and sections of the network. Meanwhile, a more abstract, graph-based description of a system is also possible, e.g. in case of discrete event systems, where vertices are standing for the different states of the system while edges represent possible state transitions. As it can be seen from these examples the graph-based system description is a powerful tool to represent several system structures on different levels of abstraction.

If we investigate a static system a graph-based representation of the system can describe structural dependencies between the components of the system where both the vertices and edges are static or passive. If the dynamics is also incorporated into the system description, the following setup is usual. The vertices represent the individual components of the system, while the edges between them set constraints on the behavior of the dynamics. The quantitative properties of the constraints usually appear as edge weights in a weighted, directed graph.

The topic of combining several, individual dynamical systems into a complex, networked system is elaborated in the theory of interconnected subsystems in systems and control theory. In [36] mathematical tools are provided to analyze the connectability of composite systems, which property ensures the controllability and observability of the complex system. The fact, that the controllability and observability of a networked system depends on the properties of

DOI:10.15774/PPKE.ITK.2014.002

the linkages between the components leads to the concept of *structural controllability and observability* [78]. These ideas are further developed in [79] showing the crucial properties of a relatively small number of driver nodes while controlling the dynamics of a complex networked system.

The importance of the quality and quantity of the connections in the graph is emphasized by the research of the dynamics appearing in extreme large networks, such as social networks [14]. The statistical analysis of the behavior patterns in these kind of networks enables us to understand the significance of weak connections in social communities, the reordering patterns in such systems etc. As several databases became available containing data about large-scale networks, the techniques to analyze the data moved towards fractal-based description [113], graph-focused data mining techniques [66] and other improved techniques, which instead of analyzing the individual elements of the network, focus on the graph-theory based description of the representing structures.

The work summarized in this thesis focuses on the application of optimization-based methods in the analysis and control of networked systems. Two classes of networked systems are investigated because they come from basically different approaches of networked system description, while they can be handled with similar mathematical tools. In case of a reaction network, the model is translated into a network by describing the relations between the terms of the underlying differential equations as links between the nodes of a network that represent the elementary nonlinearities. In contrast with this, in transportation networks (in this particular case, in railway networks) both the nodes and the links are passive, they are only a topological mapping of the routes, junctions and/or stations. Nonlinearities are introduced into this model by the absolute values appearing in the basic system model. Both system classes are nonnegative [60], and can be handled as smooth nonlinear systems (by incorporating vehicle dynamics into the microscopic modeling of the system).

1.2 Aim and structure of this thesis

The aim of the present thesis is to develop optimization based methods applied to the analysis and control of specific, practically important dynamical systems having complex networked structure. By analyzing the structure of the original, networked system and the structure of the emerging optimization problem, problem-specific improvements are proposed to decrease the complexity of the computational tasks, thus they become applicable on large scale networks, too. Generally, these tasks are formulated as MILP problems, but in several cases, they can be simplified to Linear Programming (LP) tasks.

In this thesis, two different system classes are examined having networked structure: kinetic reaction networks and railway networks. In both cases, we will use optimization-based techniques to solve problems in the field of system analysis and control. In case of reaction networks, structural (parameter-independent) system analysis and the computation

DOI:10.15774/PPKE.ITK.2014.002

of alternative reaction networks will be completed using optimization based methods.

In case of the railway networks control actions are going to be computed for optimal rescheduling in order to minimize the sum of the delays of the trains. By applying optimization based control methods, our aim is to decrease the sensitivity of the railway network against small appearing delays and by reducing the delay propagation effect. To accomplish this, a model formulation is needed which enables us to simplify the analysis of the effect of the dispatching actions with respect to the individual delays of the trains. It is also desired, that formulated optimization problems should be solvable in a reasonable time, thus the possibility to handle large-scale networks with algorithms using them is present.

The structure of the thesis is the following. In Chapter 2 the basic notions and the mathematical tools are introduced, which serve as a basis of the methods presented in the further parts of this work. Chapter 3 introduces the Kinetic Reaction Networks in details, summarizes the results known from the literature and the corresponding results of the author. In Chapter 4 the topic of railway networks are investigated, presenting the different model formulations, the proposed new methods and the simulation results. In Chapter 5 the main scientific contributions of the presented work are briefly summarized and the possible further developments are enumerated.

1.3 Optimization in system analysis and control

As it is detailed in [24], optimization problems play an important role in several fields of system theory. In particular, any controller design problem is in fact a constrained optimization problem with the control aim as loss function and the system model as a constraint.

Due to the rapid development of computer technology and the available computational and analytical methods, several new application area of optimization methods appeared in system theory. Two application domains have become particularly important: the computational methods themselves and those conceptual developments which make it possible to implement the developed methods in real-life applications (e.g. Model Predictive Control framework [53]). Considering the underlying computational methods, it can be said, that most of the formulated optimization problems can be traced back to mathematical programming problems which are able to handle cost functions and constraints on the variables.

With the help of the tools emerging from optimization several different tasks in system analysis and control can be solved: identification problems, parameter estimation tasks and control problems can also be formulated in such a framework [80]. Identification and parameter estimation form a closely inter-related set of problems where the application of optimization methods focuses on the computation of system parameters using the presented, usually noisy and limited measurement data [126]. Several methods applied in the control of dynamic systems can also be traced back to optimization: e.g. the method of Linear Matrix Inequalities [23] to design robust controllers, decentralized control [54] to deal with systems

DOI:10.15774/PPKE.ITK.2014.002

incorporating multiple decision maker units, Linear-Quadratic (LQ) control problems [83] in classical control theory or state estimation [112].

Techniques applied in optimal control were further developed to the control of networked systems, where the entities of the system are connected to each other via links, altogether considered as a graph structure as it is detailed before. Optimization tasks are formulated from the coordinated control task of the entities in case of different connectivity properties [68], synthesis of complex process plants with a networked structure [50] satisfying different type of constraints and having objective functions with multiple components etc. Also, methods are developed to compute different properties and representations of a given system having networked structure, such as minimal representation, identifying key connections etc. One of the developed methods is capable of computing a maximal superstructure corresponding to a given process network in polynomial time [49] in a centralized way. From the computed superstructure, with the help of additionally applied constraints all possible solutions of the original problem can be extracted enabling us to analyze the properties of the solutions with high computational efficiency. As the focus of the research of networked systems shifted towards the large-scale networks constructed from real-life data, the importance of the highly effective computational methods applicable for the solution of optimization problems increased.

Problem formulation in an optimization framework has a great conceptual importance: in case of a properly formulated problem its feasibility can be checked, meaning that the existence of at least one solution can be shown even if the underlying problem is hard to solve. The fact, that infeasibility (non-existence of a solution) can be explicitly detected has great importance both in theory and in application.

The aim of the present thesis is to investigate the analysis and control of complex nonlinear dynamical systems having networked structure using optimization-based techniques. This topic is examined through two problems, namely the structural and dynamical analysis of kinetic reaction networks and the dynamical rescheduling problem of railway networks. In case of both problems, we formulate the dynamical model of the corresponding networked system and the solutions of the examined tasks are derived to a mathematical optimization problem. In case of reaction networks a structural analysis has been completed and alternative realizations are computed, while in case of railway networks, the effect of delay propagation is reduced by applying optimal control actions in the schedule of the trains.

1.4 Networks and dynamics

The research of systems composed by a large number of interconnected dynamical units gained increased attention recently. The analysis and modeling of large scale coupled systems from the field of biology, chemistry, transportation, logistics, social sciences etc. became an important topic in science as the computational performance of the computer systems grows

DOI:10.15774/PPKE.ITK.2014.002

and the detailed processing of huge amount of data is possible. This gives us the opportunity to shift our attention from the investigation of individual properties of small-scale networks towards the analysis of large scale systems with complex interconnection patterns, sometimes by focusing only on the statistical properties of the modeled system. Some important results about these topics are summarized in [89].

The two main issues while investigating a complex coupled system are the following: firstly, to identify the structure of the connections between the actors of the system in order to describe the system with the help of phenomena known from classical network theory. Here the central task is to properly define the nodes in the network and the links in between them, which usually considered as vertices and edges in the graph corresponded to the network, respectively. Secondly, the type and behavior of the interactions should be identified with respect to the formulated structure. These can be described as the properties of the edges in the graph of the network. The obtained network models can be applied to analyze and if it is needed, to control the underlying dynamics. A detailed review of the possible model classes and their applications can be found in [19].

Among many others, complex networked systems can be classified based on the role of the nodes in the network. In one hand, there are model types where the nodes have active role in the dynamics of the network (they have some generalized computing task), and the network can be considered as a set of agents connected to each other as it is defined by the structure of the network. On the other hand there are models where nodes are only interconnecting elements in the network connecting different edges, but they do not have specific ”computational” tasks. These kind of models are similar to pipeline networks, where nodes are created at the junctions of pipe sections.

Let us shortly review some system classes that can be modeled with the previously introduced, agent-based network formulation.

Artificial neural networks are motivated by biological neural networks, and their aim was to create a computational method that has as strong learning capabilities as the biological neural networks have. The proposed perceptron model [106] is based on strong simplifications of the biological neurons but still tried to capture their main properties. The artificial neural network is defined as a set of processing nodes which have similar role than the cell body of neurons while weighted edges connect the processing nodes as the axons connect neurons. The connectivity pattern strongly influences the behavior of the network and some classical configuration of artificial networks emerged such as feed-forward networks and Hopfield-networks [64].

The phenomena of individual processing nodes connected by weighted edges further developed leading to the appearance of *Cellular Neural Networks (CNN)* [31]. CNNs are a general framework to build parallel processing units with nonlinear dynamics arranged topographically in a grid. The connections between the nodes are local: only neighboring processing units can communicate with each other, thus the behavior of the whole network

DOI:10.15774/PPKE.ITK.2014.002

is characterized by the local connections. The obtained architecture called CNN Universal Machine is capable to combine analog array operations with local logic. The application areas of these kind of spatio-temporal universal machines are very wide and they have interesting connections with the state-of-the-art supercomputers and many-core computational devices [107].

Another interesting example of this field is the (bio-)chemical reaction networks. In [63] it has been expressed that modern biology should not only describe the function of individual cellular components but their interconnections and interactions should also be explained, as a complex network of biochemical elements. Thus, computer-aided investigation of genetic regulator networks, intracellular signaling pathways and in-vivo reaction cascades is one of the main topics of computational biology. The nonlinear dynamics appear in these kind of networks can be captured with several mathematical model classes. In this work, we focus on the application of Chemical Reaction Network Theory (CRNT) to describe and analyze (bio-)chemical reaction networks. In these models, the nodes of the network are the chemical complexes (consisting of different chemical substances) and they are interacting with each other in chemical reactions described by the edges. CRNT can be further generalized leading us to a set of mathematical problems which can produce several complex dynamical phenomena as it is detailed in Chapter 3.

Behavioral patterns in systems consisting of living entities such as animals or humans can also be described with the help of networks. A.-L. Barabási and T. Vicsek investigate these topics in details, while uncovering several manifestation of complex network-based dynamics in living systems. As it is detailed in [14], several very complex phenomena appearing in the human society (e.g. scale-free properties of social networks, importance of weak links between people in the society) can be explained with the help of mathematical tools known from network and graph theory. The fundamental importance of network theory-based analysis of behaviors in living communities is also emphasized in [132, 131], where both the events in the groups and the evolution of the group are explained with the help of relatively small changes in a scale-free networked structure. The phenomena learned from living communities often applied in robotic systems consisting of several robots, networked sensors or cooperative components [94, 129]. In [90] a wide variety of graph-theory based mathematical constructions modeling epidemiological processes are summarized. It has also been shown that the spreading of several diseases can be efficiently simulated with network theory based models.

As it was mentioned, besides of networks having so-called agents in the nodes, there are several types of networks, where the nodes are just meeting points of edges. Let us now examine a subclass of these pipeline-like networks, namely the class of *transportation networks*. In transportation networks, the nodes are usually considered as junctions, crossings or stations and the edges are routes or tracks. Vehicles are moving along the edges towards a pre-defined target node or along a pre-defined route while following some type of scheduling and considering safety constraints. It can be seen that transportation networks usually can

DOI:10.15774/PPKE.ITK.2014.002

be depicted as topographically ordered graphs. Analysis of transportation networks (e.g. traffic flow analysis on highways, train traffic analysis in railway networks) can have several different aims: to find the most sensible parts of the network (e.g. with respect to accidents, delays, traffic jams etc.), find parts of the network which are suboptimally used etc. It is known that vehicles in the network can have complex dynamics (see e.g. [70]) emerging from the properties of the network and the presented constraints. To control these behaviors, the tracking and control of individual vehicles is needed [130]. Controlling traffic has extreme importance in case of railway networks, because of the increased load on the tracks and the fact that delays can be quickly propagated all over the network due to the limited rerouting capability and the lack of overtaking. To overcome these issues several railway control method were developed [133] to increase throughput of railway networks, to limit delay propagation, to minimize passenger delays etc. A control method to minimize delays in a railway network is proposed in Chapter 4.

Considering these, it can be said that the network-based analysis and control of large size, interconnected systems is a widely investigated but still current topic in science. The complex dynamics that can be modeled within this framework can describe a large variety of real-life phenomena, and the understanding and control these kind of systems can have huge impact on several problems both in science and everyday life, too.

1.5 Kinetic reaction networks as unified models of smooth nonlinear systems

Nonnegative systems are dynamical systems having the property that all state variables stay nonnegative if the system is started from the nonnegative orthant. If none of the states of a nonnegative system can reach zero, than we speak about a positive system. Nonnegative systems appear in several fields of science, usually in cases where there are physical constraints on the nonnegativitiy of the states, such as population dynamics or (bio-)chemistry. It should be noted that with the help of proper coordinate transformation, many systems can be transformed to be nonnegative. These transformations usually consist of two parts: the first is the shifting of the coordinates into the nonnegative orthant, then the second is a time-scaling [120] ensuring that the trajectories of the system remain in the desired operation domain.

An interesting nonnegative system class is the class of kinetic systems, which are closed thermodynamic systems under isobaric and isothermal conditions. Kinetic systems can be interpreted as an extension of chemical reaction systems, where the system contains chemical species reacting with each other influencing the evolution of the system over time. The extension involves the relaxation of some of the assumptions, such as mass conservation, naturally present in chemical reaction systems thus enabling complex nonlinear behavior of the kinetic system class.

The state vector of kinetic systems is formulated from the concentration of the species, which are nonnegative by nature. Kinetic systems can have smooth nonlinearities, and by

DOI:10.15774/PPKE.ITK.2014.002

considering the special structure of the system model some advantageous dynamic properties, such as global asymptotic stability may be ensured. Usually, these systems are described by a set of ordinary differential equations (ODEs) with polynomial right-hand sides. The appearing polynomials describe the elemental nonlinearities in the system model. The underlying dynamics can be characterized by different considerations, such as mass-action kinetics, Michaelis-Menten kinetics [85], Hill-kinetics [41] in (bio)chemical applications, etc. Due to the similarity of infection processes to chemical reactions, a wide variety of epidemic spreading models are also based on kinetic models. In this work, we are considering only kinetics systems with mass-action kinetics.

A subclass of nonnegative systems with nonlinear dynamics is the class of kinetic systems obeying the mass-action law [67]. This law is originated from the molecular collision picture of chemical reactions. In this phenomenon, a reaction occurs if two molecules which are able to react with each other collide. Hence, the probability of reaction depends on the probability of the collision, which is proportional to the concentration of the reactant species. Systems with mass-action kinetics are able to produce several important dynamical properties which are in the focus of nonlinear system analysis, such as different equilibria, oscillatory behavior etc. Deterministic positive polynomial systems with mass-action kinetics are called as Chemical Reaction Networks (CRNs) [48]. With CRNs many (bio-)chemical reaction structure can be described and because the strong descriptive capabilities of this system class, they are applied in numerous other fields such as physics or nonlinear control theory. A generalization of CRNs, namely the Kinetic Reaction Networks (KRNs) are introduced in details in Chapter 3.

Since KRNs gained increased attention recently due to their wide application area as detailed above, their analysis is an interesting and important topic. There are several cases, where some dynamical properties produced by a KRN can be predicted just from the structure of the graph independently from the actual parameter values appearing in the network model [47]. Considering this, the capability to analyze large-scale biochemical reaction networks depends only on the computational complexity of the available algorithms dealing with the structural analysis of KRNs.

In this thesis several new algorithms are presented to compute KRNs with preferred dynamical and/or structural properties. Some of the presented methods are computationally improved: by substituting the former NP-complete method with algorithms having polynomial complexity the proposed framework is now capable of handling large size networks, too. Moreover, a new method is presented which is able to incorporate new type of constraints corresponding to prescribed properties while computing alternative reaction networks.

DOI:10.15774/PPKE.ITK.2014.002

1.6 Transportation networks

Transportation networks are interesting and widely investigated examples of networks with complex dynamics. The main components of these networks are the topological network of routes (air corridors, railway tracks, highways, streets) and the vehicles moving along them. If any disturbance (accident, route blocking etc.) appear in the traffic, the capacity of the network can be reduced dramatically leading to unsatisfied passengers, increased transportation costs or other inconveniences. To avoid these problems traffic control methods are applied to reschedule or reroute vehicles if it is needed. While controlling a transportation network in such a way, a control aim is targeted (minimizing delays w.r.t. a predefined schedule, maximizing throughput of the network etc.) while important safety measures should also be considered.

The ever increasing load on the railway networks in recent years poses serious challenges for network managers. To ensure the smooth operation of the network especially in case of delayed operation, a lot of research effort has been put in the topic of timetable design. Delays can be caused by technical failures, accidents, weather conditions or other unexpected situations. Because in most cases a delayed train obstructs a whole track and through this the following and connecting trains will also be affected, delays can quickly propagate all over the network [33]. To avoid such a large scale interruption in the network stable and robust timetables [58] are designed. But in case of large delays modification of the schedule, such as rerouting or reshuffling trains or breaking connections can be necessary to minimize the effect of the disturbance. Proper rescheduling of the trains give us the opportunity to limit the propagation of the delay and recover the nominal operation of the network as soon as possible [69]. However, breaking connections can lead to high passenger delays while keeping train delays low [125]. From a computational point of view, the solution of scheduling problems boils down to mathematical programming problems. A comprehensive survey of scheduling methods used in railway management can be found in [136].

A delay-management problem handled as mixed-integer programming first appeared in [110] and more recently in [124]. In [20, 21] a permutation-based methodology was proposed which uses max-plus algebra to derive a Mixed Integer Linear Programming (MILP) to find optimal rescheduling patterns [65]. These control methods have the advantage against greedy algorithms (e.g. [127]) that they can guarantee an optimal control action with respect to the performance index, but on the other hand they could have issues regarding the computational time.

In this thesis, we propose new model formulations for model predictive controllers applied to the control of railway networks. The rescheduling problem is traced back to the solution of a MILP problem, but in case of large and dense networks the increase of solution speed is needed. With the help of the proper restructuring of the MILP problem, significant speedup is achieved. Also, a new model formulation is proposed which can lead to the development of problem-specific analytic tools and solution methods.

DOI:10.15774/PPKE.ITK.2014.002

Chapter 2

Basic tools and notations

In this Chapter, we will introduce the main concepts and tools used in this work. We will shortly review the theory of optimization and some classes of optimization problems used in the methods presented in Chapters 3-4. Also, the main ideas behind the applied solution methods of these type of optimization problems are introduced. Moreover, we will summarize some results corresponding to the topic of dynamical systems represented by networks.

2.1 Convex optimization

A mathematical optimization problem has the following form [24]:

$$\begin{cases} \min f_0(x) \\ w.r.t.\ f_i(x) \le b_i \quad i = 1, ..., \omega. \end{cases}$$

where the vector $x = (x_1, ..., x_k)$ contains the so-called optimization variables, the function $f_0 : \mathbb{R}^k \to \mathbb{R}$ is the objective function and functions $f_i : \mathbb{R}^k \to \mathbb{R}$, $i = 1, ..., \omega$ define the constraints. Constants b_i, $i = 1, ..., \omega$ are the limits for the constraints. A vector x^* is called *optimal solution* if it has the smallest objective value from the set of vectors satisfying all the constraints. An optimization problem is a *convex optimization problem* if both the objective function and the constraint functions are convex, meaning that they satisfy the following inequality:

$$f_i(\alpha x + \beta y) \le \alpha f_i(x) + \beta f_i(y)$$

for $i = 0, ..., \omega$ and for all $x, y \in \mathbb{R}^k$, $\alpha, \beta \in \mathbb{R}_0^+$ where $\alpha + \beta = 1$.

Among many other subclasses of convex optimization, we will shortly introduce two widely used special subclasses: linear optimization and least-squares. An optimization problem is *linear*, if the following holds:

$$f_i(\alpha x + \beta y) = \alpha f_i(x) + \beta f_i(y)$$

DOI:10.15774/PPKE.ITK.2014.002

for $i = 0, ..., \omega$ and for all $x, y \in \mathbb{R}^k$, $\alpha, \beta \in \mathbb{R}$. It can be seen that if an optimization problem is linear, than it is convex, too. An optimization problem is called as a least-square problem if it contains no constraints but the objective function has a special form $a_i^T x - b_i$:

$$\min f_0(x) = \|Ax - b\|_2^2 = \sum_{i=1}^{k} (a_i^T x - b_i)^2$$

where $A \in \mathbb{R}^{\omega \times k}$, $\omega \geq k$, the rows of A denoted as a_i^T and again, x stands for the vector of the optimization variables.

Both can be solved numerically very efficiently while they have a fairly complete theory and they are used in a wide variety of applications, too. However, recently several related important developments have appeared. The *interior-point methods* [77] developed in the 1980s are able to solve linear programming problems and in general, convex optimization problems as well. Besides of these, convex optimization became a central topic in the area of automatic control systems, estimation and signal processing, communications and networks, data analysis and modeling etc. as the techniques based on *Linear Matrix Inequalities* (LMIs) [23] earned more and more attention in the recent years. Convex optimization is also widely applied in combinatorial optimization and global optimization problems to find optimal solutions, approximate them or find bounds on them.

Considering these, it can be advantageous to formulate a given problem in the convex optimization framework, because it is proven that convex optimization problems can be solved (meaning that a solution can be found or it can be proven that no solution exists), moreover, reliable and efficient methods are present to compute the solution. Also it should be noted, that in general the formulation of a convex optimization problem has serious effect on the complexity and computational difficulty of the solution. Hence, the investigation of the problem structure and the methods applied during the solution are important in order to achieve advantageous problem formulation to avoid unnecessary computational issues. With the help of the theoretical results on convex optimization, a model form applicable for distributed solution can be obtained, or sometimes a formulation with advantageous properties is achievable.

There are several available software packages [139, 59, 140] that implement different solution methods and processing techniques for convex optimization problems and specially, linear optimization problems. The methods applied by the solvers are different, therefore solver selection can have serious effect on the computational complexity of the solution in case of a given optimization problem.

2.2 Linear Programming

Linear Programming (LP) is perhaps the most successful discipline of the field of operations research [104]. A linear program is a constrained convex optimization problem, where a linear function of the real-valued optimization variables is minimized (or maximized) with respect

DOI:10.15774/PPKE.ITK.2014.002

to linear equality and inequality constraints.

Application fields of linear programming are very wide. From mathematical economics to linear algebra there are several topics in which linear programming plays a central role. Linear programs can be formulated to incorporate problems like portfolio optimization tasks, manufacturing and transportation problems, routing and network design methods in the field of telecommunication, traveling salesman-type of problems used for vehicle routing or VLSI chip board design etc.

In the following we will define the LP problem itself and then the main ideas of the solution methods are summarized.

2.2.1 Problem formulation

A standard LP problem is formulated as follows:

$$\begin{cases} \min\limits_x c^T x \\ Ax \leq b \\ x_i \in \mathbb{R},\ i = 1, ..., k \end{cases} \tag{2.1}$$

where x is the k-dimensional vector of *decision variables* consisting of real valued elements. The collection of ω linear inequality constraints are defined by matrix $A \in \mathbb{R}^{\omega \times k}$ called as *constraint matrix* and vector $b \in \mathbb{R}^k$. With the above formulation, equality constraints can also be treated by rewriting the problem to contain purely inequality constraints [30]. The linear function $c^T x$ with $c \in \mathbb{R}^k$ is the *objective function* to be minimized. This formulation describes a simplex in $\mathbb{R}^k$ where the given constrains define the bordering hyperplanes. It is known that the optimal solution of the LP has to be in one of the corners of the simplex, although there may be multiple alternative optimal solutions.

Let us note that this formulation defines the so-called *primal LP*. For each primal LP there exists a *dual LP*, which can be obtained from the primal problem directly by proper algebraic transformations. The dual problem of the LP defined in eq. (2.1) can be expressed as:

$$\begin{cases} \min\limits_y b^T y \\ A^T y \geq c \\ y_i \in \mathbb{R},\ i = 1, ..., k \end{cases} \tag{2.2}$$

As it can be seen, the problem formulations of the primal and dual LPs are connected to each other, moreover, it can be said that if a linear program has an optimal solution $x*$, then so does its dual (let us denote it as $y*$) and their objective values are equal: $c^T x* = b^T y*$ [24]. These problem formulations have different properties exploited in the solution methods, too.

The solution of linear programs is a widely investigated topic because of it's crucial importance in several application areas. Both the theoretical and implementational part of the methods have a wide literature: we suggest to review for example [88, 95].

DOI:10.15774/PPKE.ITK.2014.002

2.2.2 Solution methods

The main tool for solving LP problems in practice is the class of simplex algorithms proposed by Dantzig [34]. While considering the issue of computational complexity, the practical performance of the simplex algorithm is satisfying, because in case of a wide class of LPs the number of iterations during the solution seemed polynomial or even linear in the dimensions of problems being solved. Although, examples having exponential complexity were constructed few decades later then the original publications about the simplex method appeared. However, methods derived from nonlinear programming techniques, based on Karmarkar's work [77] can also handle certain classes of linear programming problems with outstanding efficiency while ensuring polynomial computational complexity in general [88].

In the following, we will shortly review these methods in order to introduce the main elements of the algorithms. A comprehensive survey of the LP solution methods can be found in [72].

2.2.2.1 Simplex method

The simplex method is the most widely used method to solve LPs, originally proposed in [34]. Recall that any LP problem having a solution must have an optimal solution that corresponds to a corner of the simplex corresponding to the LP. Hence, the method iterates over these corners while trying to move towards the optimal solution. The simplex method is based on a tableau formulation which allows us to evaluate various combinations of decision variables to determine how to improve the solution. A specific simplex tableau describes a given corner of the simplex corresponding to the problem.

Let us summarize the main points of this method based on [105].

1. Formulate the LP and construct a simplex tableau. Add slack variables, if it is needed (e.g. because of the reformulation of inequalities into equalities). Select the initial set of the basic variables and set the other variables to 0.

2. Find the sacrifice and improvement rows. These rows indicate what will be lost and gained in the cost function by making a change in the decision variables.

3. Select an entering variable, which is a currently non-basic variable that will most improve the objective if its value is increased from 0.

4. By applying a selection method (e.g. random selection, selecting the most limiting decision variable etc.), pick a basic variable (different from the currently entering variable) that will be excluded from the basic set. Mark it as the exiting variable.

5. Construct a new simplex tableau. Replace the exiting variable in the basic variable set with the new entering variable and change the corresponding rows in the tableau properly.

6. Repeat steps 2 through 5 until you no longer can improve the solution.

DOI:10.15774/PPKE.ITK.2014.002

Step No. 5. (namely the change of the basic variable set) is called as *pivot operation*. As it can be seen, the simplex method is basically a sequence of pivot operations. Note that the selection method applied to pick the entry and exit variables determines the number of iterations needed to find the solution. Hence, this basically determines the (worst-case) behavior of the solution method in terms of computational complexity [84].

2.2.2.2 Interior Point Methods

There are at least three major types of interior point methods (IPMs): the potential reduction algorithm which most closely embodies the constructs of Karmarkar (see [77] for details), the affine scaling algorithm which is perhaps the simplest to implement, and path following algorithms which combine the excellent behavior of the above two in theory and practice. Because of its advantageous properties, the third method-family (namely the path following methods) is implemented in the state-of-the art solvers.

Let us summarize the main points of the IPM method based on [72]. For a detailed explanation of the appeared concepts see [88], too.

The primal-dual path following algorithm is an example of an IPM that operates simultaneously on the primal and dual linear programming problems. The use of path following algorithms to solve linear programs is based on three ideas [72]:

- the application of the Lagrange multiplier method of classical calculus to transform an equality constrained optimization problem into an unconstrained one;
- the transformation of an inequality constrained optimization problem into a sequence of unconstrained problems by incorporating the constraints in a logarithmic barrier function that imposes a growing penalty as the boundary defined by the constraints in the model is approached;
- the solution of a set of nonlinear equations using Newton's method, thereby arriving at a solution to the unconstrained optimization problem.

As it is detailed in [72], the steps of the IPM can be summarized as follows:

1. Look for feasible initial solutions for the primal and dual problem.
2. Test optimality by computing the optimality gap. If the gap is under the prescribed threshold, the solution is found, return with it.
3. Compute the direction of the next step for Newton's method.
4. Compute the step size for Newton's method.
5. Take a step in the Newton direction as the update of the solution.
6. Repeat steps 2-5.

DOI:10.15774/PPKE.ITK.2014.002

Note that during solution process, it is assumed that the constraint matrix A from eq. (2.1) has full rank. This is usually achieved by some preprocessing of the presented constraint set. From an implementational point of view, the main issue is performing the matrix inversions needed to compute the Newton directions, which is usually handled by implementing a proper factorization instead of direct inversion.

2.2.3 Comparison of solution methods

In the following, we will shortly compare the two main solution methods of the Linear Programming problem, namely the Simplex Method and the Interior Point Method.

	Simplex Method	Interior Point Method
Theoretical (worst-case) complexity	NP	P
Practical complexity	P	P
Interpretation	clear geometrical: visiting the vertices	complex exploration of the feasible region
Best applicable for	small problems	large, sparse problems
Generalizable to non-linear problems	no	yes

2.3 Mixed Integer Linear Programming

Mixed Integer Linear Programming (MILP) is a special case of linear programming, where some of the decision variables are integer valued. A MILP can be treated as a class of combinatorial constrained optimization problems with continuous and integer decision variables, where the objective function and the constraining linear inequalities are linear. Some optimization problems having nonlinear constraints and/or nonlinear cost functions can be transformed to MILP problems: some nonlinear constraints can be handled as a set of linear constraints (where the introduction of auxiliary variables can be necessary), and some nonlinear cost functions can be approximated by piecewise linear functions [28, 111].

A wide variety of real life problems boil down to an MILP problem. In problems, where some of the resources (represented by the decision variables) are quantized and solutions containing non-integer values for these are meaningless, the application of integer valued decision variables is inevitable. MILP problems arise in the field of logistics, economics and social science. Moreover, the combinatorial problems, like the knapsack problem, warehouse location problem, machinery selection problem, set covering problems and many scheduling problems can also be solved as MILPs.

In this Section the problem formulation of the MILP problem and its main solution techniques are reviewed.

DOI:10.15774/PPKE.ITK.2014.002

2.3.1 Problem formulation

An MILP problem can be stated as follows [25]:

$$\begin{cases} \min\limits_{x} c^T x \\ Ax \leq b \\ x_i \in \mathbb{R},\ i = 1, ..., k \\ x_j \in \mathbb{Z},\ j = k+1, ..., l \end{cases} \tag{2.3}$$

where, similarly to the LP problem (see eq. (2.1)), x is the l-dimensional vector of decision variables consisting of k real and $l-k$ integer elements. Note that those constraints that define that some of the decision variables should be integer valued, called *integrality constraints*. Matrix $A \in \mathbb{R}^{\omega \times l}$ and vector $b \in \mathbb{R}^l$ define the set of linear inequality constraints containing ω constraints. Equality constraints can also be treated by rewriting the problem to contain purely inequality constraints [30]. The linear function $c^T x$ with $c \in \mathbb{R}^l$ is the objective function to be minimized. As it can be seen, if all decision variables are real (i.e. $k = l$), then eq. (2.3) defines a standard LP problem. If all the decision variables are integer valued (i.e. $k = 0$), than the obtained problem is called as an Integer Program (IP).

2.3.2 Solution methods

The solution methods of MILPs have some similarities with the ones used to solve pure LPs. Again, the linear constraints define a simplex in an l-dimensional space, but in this case, the optimal solution is searched on a lattice of feasible integer points instead of the corners of the simplex. The integer programming problems can have multiple local optima and finding a global optimum is ensured if and only if it is shown that the given solution has better objective value than all the other feasible points. In other words, it means that MILP problems do not scale very well w.r.t. the size of the original problem. It has been shown that MILP problems are generally NP-hard [93], hence they are usually solved by computationally very intensive heuristics-driven techniques which are sometimes unreliable in case of a large-scale problem.

The three main classes of MILP solution methods are shortly reviewed in the following. As an *LP relaxation* of a MILP problem we mean the LP problem obtained from the MILP by excluding the integrality constraints. For further details about the presented methods, see e.g. [81, 115, 51].

2.3.2.1 Cutting plane method

Cutting plane method are based on polyhedral combinatorics. These methods can be shortly summarized as follows. The algorithm is looking for parts of the simplex defined by the LP relaxation of the MILP problem, which can be excluded from the search space by introducing extra constraints while the remaining space still includes the integer solution. New constraints are introduced until the obtained LP-solution (restricted by the original and the added constraints) fulfills the integrality constraints, too. It is shown in [55] that

DOI:10.15774/PPKE.ITK.2014.002

the cutting plane method presented by Gomory [56] is finitely convergent. Several different methods are known to compute proper cuts [92].

Due to the great variety of the possible techniques of cut generation this method can have increased impact on the solution process. A bunch of different cut generation techniques are included in the available solvers (e.g. using the CGL library [137]).

2.3.2.2 Tree-search-based methods

By solving the LP relaxation of the original MILP problem, a solution can be obtained which contains at least one variable which takes real value but its domain is restricted to integer values by the integrality constraints, and rounding it to an integer results in the violation of at least one constraint. In this case, auxiliary constraints can be added to the relaxed LP guiding the solution away from the current non-integer value, towards the fulfilling of the integrality constraints required by the original MILP.

Considering these, searching for the optimal solution ends up in a tree-search between possible (MI)LP problems (called partial solutions) that fulfill some of the integrality constraints, but not necessarily all of them. Heuristics-driven tree search exploration techniques are used to build up, handle and explore the search tree emerging during the solution of a MILP [101]. The efficient exploration of the emerging large size search tree is necessary for the solution of the MILP problem.

As the solution process progresses, the new nodes are introduced into the search tree representing new problems generated from the original LP by adding extra constraints to it. This process called *branching*. Some of the nodes will be thrown away, because the problem represented by them can not have better solution than other, already examined node. This step is called as *bounding*. By the combination of these two steps, the so-called *branch-and-bound method* is created.

The main steps of the branch-and-bound technique applied for MILP solution are the followings:

1. Relax the integrality constraints from the original problem. Solve the resulting LP to obtain a global upper bound on the MILP objective function value. If the LP solution has integer values for those variables that were defined as integers in the original MILP, the optimal solution is present.

2. *Branching:* (Otherwise,) there should be a variable defined as discrete but having real value. Choose a non-discrete variable and branch on it: create new nodes, one for each rounded value of the variable (e.g. two node for the down- and up-rounded values). Insert the nodes into the search tree.

3. Select a node from the tree which will be expanded later on. It should be noted that several sophisticated algorithms and heuristics are available for node selection, e.g.

DOI:10.15774/PPKE.ITK.2014.002

depth-first logic (select a partial solution with most fixed variables), best-first (select a partial solution having with best parent bounds and heuristic values) etc.

4. *Bounding:* Create an LP relaxation of the problem represented by the selected node and solve it. If the LP is infeasible or the obtained objective value exceeds the current upper bound, prune the node. If the solution is lower than the current lower bound and feasible to the MILP mark it as the new incumbent solution. The incumbent solution is the currently best (w.r.t. the objective value) feasible solution known up to this point.

5. If there is no remaining node, mark the current incumbent solution as final solution. If there is at least one unvisited node, jump to Step 2.

Naturally, there are several modification and extension of this general technique, which mainly address the issue of the computational effort needed to explore the tree (see e.g. [18, 17, 101]). In general, one should note that heuristics applied during the branch-and-bound process have enormous impact on the solution speed and the quality of the solution. Each solver has its own implementation which makes them suitable for different type of problems. A deeper analysis of the solvers' performance can be found in [143].

2.3.2.3 Decomposition algorithms

Decomposition algorithms are trying to isolate sets of constraints from the original problem to generate multiple separated, smaller size (thus easy to solve) optimization problems. Auxiliary variables are introduced to link the otherwise independent subproblems. Then the results of the subproblems are combined properly to obtain the solution of the original problem. The main decomposition techniques are summarized in [99, 100], and reviewed in details in [51]. All of these methods can be interpreted as polyhedral approximations of the optimal solution and they can be handled in a common framework, as it has been done in [51].

In general, there are two main decomposition techniques: the Dantzig-Wolfe method [35] (denoted as DWD) and the Lagrangian method [108] (denoted as LD).

The LD method works as follows. Some constraints are selected to be omitted from the constraint set and introduced into the cost function together with the multiplicators introduced in a Lagrangian fashion. Also, auxiliary constraints generated from the dual of the LP relaxation of the original problem are introduced into the model, and the model is split into two separate parts one for the original constraints and one for the dual constraints. The solution of the dual problem helps to improve the bounds on the original problem which can eliminate possible solutions from the search space.

The key idea of DWD is to reformulate the original problem by substituting its variables with a convex combination of the extreme points of the polyhedron corresponding to a substructure of the formulation. The problems generated for the substitutions are solved as

DOI:10.15774/PPKE.ITK.2014.002

independent subproblems (similarly to the column-generation methods [9]) and a coordinating program generates the result of the original problem based on the sub-results.

A common property of the decomposition methods is that their computational effectiveness strongly depends on the structure of the original problem, namely the the structure of the constraint matrix. The most advantageous formulation is when matrix A from eq. (2.3) has a block-angular form, meaning that the nonzero elements are ordered into non-overlapping blocks while the constraints connecting these blocks are also grouped together. There are automatic processes (some of them are included into state-of-the-art solvers as preprocessors) which try to reorder the constraint matrix to obtain this form [16], but in general, a well-formulated constraint set based on problem-specific knowledge outperforms these methods.

DOI:10.15774/PPKE.ITK.2014.002

Chapter 3

Applying optimization methods to find kinetic reaction networks with preferred structural and dynamical properties

The analysis of the structural properties and dynamical behavior of biologically motivated kinetic systems is a quickly developing field. The rigorous structural and dynamical analysis of biologically motivated kinetic systems such as intracellular signaling pathways and gene regulation networks has gained an increased attention. Meanwhile, the amount and quality of experimental data are continuously improving due to the fast development of sensors and computer systems. Determining the structure and the exact parameters in such a network can be difficult due to the complexity of the described system or imperfect data. It is known that there are several important properties that only depend on the structure of the model, while the reaction graph structure corresponding to a given kinetic dynamics is generally non-unique.

These facts motivate us to construct algorithms that can compute kinetic systems with preferred structures (e.g. weakly reversible, minimal or maximal number of reactions, etc.) that may provide useful information about the dynamical behavior of the system. The motivation behind the parallel improvement of modeling and computational methods is clear, to be able to handle the growing amount of data and to analyse more complex, possibly biologically relevant processes and networks. By using optimization based methods, a clear framework can be built to handle the emerging computational tasks. The advanced methods implemented in the state-of-the-art solvers enable us to solve large optimization problems in parallel which results in relatively moderated solution times.

By Kinetic Reaction Networks (KRNs) we mean deterministic kinetic systems obeying the mass action law. Some of the assumptions, however, that are natural for chemical reaction

DOI:10.15774/PPKE.ITK.2014.002

systems, such as mass conservation, are relaxed when considering the mathematical structure of KRNs that enables us to use them as an important general descriptor class in dynamic system theory. It is known that such systems form a wide class of smooth nonlinear systems that are able to produce all important qualitative phenomena in nonlinear dynamics such as oscillations, multiplicities and even chaos [43]. Thus, the kinetic system form can be useful for the description of nonnegative models outside of (bio)chemistry, e.g. for epidemic, transportation or economic models as well [60, 109].

The general applicability of kinetic models is definitely extended by the strong results of Chemical Reaction Network Theory (CRNT). CRNT was initiated in the 1970's and 80's with the first publications about the relations between the structure and qualitative dynamics of CRNs treated as a general nonlinear system class [67, 47, 46]. Since then, numerous deep and useful results have been published in this continuously developing field (see, e.g. [43, 12, 114]).

In this Chapter, the description of dynamical systems with the help of Chemical Reaction Network Theory and the corresponding concepts are described. In Section 3.2 two new LP-based algorithms are presented to compute dynamically equivalent realizations containing minimal and maximal number of reactions. In Section 3.3 a new LP-based method is proposed to find dynamically equivalent, weakly reversible realizations while in Section 3.4 a MILP-based method is introduced to compute dynamically equivalent realizations with mass conservation.

3.1 Kinetic Reaction Networks

In this Section, the structural and dynamical description of KRNs are introduced based on [118, 121, 122]. Besides the notations, some important properties are also recalled related to the scope of the current work.

3.1.1 Describing a kinetic system as a KRN

The set $\mathcal{S} = \{X_i, \dots, X_n\}$ represents the n (chemical) species contained in a given KRN. The concentrations of the species denoted by $x_i = [X_i]$, $i = 1, \dots, n$ form the state vector $\mathbf{x} \in \mathbb{R}^n$ of the system. The whole system obeys the mass action law and, therefore, all the values of the states are nonnegative [60]. Complexes are formally represented as nonnegative linear combinations of the species:

$$C_j = \sum_{i=1}^{n} \alpha_{i,j} X_i \text{ for } j = 1, \dots, m, \tag{3.1}$$

where m is the number of the complexes in the network, and $\alpha_{i,j}$ for $i = 1, \dots, n$, are the nonnegative integer stoichiometric coefficients of the jth complex. An elementary reaction step, where the source complex $C_j = \sum_{i=1}^{n} \alpha_{i,j} X_i$ is transformed into the product complex $C_l = \sum_{i=1}^{n} \beta_{i,l} X_i$ is denoted by

$$C_j \to C_l \tag{3.2}$$

DOI:10.15774/PPKE.ITK.2014.002

The reaction rate corresponding to reaction (3.2) can be written according to the mass action law as:

$$\rho_{j,l}(\mathbf{x}) = k_{j,l} \prod_{i=1}^{n} x_i^{\alpha_{i,j}}, \tag{3.3}$$

where $k_{j,l} > 0$ is the reaction rate coefficient.

If for any $i \neq l$ both reactions $C_i \to C_l$ and $C_l \to C_i$ are present in the network (i.e. we have a reversible reaction), they are handled as separate elementary reactions. It is also required from the above model class that $C_i \neq C_j$ for $i \neq j$, $i, j = 1, \dots, m$, and self-reactions (i.e. loop edges) of the form $C_i \to C_i$ are not allowed for $i = 1, \dots, m$.

3.1.1.1 Dynamical description with ODEs

In the literature dynamical systems representing chemical reaction networks are usually described by the state vector $\mathbf{x}$ containing the concentrations of the species (or chemical components), the stoichiometric matrix S and the vector-valued function $R(\mathbf{x})$ representing the reaction rates as follows [71]:

$$\dot{\mathbf{x}} = SR(\mathbf{x}). \tag{3.4}$$

The stoichiometric matrix S is a constant matrix containing structural information about the network. The matrix is composed as follows: each row of the matrix corresponds to a specie, each column of the matrix corresponds to a reaction in the network, positive values appear for products and negative values for reactants. Thus, the rank of the stoichiometric matrix equals to the number of independent reactions in the network [13]. This formulation serves as a base of many methods formulated to compute reaction networks with special properties, such as minimal networks, identifying key reactions etc. [76].

This framework can be further generalized by the introduction of kinetic reaction networks which have dynamics obeying the mass action law can be described by a set of polynomial differential equations. In this case, chemical reaction networks can be considered as a subclass of kinetic reaction networks.

From the several different possibilities, we will use the following computationally advantageous factorization of the right hand side of the kinetic ODEs describing the dynamics of the concentrations (see, e.g. [48, 118]):

$$\dot{\mathbf{x}} = Y \cdot A_k \cdot \psi(\mathbf{x}) \tag{3.5}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the vector of specie concentrations. $Y \in \mathbb{R}^{n \times m}$ is the complex composition matrix in which the jth column contains the stoichiometric coefficients of complex C_j, i.e. $Y_{i,j} = \alpha_{i,j}$. The vector mapping $\psi = [\psi_1 \ \dots \ \psi_m]^T \in \mathbb{R}^n \to \mathbb{R}^m$ is defined as:

$$\psi_j(\mathbf{x}) = \prod_{i=1}^{n} x_i^{Y_{i,j}}, \ j = 1, \dots, m \tag{3.6}$$

DOI:10.15774/PPKE.ITK.2014.002

Matrix A_k describes the reaction graph as follows:

$$[A_k]_{i,j} = \begin{cases} k_{j,i}, \text{ if } i \neq j \text{ and reaction } C_j \to C_i \text{ is present in the CRN} \\ 0, \text{ if } i \neq j \text{ and } C_j \to C_i \text{ is not present in the CRN} \end{cases} \tag{3.7}$$

Moreover, the non-positive diagonal elements of A_k are given by

$$[A_k]_{i,i} = - \sum_{\substack{l=1 \\ l \neq i}}^{m} [A_k]_{l,i}. \tag{3.8}$$

Hence, A_k is a Metzler-type column conservation matrix (that is actually the negative transpose of the Laplacian matrix of the reaction graph), often called the Kirchhoff matrix of the KRN [86, 87].

A set of polynomial ODEs is called kinetic, if it can be written in the form eq. (3.5), where Y contains pairwise different columns of nonnegative integers, eq. (3.6) holds, and A_k is a Kirchhoff matrix. Necessary and sufficient conditions of the kinetic property for polynomial systems with a constructive proof were first given in [62] (see also [27]). Let us introduce the matrix M for the monomial coefficients of the model eq. 3.5, i.e.

$$M = Y \cdot A_k. \tag{3.9}$$

Using the above notation, eq. (3.5) can be written as

$$\dot{\mathbf{x}} = M \cdot \psi(\mathbf{x}). \tag{3.10}$$

3.1.1.2 Graph representation

A KRN can be represented as a weighted, directed graph $D = (V_d, E_d)$ called *reaction network* or simply *reaction graph*, consisting of a finite nonempty set V_d of vertices and a finite set E_d containing ordered pairs of distinct vertices called directed edges. The complexes are represented by the vertices, i.e. $V_d = \{C_1, \dots C_m\}$, and the edges stand for the reactions: $(C_j, C_l) \in E_d$ if complex C_j is transformed to C_l in one of the reactions in the network. The weight of the edge (C_j, C_l) is the reaction rate coefficient $k_{j,l}$. A reaction network is called *reversible*, if whenever the reaction $C_j \to C_l$ exists, then the reverse reaction $C_l \to C_j$ is also present in the network. By the structure of a KRN we mean the unweighted directed graph of the reaction network.

DOI:10.15774/PPKE.ITK.2014.002

3.1.2 Dynamical equivalence and linear conjugacy of KRNs

It is known that the polynomial differential equations of a given kinetic system can have multiple different reaction graph structures. Considering that in this thesis we use the form eq. (3.9), still several (Y, A_k) pairs can be found which can describe the same dynamical behavior. In this present work, we limit the set of complexes to a fixed set (generated using the method described in [62]), hence matrix Y is fixed.

The matrix pair (Y, A_k) is called a realization of a KRN described by M if eq. (3.9) holds, all the elements of Y are nonnegative integers, and A_k is a Kirchhoff matrix. It follows from the structural non-uniqueness that many important structural properties are not encoded uniquely in the polynomial differential equations of a kinetic system i.e. they are realization properties. In the following two Subsections, two phenomena are introduced corresponding to the explained non-uniqueness of the realizations.

3.1.2.1 Dynamical equivalence

Since the factorization eq. (3.9) is generally not unique (even if Y is fixed), the KRNs defined by the pairs $(Y^{(1)}, A_k^{(1)})$ and $(Y^{(2)}, A_k^{(2)})$ are called dynamically equivalent realizations of the kinetic system in eq. (3.10) (or that of each other) if $Y^{(i)}$ are valid complex composition matrices (a complex composition matrix is called valid if it contains nonnegative integer elements and there are no identical columns in it), $A_k^{(i)}$ are Kirchhoff for $i = 1, \dots, 2$, and

$$Y^{(1)} \cdot A_k^{(1)} = Y^{(2)} \cdot A_k^{(2)} = M. \tag{3.11}$$

In general the alternative realizations may contain complexes that do not appear in the original network. In this work we restrict the focus of our search to alternative KRNs where the set of complexes are the subset of the set of complexes of the original network.

Considering the above facts, it is possible to determine different realizations for the same dynamics for which the reaction graph has prescribed structural properties. Let us introduce the following notations: we call a realization *sparse* if the number of the non-zero elements in matrix A_k^s is minimal. A realization is called *dense* if it contains the maximal number of non-zero elements in matrix A_k^d. It was shown in [118] that the dense realization is a unique superstructure containing all mathematically possible reactions, i.e for a given complex set, it contains all possible other realization structures as sub-graphs of the dense one. It should be noted that while the dense realization is necessarily unique the sparse realization may not be unique: possibly several different sparse structures and/or parameter set can represent the same dynamical behavior.

Example. To illustrate the notion of dynamical equivalence on a small-scale example, we will use a classical 3-dimensional kinetic Lorenz system that was introduced in [128]. The classical Lorenz system can be transformed to kinetic form through coordinates-shifting and an

DOI:10.15774/PPKE.ITK.2014.002

appropriate time-scaling (see [128] for the computation details). After these transformations, the kinetic ODEs of the system are

$$\begin{aligned}
\dot{x}_1 &= \sigma x_1 x_2^2 x_3 - \sigma x_1^2 x_2 x_3 + \sigma(w_1 - w_2) x_1 x_2 x_3 \\
\dot{x}_2 &= (\rho + c_3) x_1^2 x_2 x_3 + (w_2 - w_1\rho - w_1 w_3) x_1 x_2 x_3 - x_1 x_2^2 x_3 - x_1^2 x_2 x_3^2 + w_1 x_1 x_2 x_3^2 \\
\dot{x}_3 &= x_1^2 x_2^2 x_3 - w_2 x_1^2 x_2 x_3 - w_1 x_1 x_2^2 x_3 + (w_1 w_2 + \beta w_3) x_1 x_2 x_3 - \beta x_1 x_2 \bar{x}_3^2
\end{aligned} \quad (3.12)$$

where $\sigma = 10, \rho = 28, \beta = 8/3$, and $W = [w_1\ w_2\ w_3] = [24\ 25\ 26]$. It is shown in [128] that using these parameters, eq. (3.12) shows chaotic behavior with an attractor that is very similar to the attractor of the classical non-kinetic Lorenz system.
The complex composition matrix of the system is given by:

$$Y^l = \begin{bmatrix} 1 & 0 & 2 & 1 & 2 & 1 & 1 & 1 & 2 & 2 & 2 & 1 & 2 \\ 1 & 1 & 1 & 2 & 2 & 0 & 1 & 2 & 1 & 0 & 1 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 & 0 & 0 & 2 \end{bmatrix},$$

where the ith column contains the composition of complex C_i (i.e. $C_1 = X_1 + X_2 + X_3$, $C_2 = X_2 + X_3$ etc.). The non-zero off-diagonal elements of the network's Kirchhoff matrix A_k^l are the following:

$[A_k^l]_{2,1} = 679.3324,\ [A_k^l]_{6,1} = 1940.3342,\ [A_k^l]_{13,1} = 669.3342,\ [A_k^l]_{11,3} = 59,\ [A_k^l]_{12,3} = 10,$ $[A_k^l]_{13,3} = 44,\ [A_k^l]_{10,4} = 0.5,\ [A_k^l]_{12,4} = 34,\ [A_k^l]_{13,4} = 9.5,\ [A_k^l]_{13,5} = 1,\ [A_k^l]_{8,7} = 22.6666,$ $[A_k^l]_{12,7} = 1.3334,\ [A_k^l]_{10,9} = 1.$

Then the monomial coefficient matrix can be written as

$$M = Y^l \cdot A_k^l = \begin{bmatrix} -10 & 0 & -10 & 10 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1271 & 0 & 54 & -1 & 0 & 0 & 24 & 0 & -1 & 0 & 0 & 0 & 0 \\ 669.3342 & 0 & -25 & -24 & 1 & 0 & -2.6667 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \quad (3.13)$$

The graph representation of this network can be seen in Fig. 3.1a. Alternative realizations containing minimal and maximal number of edges can be seen in Figs 3.1a and 3.1b, respectively. The reaction rates of the sparse realizations are depicted as edge weights in Figs 3.1a and 3.1b. In Fig. 3.1c, the structure of the reaction graph can be seen, while the non-zero off-diagonal elements of the corresponding Kirchhoff matrix A_k^d (containing 51 reactions) are

DOI:10.15774/PPKE.ITK.2014.002

the following:

$[A_k^d]_{2,1} = 679.6342,\ [A_k^d]_{3,1} = 0.1,\ [A_k^d]_{4,1} = 0.1,\ [A_k^d]_{5,1} = 0.1,\ [A_k^d]_{6,1} = 602.3658,\ [A_k^d]_{7,1} = 0.1,$
$[A_k^d]_{8,1} = 0.1,\ [A_k^d]_{9,1} = 0.1,\ [A_k^d]_{10,1} = 669.1342,\ [A_k^d]_{11,1} = 0.1, [A_k^d]_{12,1} = 0.1,\ [A_k^d]_{13,1} = 0.1,$
$[A_k^d]_{1,3} = 0.1,\ [A_k^d]_{2,3} = 0.1,\ [A_k^d]_{4,3} = 0.1,\ [A_k^d]_{5,3} = 44.6,\ [A_k^d]_{6,3} = 0.1,\ [A_k^d]_{7,3} = 0.1,$
$[A_k^d]_{8,3} = 0.1,\ [A_k^d]_{9,3} = 0.1,\ [A_k^d]_{10,3} = 0.1,\ [A_k^d]_{11,3} = 16.2,\ [A_k^d]_{12,3} = 9.3,\ [A_k^d]_{13,3} = 0.1,$
$[A_k^d]_{1,4} = 0.1,\ [A_k^d]_{2,4} = 0.1,\ [A_k^d]_{3,4} = 0.1,\ [A_k^d]_{5,4} = 9.6,\ [A_k^d]_{6,4} = 0.1,\ [A_k^d]_{7,4} = 0.1,$
$[A_k^d]_{8,4} = 0.1,\ [A_k^d]_{9,4} = 0.1,\ [A_k^d]_{10,4} = 0.1,\ [A_k^d]_{11,4} = 0.1,\ [A_k^d]_{12,4} = 24.4,\ [A_k^d]_{13,4} = 0.1,$
$[A_k^d]_{13,5} = 1,\ [A_k^d]_{1,7} = 0.1,\ [A_k^d]_{2,7} = 0.6,\ [A_k^d]_{3,7} = 0.1,\ [A_k^d]_{4,7} = 0.1,\ [A_k^d]_{5,7} = 0.1,$
$[A_k^d]_{6,7} = 0.1,\ [A_k^d]_{7,7} = -25.4, [A_k^d]_{8,7} = 23.2166,\ [A_k^d]_{9,7} = 0.1,\ [A_k^d]_{10,7} = 0.1,\ [A_k^d]_{11,7} = 0.1,$
$[A_k^d]_{12,7} = 0.68335,\ [A_k^d]_{13,7} = 0.1,\ [A_k^d]_{10,9} = 1.1,\ [A_k^d]_{13,9} = 0.1.$

3.1.2.2 Linear conjugacy

It is known from the literature that the kinetic property of a system of ODEs is generally preserved up to the re-ordering and positive scaling of the state variables [45]. Therefore, in [73] the notion of linear conjugacy was introduced, where two KRNs are called linearly conjugate if (in the case of appropriate initial conditions) there is a positive linear diagonal mapping between the solutions of the corresponding kinetic ODEs. In order to explain this, let us consider the kinetic systems defined by (Y, A_k) and (Y, A'_k):

$$\begin{aligned}\Sigma_1 :\ & \dot{x} = Y \cdot A_k \cdot \psi(x)\\ \Sigma_2 :\ & \dot{\bar{x}} = Y \cdot A'_k \cdot \psi(\bar{x}),\end{aligned}$$

where $x, \bar{x} \in \bar{\mathbb{R}}^n_+$, $Y \in \mathbb{R}^{n\times m}$ and $A_k, A'_k \in \mathbb{R}^{m\times m}$ are Kirchhoff matrices and $\psi(x) = [\psi_1(x)\ \psi_2(x)\ \dots\ \psi_m(x)]^T$ as it is defined in eq. (3.6). Now Σ_1 and Σ_2 are called as linearly conjugate kinetic systems if there is a vector $\mathbf{d} \in \mathbb{R}^n_+$ for which $T = \mathrm{diag}(\mathbf{d})$ and in case of $x(0) = T\bar{x}(0)$ the following holds:

$$x(t) = T\bar{x}(t)\ \forall t > 0.$$

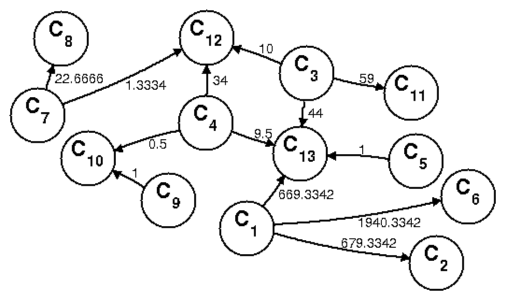
Let's assume that Σ_1 and Σ_2 are linearly conjugate. Than the following can be derived:

$$\dot{\bar{x}} = T^{-1}\dot{x} = T^{-1}YA_k\psi(x) = T^{-1}YA_k\psi(T\bar{x}) = T^{-1}YA_k \cdot \mathrm{diag}(\psi(\mathbf{d})) \cdot \psi(\bar{x}).$$

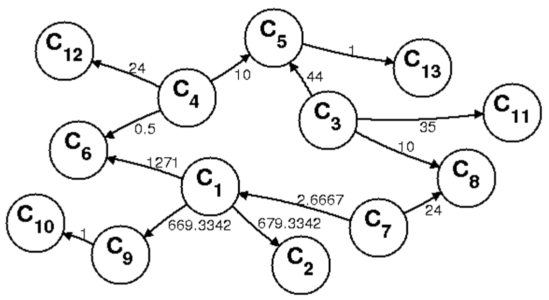
Considering this and the definition of Σ_2 we can conclude that

$$T^{-1}YA_k \cdot \mathrm{diag}(\psi(\mathbf{d})) = YA'_k,$$

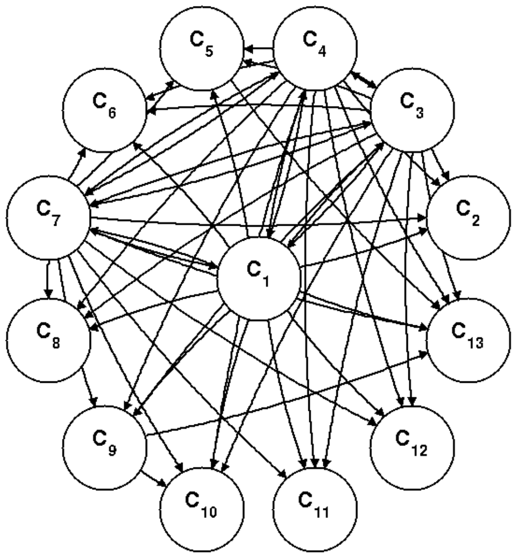
DOI:10.15774/PPKE.ITK.2014.002



(a) A possible KRN structure realizing eq. 3.12. The corresponding matrices are (Y^l, A_k^l). Reaction rates appear as edge weights.



(b) Another possible KRN structure containing minimal number of edges realizing eq. 3.12. Reaction rates appear as edge weights.



(c) A KRN structure containing maximal number of edges, realizing eq. 3.12. Edge weights are listed in A_k^d.

Figure 3.1: Alternative realizations of a KRN. C_i refers to the complex defined in the ith column of Y. Example appears in Subsection 3.1.2.1.

which leads us to the following result:

$$YA_k = TYA_k' \cdot (\text{diag}(\psi(\mathbf{d})))^{-1} = TYA_b, \tag{3.14}$$

where $A_b = A_k' \cdot (\text{diag}(\psi(\mathbf{d})))^{-1}$, so $A_k' = A_b \cdot \text{diag}(\psi(\mathbf{d}))$. It can be seen that A_b is a Kirchhoff matrix which is structurally equivalent with A_k' (meaning that the indices of the zero and non-zero elements are the same in these matrices), because A_b is computed from A_k' by multiplying its columns with positive scalar values.

From the above detailed results the following can be concluded. Let us consider a kinetic system denoted by Σ_1. Also, let's assume that there is a Kirchhoff matrix A_b and a vector $\mathbf{d} \in \mathbb{R}_+^n$ that $YA_k = TYA_b$, where $T = \text{diag}(\mathbf{d})$. Than, there exists a kinetic system $\bar{\Sigma}$

DOI:10.15774/PPKE.ITK.2014.002

described by $(Y, \bar{A}_k)$ such as Σ_1 and $\bar{\Sigma}$ are linearly conjugate and

$$\bar{A}_k = A_b \cdot \operatorname{diag}(\psi(\mathbf{d})). \tag{3.15}$$

Also, considering eq. (3.14), eq. (3.5) can be formulated as follows:

$$M = TYA_k \tag{3.16}$$

As it can be seen, linear conjugacy can be considered as an extension of dynamical equivalence. The two phenomena meet by taking the transformation to be the identity. It is also clear that the qualitative properties of the solutions (number and stability of equilibrium points, persistence/extinction of species, dimensions of invariant spaces etc.) of two linearly conjugate KRNs are always the same.

Example. In the following example originally appeared in [75] two linearly conjugate reaction networks are shown. The network is characterized by

$$Y = \begin{bmatrix} 1 & 2 & 2 & 2 & 1 & 2 & 1 & 2 & 1 & 1 \\ 2 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 2 & 1 \end{bmatrix}$$

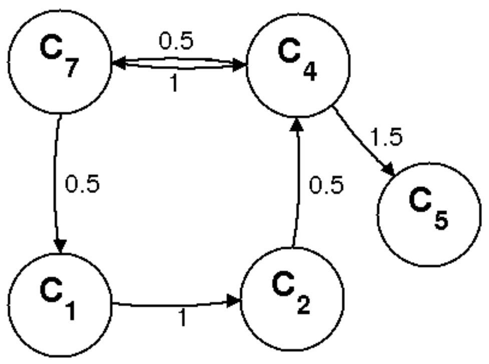
and the Kirchhoff matrix $A_k^{(1)}$ containing the following non-zero off-diagonal elements: $[A_k^{(1)}]_{2,1} = 1$, $[A_k^{(1)}]_{4,2} = 0.5$, $[A_k^{(1)}]_{5,4} = 1.5$, $[A_k^{(1)}]_{7,4} = 0.5$, $[A_k^{(1)}]_{1,7} = 0.5$, $[A_k^{(1)}]_{4,7} = 1$. The corresponding reaction graph can be seen in Fig. 3.2a.

A linearly conjugate realization can be described with the following non-zero off-diagonal elements in the matrix $A_k^{(2)}$: $[A_k^{(2)}]_{2,1} = 0.001$, $[A_k^{(2)}]_{4,2} = 0.005$, $[A_k^{(2)}]_{7,4} = 0.002$, $[A_k^{(2)}]_{1,7} = 0.005$, $[A_k^{(2)}]_{4,7} = 0.001$ and linear conjugacy matrix $T = diag([\frac{1}{0.001}, \frac{1}{0.001}, \frac{1}{0.004}])$. The corresponding reaction graph can be seen in Fig. 3.2b. For these values, the equation for linear conjugacy $Y \cdot A_k^{(1)} = T \cdot Y \cdot A_k^{(2)}$ holds (see eq. (3.16)).

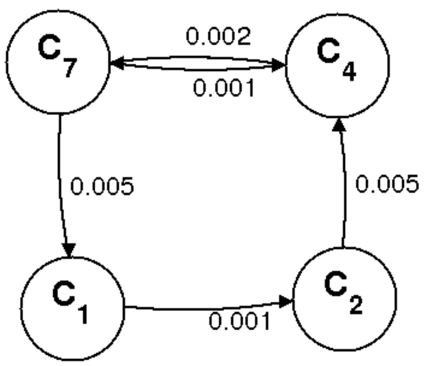
3.1.3 Known methods to compute dynamically equivalent KRNs

The original method of computing alternative realizations is based on the natural formulation of the problem as a MILP problem [25]. However, some alternative LP-based methods (see e.g. [119] or [135]) have also been published in the literature recently. We will shortly review these known solutions to be able to compare them with our new methods detailed in Section 3.2.

DOI:10.15774/PPKE.ITK.2014.002



(a) Reaction graph of the KRN described by $(Y, A_k^{(1)})$.



(b) Reaction graph of the KRN described by $(Y, A_k^{(2)})$.

Figure 3.2: Linearly conjugate realizations of a KRN. C_i refers to the complex defined in the ith column of Y. Isolated complexes are omitted from the figure. Reaction rates appear as edge weights. Example appears in Subsection 3.1.2.2.

3.1.3.1 Computing dynamically equivalent realizations with MILP

Let us formulate the KRN alternative realization problem as an MILP. This formulation has also a crucial role in one of the new LP-based methods presented later on. We will represent the Kirchhoff matrix defined in Subsection 3.1.1 as:

$$A_k = \begin{bmatrix} -a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & -a_{22} & & a_{2m} \\ \vdots & & & \vdots \\ a_{m1} & a_{m2} & \dots & -a_{mm} \end{bmatrix}. \tag{3.17}$$

The Kirchhoff property of A_k can be expressed by the following linear constraints:

$$\sum_{i=1}^{m} [A_k]_{i,j} = 0,\ j = 1, \dots, m \tag{3.18}$$

$$[A_k]_{i,j} \geq 0, \quad i,j = 1, \dots, m,\ i \neq j \tag{3.19}$$

Let us call eq. (3.9) and eqs. (3.18)-(3.19) altogether as *kinetic constraints* because they clearly characterize the kinetic system structure of the model, and they can be considered as a constraint set in an optimization problem.

To formulate a MILP structure, where the real-valued decision variables correspond to the off-diagonal elements of A_k, the following additional bounds for these elements are given:

$$[A_k]_{i,j} \leq l_{i,j}, \quad i,j = 1, \dots, m,\ i \neq j \tag{3.20}$$

$$l_{i,i} \leq [A_k]_{i,i}, \quad i = 1, \dots, m. \tag{3.21}$$

DOI:10.15774/PPKE.ITK.2014.002

where $l_{i,j}$ for $i, j = 1, \dots, m$ are appropriately chosen bounds (with sufficiently large absolute values). Then the nonzero property of the individual reaction rate coefficients can now be written as

$$\delta_{i,j} = 1 \leftrightarrow [A_k]_{i,j} > \epsilon, \quad i, j = 1, \dots, m, \ i \neq j. \tag{3.22}$$

where $\delta_{i,j}$ are binary decision variables, $\leftrightarrow$ denotes the '*if and only if*' relation, and ϵ is a small positive number used for distinguishing between practically zero and non-zero values.

The linear constraints in eq. (3.22) can be translated to the following linear inequalities [25]

$$0 \leq [A_k]_{i,j} - \epsilon\delta_{i,j}, \quad i, j = 1, \dots, m, \ i \neq j \tag{3.23}$$

$$0 \leq -[A_k]_{i,j} + l_{i,j}\delta_{i,j}, \quad i, j = 1, \dots, m, \ i \neq j. \tag{3.24}$$

From now, the constraint matrix can be formulated (see eq. (2.3)), while the minimization or maximization of the objective function

$$c(\delta) = \sum_{\substack{i,j=1 \\ i \neq j}}^{m} \delta_{i,j} \tag{3.25}$$

leads to the computation of sparse or dense realizations, respectively.

3.1.3.2 Computing sparse realizations with an iterative LP-based technique

In [135] an LP-based algorithm is presented to search for sparse linear models of gene regulation networks. The algorithm uses an iterative method to approximate the elements of the A_k matrix. The approach is quite efficient: as it is mentioned in [135], the method usually doesn't need more than a few (less than 10) iterations to converge.

We briefly describe how to apply this method for the computation of sparse KRN realizations. Let A_k^p denote the A_k in the pth step. Firstly set all off-diagonal elements of A_k^0 to 1. Let us define a weight $w_{i,j}^p$ for each off-diagonal element of A_k^p and initialize them as $w_{i,j}^0 = 1$ for each $i, j = 1, \dots, m, \ i \neq j$. In the pth iteration step, let us recalculate the weights $w_{i,j}^p$ as follows:

$$w_{i,j}^p = \frac{\beta}{\beta + |[A_k]_{i,j}^{p-1}|} \text{ for } i, j = 1, \dots m, \ i \neq j \tag{3.26}$$

for an appropriate $\beta > 0$ value. The off-diagonal elements of A_k^p are obtained by solving a linear programming problem constrained by eq. (3.9) and eqs. (3.18)-(3.70) with the following

DOI:10.15774/PPKE.ITK.2014.002

objective function:

$$J = \min \sum_{\substack{i,j=1 \\ i \neq j}}^{m} w^p_{i,j} [A_k]^p_{i,j} \tag{3.27}$$

The incorporated *kinetic constraints* (see eqs. (3.18)-(3.19)) ensure that in each iteration the obtained matrix A^p_k represents a dynamically equivalent realization and it has the Kirchoff property. The algorithm repeats eq. (3.27)-(3.26) until the objective value J will not change significantly between two successive steps. At the end of the iteration process the algorithm returns with A^p_k obtained in the last iteration step as a sparse realization of the network. In the following, we will refer to this algorithm as *Iterative LP*.

3.1.3.3 Computing dense realizations using multiple LP steps

This method was published in [119] and it can find dense realizations in polynomial time. The method uses $m \cdot (m-1)$ LP computation steps to generate the result. We will use the method for comparison with our new approach, therefore we summarize it for convenience (more details can be found in [119]).

For each $p, q = 1, \dots m,\ p \neq q$, solve the optimization problem:

$$\max c_{pq} = [A_k]_{p,q} \tag{3.28}$$

with respect to the eq. (3.9) and eqs. (3.18)-(3.19) together with appropriate bounds eq. (3.20)-(3.21).

The reaction $C_q \to C_p$ is present in the dense realization only if $\max c_{pq} > 0$. Let us denote the solution corresponding to $(p,q),\ p \neq q$ by $\bar{A}^{pq}_k$. Now we will use these solutions to compute the final optimization step to generate the dense realization using the following quantities:

$$\epsilon_{ij} = \left[\frac{1}{m(m-1)} \sum_{\substack{p,q=1 \\ p \neq q}}^{m} \bar{A}^{pq}_k \right]_{i,j},\ i \neq j \tag{3.29}$$

As we can see, $\epsilon_{i,j} \geq 0\ \forall i \neq j$ and $\epsilon_{i,j} > 0$ *iff* the reaction $C_j \to C_i$ is in the dense realization. By solving the following LP feasibility problem for A_k with arbitrary linear cost function, the dense realization can be obtained:

DOI:10.15774/PPKE.ITK.2014.002

$$\begin{cases} Y \cdot A_k = M & \\ \sum_{i=1}^{m} [A_k]_{i,j} = 0 & j = 1, \dots m \\ \epsilon_{i,j} \leq [A_k]_{i,j} \leq UB_{i,j} & i, j = 1, \dots m, \ i \neq j \\ [A_k]_{i,i} \leq 0 & i = 1, \dots m \end{cases} \tag{3.30}$$

where $UB_{i,j}$ are proper upper bounds with sufficiently large positive value. In the following, we will refer to this algorithm as *Element-wise LP*.

3.1.4 Weak reversibility

The existence of a dynamically equivalent or linearly conjugate weakly reversible KRN realization can be useful in the analysis of the qualitative dynamical properties of the system [74]. This property has a crucial role in the theory of KRNs, since it connects structural properties of the reaction graph to qualitative features of the dynamical behavior of the reaction network which is especially useful in the deficiency zero and deficiency one cases. The deficiency of a reaction network can be defined as follows [47, 48]: for each reaction $C_i \to C_j$ in the network we can define the *reaction vector* $e_k = [Y]_{\cdot,j} - [Y]_{\cdot,i}$, $k = 1, ..., r$ where $[Y]_{\cdot,i}$ stands for the ith column of matrix Y. From the set of the reaction vectors the *stoichiometric subspace* can be defined as $S = span\{e_1, ..., e_r\}$. The positive *stoichiometric compatibility class* containing a concentration x_0 is the following set: $(x_0 + S) \cap \mathbb{R}^n_+$ where $\mathbb{R}^n_+$ denotes the positive orthant in $\mathbb{R}^n$. The deficiency δ of a reaction network is calculated as: $\delta = m - l - rank(\{e_1, ..., e_r\})$, where m is the number of the complexes, l is the number of linkage classes in the reaction graph. As it is formulated in the *Deficiency Zero Theorem* [46] for a KRN having zero deficiency and a weakly reversible structure, there exists precisely one asymptotically stable equilibrium point in each stoichiometric compatibility class. According to the *Boundedness conjecture* for which no counterexamples have been found, the solutions of any weakly reversible KRN are bounded. The conjecture was proved in [11] for the single linkage class case. Moreover, there exist important general results about the existence of equilibrium points in weakly reversible reaction networks [37, 22].

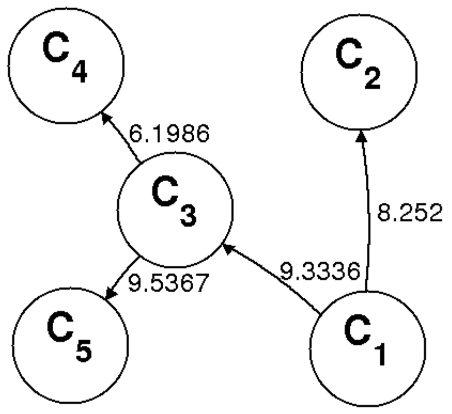
From a graph-theoretic point of view, weak reversibility holds if and only if all components (i.e. linkage classes) of the reaction graph are strongly connected components (i.e. if there exists a directed path between nodes C_i and C_j then there exists a directed path from C_j to C_i).

As an example, let us consider a KRN having the following complex set:

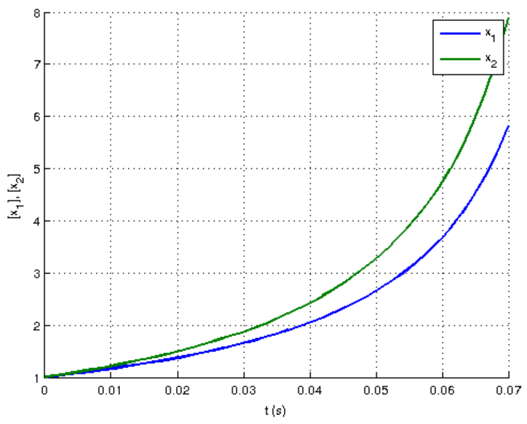
$$Y = \begin{bmatrix} 1 & 2 & 1 & 2 & 1 \\ 0 & 0 & 1 & 1 & 2 \end{bmatrix} \tag{3.31}$$

and a reaction graph depicted in Fig. 3.3a. The dynamics of this reaction network can be seen in Fig. 3.3b. Later on, the reaction graph has been extended to contain addition edges

DOI:10.15774/PPKE.ITK.2014.002

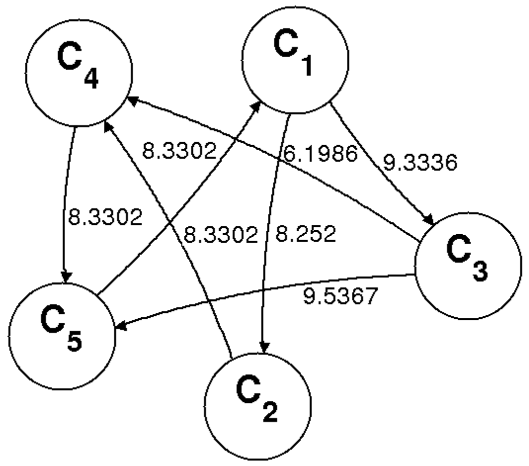


(a) Reaction graph.

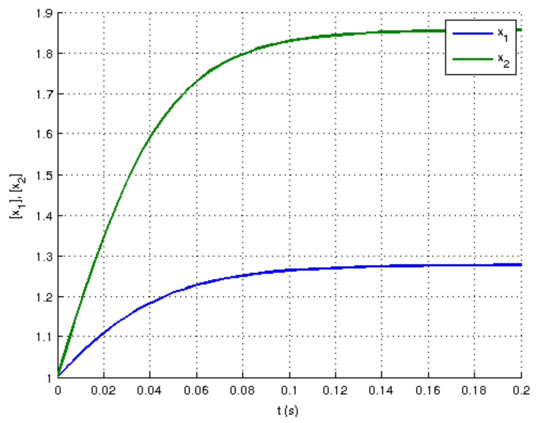


(b) Dynamics of the system.

Figure 3.3: Structure and dynamics of a reaction network. Complexes are from eq. (3.31).



(a) Reaction graph.



(b) Dynamics of the system.

Figure 3.4: Structure and dynamics of a weakly reversible reaction network obtained by extending a non-reversible network (see Fig. 3.3a.). Complexes are from eq. (3.31).

as it can be seen in Fig. 3.4a. The obtained network is weakly reversible. The corresponding (stable) dynamics can be seen in Fig. 3.4b.

Additionally, it is known that a KRN with a Kirchhoff matrix A_k is weakly reversible if and only if there exists a strictly positive vector in the kernel of A_k [74], i.e.

$$A_k \cdot \mathbf{h} = 0 \tag{3.32}$$

$$h_j > 0, \quad j = 1, \dots, m \tag{3.33}$$

where $\mathbf{h} = [h_1 \ \dots \ h_m]^T$.

For simplicity, we introduce the following notions. A Kirchhoff matrix is called weakly reversible if the corresponding reaction graph is weakly reversible. A vector $\mathbf{p} \in \mathbb{R}^n$ is called

DOI:10.15774/PPKE.ITK.2014.002

strictly positive if it is element-wise strictly positive, i.e. $p_i > 0$ for $i = 1, \dots, n$. Two $n \times n$ matrices V and W are called structurally equal if the following is fulfilled: $V_{i,j} \neq 0$ if and only if $W_{i,j} \neq 0$ for $i, j = 1, \dots, n$. (Therefore, two structurally equal Kirchhoff matrices encode the same unweighted reaction graph structure.)

It can be seen that eq. (3.32) itself is a nonlinear constraint if both A_k and $\mathbf{h}$ are unknowns. In order to formulate it as set of linear constraints, we can introduce a scaled Kirchhoff matrix $\tilde{A}_k$ as follows:

$$[\tilde{A}_k]_{i,j} = [A_k]_{i,j} \cdot b_j, \;\; i, j, = 1, \dots, m \tag{3.34}$$

Now, A_k corresponds to a weakly reversible KRN if and only if $\mathbf{1}^{(m)} = [1 \; 1 \; \dots \; 1]^T \in \mathbb{R}^m$ is an element of $\ker(\tilde{A}_k)$. Moreover, it is trivial that $\tilde{A}_k$ is weakly reversible if and only if the original Kirchhoff matrix A_k is also a weakly reversible one. Based on these facts, the linear constraint set for weak reversibility can be formulated as follows:

$$\begin{aligned}
&\sum_{i=1}^{m} [\tilde{A}_k]_{i,j} = 0, \quad j = 1, \dots, m \\
&\sum_{i=1}^{m} [\tilde{A}_k]_{j,i} = 0, \quad j = 1, \dots, m \\
&[\tilde{A}_k]_{i,j} \geq 0, \quad i, j = 1, \dots, m, \; i \neq j,
\end{aligned} \tag{3.35}$$

where A_k and $\tilde{A}_k$ are structurally equal.

3.1.4.1 The dense weakly reversible realization forms a super-structure for a fixed complex set

It was shown in [118] that the dense realization is a unique superstructure containing all mathematically possible reactions. In the following, we will prove that a dense and weakly reversible realization contains all possible weakly reversible realizations if the complex set is fixed.

Theorem 1. *Consider a kinetic system $\Sigma : \dot{\mathbf{x}} = M \cdot \psi(\mathbf{x})$. Suppose that (Y, A_k) is a weakly reversible dynamically equivalent realization of Σ that contains the maximal number of nonzero elements in A_k. Then, for any weakly reversible Kirchhoff matrix A'_k for which $Y \cdot A_k = Y \cdot A'_k$ the following holds: $[A'_k]_{i,j} > 0$ implies $[A_k]_{i,j} > 0$ for any $i \neq j$.*

Proof. (by contradiction) Consider a weakly reversible Kirchhoff matrix A'_k for which $Y \cdot A_k = Y \cdot A'_k$. Suppose that there exists $1 \leq i, j \leq m$, $i \neq j$ for which $[A'_k]_{i,j} > 0$, but $[A_k]_{i,j} = 0$. Let us define the matrix $\tilde{A}_k$ as

$$\tilde{A}_k = \frac{A_k + A'_k}{2} \tag{3.36}$$

Clearly, $Y \cdot A_k = Y \cdot \tilde{A}_k$, and $[\tilde{A}_k]_{i,j} > 0$. It follows from the weak reversibility of A_k that there exists a strictly positive vector $\mathbf{p} \in \mathbb{R}^m$ such that $A_k \cdot \mathbf{p} = 0$. Similarly, there exists a strictly positive vector $\mathbf{p'}$ in the kernel of A'_k, too. Let us define the following scaled Kirchhoff

DOI:10.15774/PPKE.ITK.2014.002

matrices: $\bar{A}_k = A_k \cdot \mathrm{diag}(\mathbf{p})$, $\bar{A}'_k = A'_k \cdot \mathrm{diag}(\mathbf{p}')$. Then $\bar{A}_k$ and $\bar{A}'_k$ are Kirchhoff, and they are structurally equal to A_k and A'_k, respectively. Moreover, $\bar{A}_k \cdot \mathbf{1}^{(m)} = \bar{A}'_k \cdot \mathbf{1}^{(m)} = 0$, where $\mathbf{1}^{(m)}$ denotes the m dimensional column vector composed of ones. Let $\hat{A}_k = \bar{A}_k + \bar{A}'_k$. Then $\hat{A}_k$ is a weakly reversible Kirchhoff matrix, since $\hat{A}_k \cdot \mathbf{1}^{(m)} = (\bar{A}_k + \bar{A}'_k) \cdot \mathbf{1}^{(m)} = 0$. It is also clear that $\hat{A}_k$ is structurally equal to $\tilde{A}_k$. This implies that $\tilde{A}_k$ is a weakly reversible Kirchhoff matrix containing more non-zero off-diagonal elements than A_k, which is a contradiction. □

To briefly illustrate the above theorem, consider a kinetic system $\Sigma : \dot{\mathbf{x}} = M \cdot \psi(\mathbf{x})$ characterized by the following matrices:

$$Y = \begin{bmatrix} 1 & 2 & 1 & 1 \\ 2 & 1 & 3 & 1 \end{bmatrix}, \quad M = \begin{bmatrix} 0 & -2 & 0 & 2 \\ -3 & 2 & -2 & 0 \end{bmatrix} \tag{3.37}$$

This kinetic system was studied before in [73] and in [123]. A possible dense weakly reversible realization $(Y, A_k^{(1)})$ of Σ is given by the Kirchhoff matrix:

$$A_k^{(1)} = \begin{bmatrix} -3.2 & 1.8 & 0.1 & 0 \\ 0 & -2 & 0 & 2 \\ 0.1 & 0.1 & -1.05 & 0 \\ 3.1 & 0.1 & 0.95 & -2 \end{bmatrix}. \tag{3.38}$$

On the other hand, the following Kirchhoff matrix encodes a complex balanced and thus weakly reversible realization $(Y, A_k^{(2)})$ of Σ (see [123]):

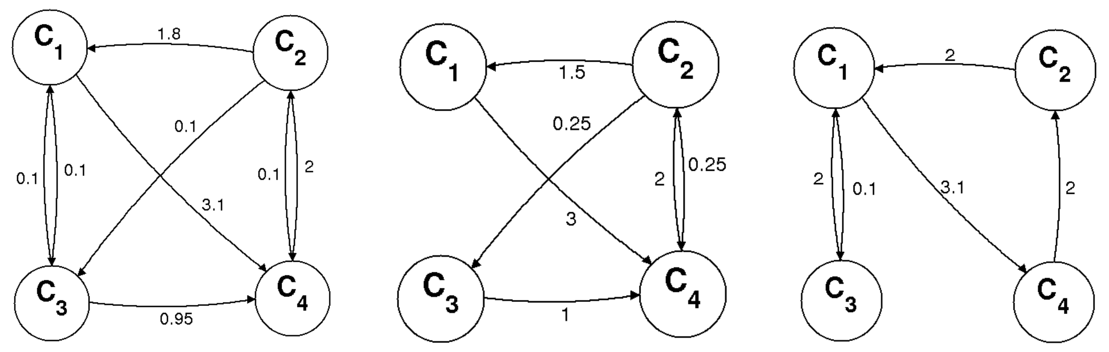
$$A_k^{(2)} = \begin{bmatrix} -3 & 1.5 & 0 & 0 \\ 0 & -2 & 0 & 2 \\ 0 & 0.25 & -1 & 0 \\ 3 & 0.25 & 1 & -2 \end{bmatrix}. \tag{3.39}$$

Finally, a sparse weakly reversible realization containing only 5 reactions $(Y, A_k^{(3)})$ is defined by:

$$A_k^{(3)} = \begin{bmatrix} -3.2 & 2 & 2 & 0 \\ 0 & -2 & 0 & 2 \\ 0.1 & 0 & -2 & 0 \\ 3.1 & 0 & 0 & -2 \end{bmatrix}. \tag{3.40}$$

It can be easily checked that all $(Y, A_k^{(1)})$, $(Y, A_k^{(2)})$ and $(Y, A_k^{(3)})$ are dynamically equivalent weakly reversible realizations of Σ. The reaction graph structures of the three realizations are depicted in Fig. 3.5. It is clearly visible from the figure that the unweighted reaction graphs of $(Y, A_k^{(2)})$ and $(Y, A_k^{(3)})$ are indeed proper subgraphs of the unweighted reaction graph of $(Y, A_k^{(1)})$.

DOI:10.15774/PPKE.ITK.2014.002



(a) Dense weakly reversible structure defined by $A_k^{(1)}$.

(b) Weakly reversible structure of a complex balanced realization given by $A_k^{(2)}$.

(c) Sparse weakly reversible structure with 5 reactions defined by $A_k^{(3)}$.

Figure 3.5: Possible dynamically equivalent weakly reversible reaction graph structures of the kinetic system eq. (3.37).

3.1.5 Known methods to compute dynamically equivalent, weakly reversible KRNs

In this Subsection a short review of several different existing algorithms to find dynamically equivalent weakly reversible realizations of a KRN can be found. Besides of two MILP-based methods, an LP-based method has been described, too. Later on, these methods will be compared in terms of computational time to the proposed LP-based methods (see Sec. 3.3.2).

3.1.5.1 One-step MILP procedure to compute weakly reversible realizations with additional preferred structural properties

First, we briefly recall the part ensuring the structural equality of A_k and $\tilde{A}_k$ of the algorithm presented in [75]. In that paper, a set of boolean decision variables are introduced and used as follows:

$$[A_k]_{i,j} > \epsilon \leftrightarrow [\tilde{A}]_{i,j} > \epsilon, \quad i,j = 1,\dots,m, \quad i \neq j, \tag{3.41}$$

where again, $\leftrightarrow$ means the '*if and only if*' relation from classical binary logic and ϵ is a small fixed positive threshold value to distinguish between practically zero and nonzero edge weights in the reaction network. This logical condition can be expressed in the form of equivalent linear constraints (for the general framework, see e.g. [103]):

$$\begin{aligned}
&0 \leq [A_k]_{i,j} - \epsilon\delta_{i,j}, \quad i,j = 1,\dots,m, \quad i \neq j,\\
&0 \leq -[A_k]_{i,j} + p_u \cdot \delta_{i,j}, \quad i,j = 1,\dots,m, \quad i \neq j.\\
&0 \leq [\tilde{A}_k]_{i,j} - \epsilon\delta_{i,j}, \quad i,j = 1,\dots,m, \quad i \neq j\\
&0 \leq -[\tilde{A}_k]_{i,j} + p_u \cdot \delta_{i,j}, \quad i,j = 1,\dots,m, \quad i \neq j,
\end{aligned} \tag{3.42}$$

where $\delta_{i,j}$, $i, j = 1, \ldots, m$, $\quad i \neq j$ are boolean decision variables and p_u is the upper bound for the elements of A_k and $\tilde{A}_k$.

Due to the introduction of the δ variables, the resulting problem could be solved in the framework of MILP, making the handling of larger networks difficult. On the other hand, these boolean variables can be used to keep track of the presence of individual reactions, and by minimizing or maximizing the sum $\sum_{i,j=1}^{m} \delta_{i,j}$ in the objective function of the optimization problem, a sparse or dense realization (containing the maximal or minimal number of reactions, respectively) can be obtained [118]. It should be noted that this method do not scale up very well with the size of the network due to the integer variables in the optimization.

3.1.5.2 Graph-theory inspired, iterative procedure to find dense weakly reversible realizations

In [123] a completely different, graph-theory motivated method is presented (only for the case of dynamical equivalence) which leads to an iterative MILP-based algorithm. The algorithm requires Y and M as inputs, and computes an initial dense realization for them. Afterwards, it determines all the edges in the network that connect different strong components. In the next step, by solving an MILP problem a valid dense realization is determined without these edges (if it exists). These steps are repeated until the reaction graph of the resulting KRN is found to be weakly reversible. The algorithm ends with failure if there does not exist any dynamically equivalent realization that does not contain the directed edges to be excluded.

In the original publication, an MILP problem is solved in each iteration step. Fortunately, as it is described in [1], the MILP problem for the computation of dense reaction structures can be safely replaced with a purely LP-based algorithm. This enables us to significantly speed up the solution process of the original method published in [123]. Therefore, we implemented and used the LP-based modified version of this graph-theory inspired algorithm for the present work to compare its performance to our new methods. This algorithm will be shortly called the *graph-based-LP* method in this work.

3.1.6 Mass conservation in KRNs

During the mathematical generalization of mass-action KRN systems the mass conservation assumptions which are present in classical chemical theory were usually omitted, because this property is not essential for the kinetic representation of general nonlinear systems. However, mass conservation is an important constraint when modeling physically plausible reaction sets [61].

In [61] (stoichiometric) *mass conservation* is formulated as follows. Let us define g_v as the scaled molecular weight of the species X_v with strictly positive value. If reaction $C_i \to C_j$ is

DOI:10.15774/PPKE.ITK.2014.002

present in the network, the following can be written:

$$\sum_{v=1}^{n} \alpha_{vi} g_v = \sum_{v=1}^{n} \alpha_{vj} g_v = c_s. \tag{3.43}$$

where c_s is a strictly positive scalar value. Let us define vector $g \in \mathbb{R}_+^n$ as a row vector formulated from the scaled molecular weights. Now eq. (3.43) can be rewritten as $g \cdot Y(\cdot, i) = g \cdot Y(\cdot, j) = c_s$ where $Y(\cdot, i)$ refers to the ith column of matrix Y. Finally, it can be said that a reaction is mass conservative if the following holds:

$$g \cdot \rho^{(i,j)} = 0 \tag{3.44}$$

where $\rho^{(i,j)} = Y(\cdot, i) - Y(\cdot, j)$ is the *reaction vector* and g is strictly positive. In [61] the *mass conservative reaction set* is defined as a set of the reactions having the above property. It should be noted that a given KRN is called mass conservative if all of its reactions are in the mass conservative reaction set and a common strictly positive g can be determined for them.

In Section 3.4, an algorithm will be defined to compute dynamically equivalent realizations which are mass conservative in the above defined sense.

3.2 Finding dynamically equivalent realizations with LP

In Subsection 3.1.2 it has been mentioned that in case of a specific KRN with given dynamics and fixed complex set, different (Y, A_k) pairs can fulfill eq. (3.9). Based on these, one can define structural properties such as minimal or maximal number of nonzero elements in matrix A_k and search for alternative representations of the given network which fulfill these criteria. In this particular case, we want to determine the sparse and dense realizations denoted by (Y^s, A_k^s) and (Y^d, A_k^d), respectively.

Using the 3-dimensional Lorenz system introduced in Subsection 3.1.2, the validation of the proposed methods on a small-scale KRN has been completed. The comparative performance analysis of the presented methods on large random networks is detailed in Subsection 3.2.3. Let us note that it is known from [128] that this system can be represented by KRNs having 3 species and 13 complexes with sparse realizations containing 13 off-diagonal non-zero elements, and its dense realization contains 51 off-diagonal non-zero elements.

In this Section, the results corresponding to Thesis I. (see 4.3.3) are detailed.

3.2.1 Finding sparse realizations

The results in [38] show that in the case of large size, under-determined system of linear equations (even in the case of non-negative decision variables) formulated as $Ax = b$, the l^1-norm based minimization can produce a sparse solution out of the infinitely many possible solutions of the problem. Furthermore, in [39] it has been shown that similar technique can be used to find sparse nonnegative solutions for these kind of problems. In this context, sparse

DOI:10.15774/PPKE.ITK.2014.002

solution means that the given x vector contains the minimal number of non-zero elements. In other words, under specific conditions, the combinatorial optimization problem of finding the sparse realization can be efficiently approached as a convex optimization problem. This is possible if the solution vector x with length n is sparse enough: it should not have more than $\rho \cdot n$ nonzero elements. In [38] an empirical limit $\rho = 0.3$ is suggested, we also used this value.

The l^1 minimization method is applied in our case as follows. Recall that in a given KRN there are n species and m complexes. The equality constraints of the optimization problem are the *kinetic constraints* (see eq. (3.9) and eqs. (3.18)-(3.70)). For a single column of A_k containing m elements, the number of kinetic constrains will be $n+1$. Hence, in case of $n+1 < m$ the emerging equality-type constraints as a system of linear equations remains under-determined. Therefore, a column-wise l^1-norm based minimization is completed and the resulting vectors are considered as the column vectors of the sparse realization of the KRN:

$$\min \sum_{i=1}^{m} abs([A_k]_{i,j}) \ for \ \forall j = 1, \ldots m, \ i \neq j \tag{3.45}$$

Let us denote the ratio of non-zero and zero elements in $[A_k]_{\cdot,j}$ as τ. If $\tau < \rho \cdot m$, then the minimization successfully finds the sparse solution. In the following, we will refer to this algorithm as the l^1-norm based algorithm.

Computing the sparse realization of the 3-dimensional Lorentz system took 17.172 seconds in case of the MILP based method. *Iterative LP* were able to find a valid sparse realization in 0.0144 seconds while the proposed l^1-norm based sparse search can complete this task in 0.0166 seconds.

The above presented results are summarized in Thesis I.a. Detailed computational results can be found in Subsection 3.2.3. These results were originally published in [1].

3.2.2 Finding dense realizations

The proposed method is based on the construction of the MILP problem formulation given in Subsection 3.1.3. The main idea was to formulate the problem by relaxing the integrality constraints and then solve the remaining LP problem. By maximizing the sum of the relaxed auxiliary variables, the number of directed edges (i.e. the number of non-zero off-diagonal elements of A_k) is maximized, too, in the reaction graph of the KRN.

The constraint matrices were built up again using the *kinetic constraints* (eqs. (3.18)-(3.19)). Similarly to the MILP case, a set of auxiliary variables were defined: a σ_i variable for each decision variable a_i (recall that each decision variable represents an off-diagonal element of the A_k matrix). All these auxiliary variables are real valued. The constraints keeping track of nonzero reaction rate coefficients are given by

$$\sigma_i = 1 \longleftrightarrow a_i > \epsilon_e \tag{3.46}$$

where ϵ_e is a minimal (naturally positive) edge weight under which the edge is excluded from

DOI:10.15774/PPKE.ITK.2014.002

the network (i.e. the corresponding reaction rate coefficient is considered zero). After the relaxation of the integrality constrains the relation in eq. (3.46) becomes:

$$\epsilon \cdot \sigma_i \leq a_i \tag{3.47}$$

where $\sigma_i \in [0;\, 1]$ and $\epsilon > 0$ is a sufficiently small number, but $\epsilon > \epsilon_e$. By considering ϵ as a scaling factor, after short reformulation we obtain:

$$\begin{aligned} -a_i + \sigma_i &\leq 0 \qquad (3.48)\\ \sigma_i &\in [0;\, \epsilon] \qquad (3.49) \end{aligned}$$

where ineq. (3.48) is formulated as a constraint and eq. (3.49) is formulated as lower and upper bounds in the LP problem.

Now the optimization problem is defined as follows:

$$\max \sum_{i=1}^{m} \sigma_i \tag{3.50}$$

$$\begin{cases} Y \cdot A_k = M & \\ \sum_{i=1}^{m} [A_k]_{i,j} = 0 & j = 1, \dots m \\ 0 \leq \sigma_i \leq \epsilon & i = 1, \dots m \\ -a_i + \sigma_i \leq 0 & i = 1, \dots m \end{cases} \tag{3.51}$$

Let us show that the solution of the above optimization problem indeed determines the dense network: suppose that a given edge (corresponding to the decision variable a_i) can be present in the network which means that there is no such kinetic constraint which forbids that a_i could be larger than zero. In this case, according to the constraint formulated in ineq. (3.48), $\sigma_i > 0$ will occur which yields $a_i \geq \epsilon$. Because of $\epsilon > \epsilon_e$, the edge will be present in the network. In the opposite case, when the edge represented by a_i should be excluded from the network according to the kinetic constraints, $a_i = 0$ leading to $\sigma_i = 0$ according to ineq. (3.48). As a result of the maximization, a dense realization with edge weights larger than ϵ_e is obtained. In the following, we will refer to this algorithm as the *LP-MAX* algorithm.

Computing the dense realization of the 3-dimensional Lorentz system was also successfully computed by all three algorithms. The original MILP based algorithm completed it in 5.3477 seconds, *Element-wise LP* consumed 0.2524 seconds, and the *LP-MAX* algorithm needs 0.0446 seconds to compute the solution.

The above presented results are summarized in Thesis I.b. Detailed computational results can be found in Subsection 3.2.3. These results were originally published in [1].

DOI:10.15774/PPKE.ITK.2014.002

3.2.3 Comparative study of the presented algorithms

Both the l^1-norm based search and the *LP-MAX* methods were involved in a comparative study in which extensive simulations were completed to evaluate the performance of the proposed methods. Large scale problems were investigated to present the capability of dealing with biologically relevant problems, too. The MILP-based methods and the LP-based methods presented in Subsection 3.1.3 were used as a basis of comparison. These results were originally published in [1].

To solve the LP problems, we used the CLP solver [138] from the COIN-OR community. The formulated MILP problems were solved by the DIP solver [52] from the COIN-OR community [139] because it can solve these type of problems very efficiently. All the simulations were done on a 8-core 3.0GHz computer. The computational problems were handled in MATLAB environment with the help of the CRNReals Toolbox [117] and YALMIP Toolbox [82].

All the methods were tested on several large, randomly generated KRNs to be able to test their performance and time consumption of the solution. The methodology of generating the random kinetic systems was the following. n as the number of species and m as the number of complexes were defined as parameters. First, a polynomial representation of the system with the given parameters was generated. Then a kinetic polynomial system of the form eq. (3.10) was generated where the elements of M were uniformly distributed random real numbers from the interval [10, 110]. The exponents of the monomials of ψ were chosen as uniformly distributed random integers from the interval [0, 5]. This kinetic polynomial system was converted to a so-called *canonical* KRN representation (Y, A_k) as it is described in [62].

Altogether more than 100 different scenarios were used to examine the performance of the proposed algorithms. The networks used during these tests were different both in their structure and in their size. In the following, the numerical results of some scenarios are presented. More results can be found in an electronic supplement at
`http://daedalus.scl.sztaki.hu/PCRG/works/CRN_Alter_Struct.zip`.

In all cases the following numerical values were used: reactions having reaction rate smaller than 10^{-6} were handled as zero in the networks. Let us consider two different realizations of the same KRN, namely (Y, A_k^1) and (Y, A_k^2) and define the R matrix as follows: $R = \text{abs}(Y \cdot A_k^1 - Y \cdot A_k^2)$. These two realizations were considered as dynamically equivalent representations if $R < 10^{-3}$.

In Table 3.1, the summary of the search for the sparse realization can be found. Each row represents a different problem size: the number of species was fixed to 10 but the number of complexes was increased to enlarge the computational task. In the first row block the number of off-diagonal non-zero elements can be found in the resulting A_k matrices. One can see the match in the values which implies that all the algorithms successfully found the realization containing the minimal number of reactions. In the second row block, the running

DOI:10.15774/PPKE.ITK.2014.002

Number of complexes	Number of reactions in the sparse realization		
	l^1-norm based	MILP-based	*Iterative LP*
220	200	200	200
330	300	300	300
440	400	400	400
550	500	500	500
	Computational time (s)		
220	0.49	603.32	29.16
330	1.55	1597.21	134.17
440	3.35	2784.53	447.85
550	4.22	4330.62	1028.59

Table 3.1: Comparing the LP- and MILP-based methods while searching for the sparse realization. The size of the computational problem grows as the number of complexes increases.

Number of complexes	Number of reactions in the dense realization		
	LP-MAX	MILP-based	*Element-wise LP*
220	4380	4380	4380
330	10528	10528	10528
440	18438	18438	18438
550	30936	30936	30936
	Computational time (s)		
220	6.25	1413.16	156.41
330	35.53	2251.98	703.77
440	66.71	3828.73	1736.76
550	304.75	5422.87	4093.58

Table 3.2: Comparing the LP- and MILP-based methods while searching for the dense realization. The size of the computational problem grows as the number of complexes increases.

time of the algorithms can be seen while evaluating the previously presented scenarios. In Table 3.2, the result of the search for the dense superstructure is summarized. The structure of the table is similar to Table 3.1. To summarize the above presented results we can say that the presented new, LP-based algorithms successfully compute both the dense and sparse realizations while outperform all the previously presented methods. With the help of these methods the computation of the alternative realizations of biologically relevant, large size networks can be done efficiently.

3.2.4 Possible parallelization of the proposed algorithms

As the size of the emerging computational problem grows during the search for the alternative realizations, the need for possible parallelization is increasing. Fortunately, as

DOI:10.15774/PPKE.ITK.2014.002

the reader can notice, all the proposed algorithms can be parallelized easily because all the optimization problems are built up by taking the A_k matrix column-wise. Moreover all the results of the optimizations are incorporated into the final result independently of each other.

This means that in case of having an A_k matrix with size $m \times m$ the search for the alternative realizations (both in case of sparse and dense realizations) can be split up into m independent, parallelly solvable problems. This gives the opportunity to gain even more speedup in the solution process.

3.2.5 Finding sparse/dense realizations in case of large kinetic reaction networks

As it was mentioned before, we would like to use our methods to study the possible structures of large scale, biologically relevant networks. As a case study the *ErbB* network described in [29] was investigated and the obtained results were originally published in [1]. The *ErbB* signaling pathways regulate several physiological responses such as cell survival, proliferation and motility. The malfunction or hyperfunction of these pathways are involved in the explanation of various types of human cancers. These types of pathways are under active examination as possible drug targets.

In our representation the *ErbB* signaling pathway model consists of 504 species, 1082 complexes and 1654 reactions. The model description was originally a sparse representation. With the help of the *LP-MAX* algorithm the dense realization is computed. The algorithm using the MILP approach was unable to complete the optimization within reasonable time. The resulting dense realization contains 1683 reactions: 29 mathematically possible extra reactions originating from 15 different complexes compared to the published model. The overall computational time was 4993 seconds. The extra reactions introduced into the network can be seen in Fig. 3.6. The list of the extra reactions can be found in Table 3.3. The notations of the complexes in Table 3.3 are the same as in the original model description published at [141].

The sparse realization was also extracted from the dense realization with the help of the l^1-norm based sparse search. The resulting network had the same structure as the original sparse representation. The computational time was around 430 seconds.

3.3 Finding dynamically equivalent weakly reversible realizations with LP

In the following, new, LP-based methods will be introduced to compute weakly reversible realizations of KRNs. The presented methods are compared to the previously enumerated methods known from the literature (see Subsection 3.1.5 for details) in terms of computational time while trying to find weakly reversible realizations of large-size KRNs.

DOI:10.15774/PPKE.ITK.2014.002

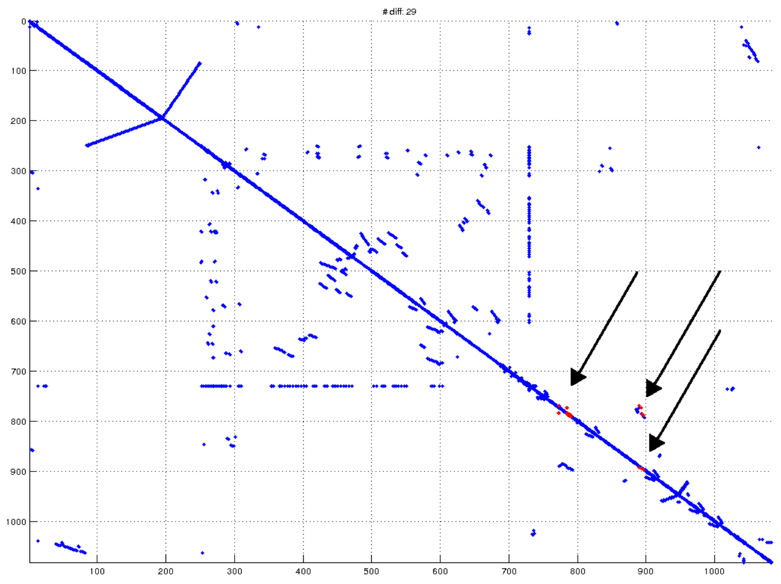


Figure 3.6: Structure of the A_k matrix representing the dense realization of the *ErbB* network. The differences between the sparse and dense realizations are colored and marked by the arrows.

3.3.1 Introduction of new algorithms to compute weakly reversible realizations

The algorithms to compute weakly reversible realizations of KRNs presented in the literature are based on MILP, which makes them computationally hard. Thus, the analysis of large scale networks is practically impossible. Besides of that, these algorithms are unable to handle the phenomena of linear conjugacy.

In the following, two LP-based methods are introduced. The first can compute dynamically equivalent, weakly reversible realizations of KRNs while the other is able to extend the search to linearly conjugate networks.

3.3.1.1 New, LP-based method to compute dynamically equivalent weakly reversible realizations

In [2] a new, LP-based method appeared to compute dynamically equivalent, weakly reversible realizations of KRNs. In the following this algorithm, shortly referred to as *WR-LP1* in the rest of the thesis, is briefly introduced.

Let us denote the ith column of A_k and M with $\mathbf{z}_i$ and $\mathbf{m}_i$, respectively. Then eq. (3.9) can be written as

$$Y\mathbf{z}_i = \mathbf{m}_i, \quad i = 1, \dots, m \tag{3.52}$$

It is well-known from linear algebra that all solutions for eq. (3.52) can be characterized as the sum of particular solutions and the linear combinations of the solutions of the homogeneous

DOI:10.15774/PPKE.ITK.2014.002

	Starting complex → ending complex
1.	(ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K + PIP3 → PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:PIP2
2.	(ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:PIP2 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K → PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:PIP2
3.	PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:PIP2 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:PIP2 → PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)2
4.	(ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)2 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:PIP2 → PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)2
5.	PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)2 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)2 → PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)3
6.	(ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)3 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)2 → PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)3
7.	PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)3 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)3 → PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)4
8.	(ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)4 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)3 → PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)4
9.	PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)4 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)4 → PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)5
10.	(ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)5 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)4 → PIP3 + (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)5
11.	(ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K + PIP2 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:PIP2 + PIP2
12.	(ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:PIP2 + PIP2 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:PIP2 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)2 + PIP2
13.	(ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)2 + PIP2 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)2 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)3 + PIP2
14.	(ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)3 + PIP2 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)3 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)4 + PIP2
15.	(ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)4 + PIP2 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)4 → (ErbB3:ErbB2)_P:GAP:Grb2:Gab1_P:PI3K:(PIP2)5 + PIP2

Table 3.3: List of all the extra reactions in the dense realization of the *ErbB* signaling pathway network. 29 mathematically possible extra reactions originating from 15 different complexes were found.

DOI:10.15774/PPKE.ITK.2014.002

system that can be written as

$$\mathbf{z}_i = \mathbf{z}_i^{(p)} + \sum_{j=1}^{r} \kappa_{i,j} \mathbf{z}_j^{(f)}, \quad i = 1, \ldots, m \tag{3.53}$$

where $\kappa_{i,j} \in \mathbb{R}$, r is the nullity (i.e. kernel dimension) of Y, $\mathbf{z}_i^{(p)}$ is a particular solution for (3.52), and $\mathbf{z}_j^{(f)}$ is the jth kernel base vector of Y. Since Y and M are given, $\mathbf{z}_i^{(p)}$ and $\mathbf{z}_j^{(f)}$ are known for $i = 1, \ldots, m$ and $j = 1, \ldots, r$. Thus, the unknowns in the problem will be the coefficients $\kappa_{i,j}$.

Recall eq. (3.32), namely that a KRN is weakly reversible if and only if there is a strictly positive vector $\mathbf{h} = [h_1 \ \ldots \ h_m]^T$ in the kernel of the matrix A_k. Then the condition eq. (3.32) is given by

$$\sum_{i=1}^{m} \mathbf{z}_i h_i = 0. \tag{3.54}$$

Substituting eq. (3.53) into eq. (3.54) gives

$$\sum_{i=1}^{m} \mathbf{z}_i^{(p)} h_i + \sum_{i=1}^{m} \sum_{j=1}^{r} h_i \kappa_{i,j} \mathbf{z}_j^{(f)} = 0 \tag{3.55}$$

Let us introduce the following new variables

$$v_{i,j} = h_i \kappa_{i,j}, \quad i = 1, \ldots, m, \quad j = 1, \ldots, r \tag{3.56}$$

Using this notation, eq. (3.55) reads

$$\sum_{i=1}^{m} \mathbf{z}_i^{(p)} h_i + \sum_{i=1}^{m} \sum_{j=1}^{r} v_{i,j} \mathbf{z}_j^{(f)} = 0 \tag{3.57}$$

that is now linear in the variables h_i and $v_{i,j}$.

We are in a lucky situation considering the sign constraints of the elements of the original matrix A_k, since $\mathbf{h}$ is element-wise strictly positive (i.e. multiplying with $\mathbf{h}$ does not alter the signs of the elements in A_k). Let us denote by $z_{i,j}^{(p)}$ and $z_{i,j}^{(f)}$ the jth scalar elements of the vectors $\mathbf{z}_i^{(p)}$ and $\mathbf{z}_i^{(f)}$, respectively. Then we can set the following constraints:

$$z_{i,k}^{(p)} h_i + \sum_{j=1}^{r} v_{i,j} z_{j,k}^{(f)} \geq 0, \quad i, k = 1, \ldots, m, \quad i \neq k \tag{3.58}$$

$$z_{i,i}^{(p)} h_i + \sum_{j=1}^{r} v_{i,j} z_{j,i}^{(f)} \leq 0, \quad i = 1, \ldots, m. \tag{3.59}$$

For convenience and easy implementation, the steps of the *WR-LP1* algorithm are summarized in Table 3.4, where the input data are Y and M, and the output is the Kirchhoff

```
$A_k$=WR-LP1($Y$,$M$)
1   $A_k$:=$0 \in \mathbb{R}^{m \times m}$
2   Determine the particular and homogeneous solutions $z_i^{(p)}$
    for $i = 1, \dots m$, and $z_j^{(f)}$ for $j = 1, \dots, r$ from eq. (3.52).
3   Check the feasibility of eq. (3.55) and eqs. (3.58)-(3.59) with variables $\mathbf{h}$ and $v$
    as a linear programming problem with arbitrary linear objective function.
4   If there exists a feasible solution:
5       Determine $\kappa_{i,j}$ from eq. (3.56).
6       Compute the values of the original variable $A_k$ according to eq. (3.53).
7       return $A_k$;
8   Else
9       return 0;
```

Table 3.4: Steps of the method *WR-LP1* for finding weakly reversible, dynamically equivalent realization.

matrix of the computed weakly reversible dynamically equivalent realization if such exists, or 0 if the problem is infeasible.

3.3.1.2 New, LP-based method to compute linearly conjugate weakly reversible realizations

In this Subsection, we are going to present a new algorithm which is related to the MILP-based method briefly summarized in Subsection 3.1.5. The presented results correspond to Thesis II. (see 4.3.3). The mentioned method is now extended to be able to deal with linear conjugacy and is also modified to eliminate the boolean decision variables from the model to obtain an LP-based algorithm. This algorithm originally appeared in [2].

Let us assume that the matrix M defined in eq. (3.9) containing the monomial coefficients of a kinetic system is given. It is known from [74] that linear conjugacy between two KRN models can be expressed by the following constraints:

$$M = T \cdot Y \cdot A_k \tag{3.60}$$

$$\sum_{i=1}^{m} [A_k]_{i,j} = 0, \quad j = 1, \dots, m \tag{3.61}$$

$$[A_k]_{i,j} \geq 0, \quad i, j = 1, \dots, m, \quad i \neq j, \tag{3.62}$$

$$d_i > 0, \quad i = 1, \dots, n, \tag{3.63}$$

where the unknowns are the parameters of the positive diagonal transformation $T = \mathrm{diag}(\mathbf{d})$, and the off-diagonal elements of the Kirchhoff matrix A_k. The actual Kirchhoff matrix A_k' of the KRN realization that is linearly conjugate to the original kinetic system eq. 3.10 defined by M and Y, can be computed from A_k and $\mathbf{d}$ using the following scaling (see for [74] the details):

$$A'_k = A_k \cdot \mathrm{diag}(\psi(\mathbf{d})). \tag{3.64}$$

Additionally, we use the auxiliary variable $\tilde{A}_k$ defined in eq. (3.34) and constraints (3.35) to ensure weak reversibility, where again, A_k and $\tilde{A}_k$ are structurally equal.

Using the fact that the off-diagonal elements of Kirchhoff matrices are non-negative, we can enforce the structural equality of A_k and $\tilde{A}_k$ in our improved method using linear constraints without integer variables. Similarly to the solution in [74] (the constraints of which were summarized in eqs. (3.41)-(3.42)), we consider an off-diagonal element of a Kirchhoff matrix practically nonzero if it is greater than an appropriately chosen small positive value p_l such that $0 < p_l \ll 1$. (This means that off-diagonal elements less than p_l are truncated to zero in A_k and $\tilde{A}_k$.) Moreover, the following upper bounds are assumed for the off-diagonal elements of A_k and $\tilde{A}_k$ with $p_u = \frac{1}{p_l}$.

$$[A_k]_{i,j} < p_u \text{ and } [\tilde{A}_k]_{i,j} < p_u \text{ for } i,j = 1,\dots,m, \quad i \neq j. \tag{3.65}$$

Now we set the following constraints for ensuring the structural equality of A_k and A'_k.

$$[A_k]_{i,j} - p_u^2 \cdot [\tilde{A}_k]_{i,j} \leq 0, \quad i,j = 1,\dots,m, \quad i \neq j \tag{3.66}$$

$$-[A_k]_{i,j} + p_l^2 \cdot [\tilde{A}_k]_{i,j} \leq 0, \quad i,j = 1,\dots,m, \quad i \neq j \tag{3.67}$$

Let us examine the correctness of the constraints (3.66)-(3.67). For this, we have to take into account the upper bounds in eq. (3.65), too.

- If $[A_k]_{i,j} > p_l$ and $[\tilde{A}_k]_{i,j} > p_l$ then one can see that $[A_k]_{i,j} < p_u^2 \cdot [\tilde{A}_k]_{i,j}$ so ineq. (3.66) is fulfilled. Moreover, because $[A_k]_{i,j} > p_l^2 \cdot [\tilde{A}_k]_{i,j}$, ineq. (3.67) holds, too.
- If $[A_k]_{i,j} = 0$ and $[\tilde{A}_k]_{i,j} = 0$ then ineqs. (3.66) and (3.67) are trivially fulfilled.
- If $[A_k]_{i,j} > p_l$ and $[\tilde{A}_k]_{i,j} = 0$ then ineq. (3.66) is violated.
- Similarly, if $[A_k]_{i,j} = 0$ and $[\tilde{A}_k]_{i,j} > p_l$ then ineq. (3.67) is violated.

In summary, the linear constraint set containing only continuous variables to find weakly reversible linearly conjugate KRN realizations is the following: eq. (3.60) stands for the linear conjugacy, (3.61)-(3.62) encode the Kirchhoff property of A_k, and ineq. (3.63) ensures the positivity of the linear conjugacy transformation T. Constraints (3.35) introduce a scaled auxiliary Kirchhoff matrix $\tilde{A}_k$ that is weakly reversible, and finally, (3.66)-(3.67) ensure the structural equality of A_k and $\tilde{A}_k$. The input data of the method are Y and M and the decision variables are the matrix elements $[A_k]_{i,j}$, $[\tilde{A}_k]_{i,j}$ for $i,j = 1,\dots,m$, $i \neq j$, and the scaling factors d_k for $k = 1\dots,n$ for the transformation T. Clearly, the feasibility of the

DOI:10.15774/PPKE.ITK.2014.002

constraints can be checked within the framework of linear programming. In this case, the objective function can be utilized to prescribe certain additional properties of the solution (if it exists). For example, to obtain a sparse weakly reversible realization of the studied kinetic system, the l^1-norm of the elements of A_k can be minimized, provided that the number of complexes in the KRN is large enough [40, 1].

Later on, we will refer to this method as the *WR-LP2* algorithm.

3.3.2 Performance analysis of the different methods

The capabilities of the above presented algorithms are illustrated through three examples. The presented results originally appeared in [2]. In the first example a dynamically equivalent weakly reversible realization does not exist but interestingly, a linearly conjugate, weakly reversible one does exist, and both cases are handled correctly by the applied computational method. The second example highlight a case where the algorithms are able to show the non-existence of a dynamically equivalent weakly reversible realization of a given KRN, as it is previously expected. Finally, the algorithms are compared in terms of computational time. All the computations were performed in the same environment described in Subsection 3.2.3. The CLP solver [138] was used to solve the LP problems, while the GLPK solver [142] was used to compare the results with the previously published MILP-based method [74]. The threshold to discriminate between zero and nonzero rate coefficients was set to 10^{-3}.

Example 1. In this example which originally appeared in [75], a reaction network is shown which doesn't have a dynamically equivalent weakly reversible realization, but it has a linearly conjugate weakly reversible one. The network is characterized by

$$Y = \begin{bmatrix} 1 & 2 & 2 & 2 & 1 & 2 & 1 & 2 & 1 & 1 \\ 2 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 2 & 2 & 1 \end{bmatrix}$$

and the Kirchhoff matrix A_k containing the following non-zero off-diagonal elements: $[A_k]_{2,1} = 1$, $[A_k]_{4,2} = 0.5$, $[A_k]_{5,4} = 1.5$, $[A_k]_{7,4} = 0.5$, $[A_k]_{1,7} = 0.5$, $[A_k]_{4,7} = 1$. One can see that this (Y, A_k) realization is non-reversible.

Firstly, while applying the presented *WR-LP2* algorithm on the above described system, we have fixed the T matrix as an identity. This means that instead of looking for linearly conjugate realizations during the search, only the dynamically equivalent realizations were considered. The algorithm found the constraint set infeasible as expected.

Then, by relaxing the fixed value of the matrix T we enabled the search for linearly conjugate realizations, too. Now the algorithm succeeded, determining a linearly conjugate weakly reversible realization with the following non-zero off-diagonal elements in the matrix $A_k^{(2)}$: $[A_k^{(2)}]_{2,1} = 0.001$, $[A_k^{(2)}]_{4,2} = 0.005$, $[A_k^{(2)}]_{7,4} = 0.002$, $[A_k^{(2)}]_{1,7} = 0.005$, $[A_k^{(2)}]_{4,7} = 0.001$ and linear conjugacy matrix $T = diag([\frac{1}{0.001}, \frac{1}{0.001}, \frac{1}{0.004}])$. For these values, the equation for linear conjugacy $Y \cdot A_k = T \cdot Y \cdot A_k^{(2)}$ holds (see eq. (3.60)).

DOI:10.15774/PPKE.ITK.2014.002

Example 2. Let us consider the KRN representation of the classical 3-dimensional kinetic Lorenz system as it is described in Subsection 3.1.2.1. The system is characterized by the matrices (Y^l, A_k^l) from eq. 3.13.

The *WR-LP2* algorithm found the problem infeasible during the search for dynamically equivalent and linearly conjugate weakly reversible realization. This coincides with the results of [128], where several thousand dynamically equivalent sparse realizations were computed with an efficient method, but none of them was weakly reversible.

Example 3. In this example containing several randomly generated KRNs, the results of the performance comparison of the presented algorithms are summarized while dealing with large scale networks. As it was shown, all three algorithms, namely the *graph-based* method and the *WR-LP1* method presented in Subsection 3.1.5, and the *WR-LP2* method from Subsection 3.3.1.2 are purely LP-based algorithms. To be able to compare the three methods, only dynamically equivalent realizations were searched for.

All the algorithms were tested on a set of randomly generated KRNs. All the networks were built up from 10 species but contained different number of complexes: scenarios with 9, 30, 56, 90 complexes were set up, respectively. The methodology of generating the random kinetic systems was the same as in Subsection 3.2.3. The obtained random KRN was extended with additional directed edges (if needed) to ensure that the resulting reaction graph is weakly reversible. This step was solved as an unweighted graph augmentation task [44, 98]. The Kirchhoff matrix of the augmented weakly reversible network is denoted by A_k'. The inputs for the realization computation algorithms were the matrices Y and $M' = Y \cdot A_k'$.

The evaluation of the effectiveness of the algorithms - i.e. how the solution time is changing as the size of the computational task is growing - can be found in Table 3.5. The columns of the table show the size of the matrix A_k (i.e. the number of complexes in the network) which basically determines the number of variables and constraints (depending also on the individual method). For each problem size, 10 different random KRNs were tested. In some cases the solver was unable to solve the problem in the given time limit (300s), these were considered as unsuccessful solution attempts. For any method, no incorrect solutions were obtained. Only the successful solutions were taken into account during the calculation of the average solution times. One can find the number of successful solutions (out of the original 10 problems for each problem size) in the corresponding rows of Table 3.5. In the remaining rows, the averaged solution times and the sizes of the generated LPs can be found for each algorithm.

One can note, that despite of the fact that both the *WR-LP1* and the *WR-LP2* algorithms solve a single LP problem, there is a significant difference between the solution times. This is caused by the different structure of the problem generated by the two algorithms. Although the number of variables and constraints is higher, the algorithm *WR-LP2* generates a clear and sparse structure both for the equality and inequality constraints as it can be seen in Fig. 3.8, while *WR-LP1* generates a nearly full coefficient matrix to describe the equality

	Network size (m)	9	30	56	90
Graph-based-LP					
	time (s)	0.04	0.77	4.26	18.98
	success	10/10	10/10	10/10	10/10
	# of optim. vars *	14	37	65	101
	# of eq. constr. *	4	6	8	10
	# of ineq. constr. *	14	37	65	101
WR-LP1					
	time (s)	0.001	2.04	5.98	-
	success	10/10	10/10	4/10	0/10
	# of optim. vars	63	780	2800	7380
	# of eq. constr.	9	30	56	90
	# of ineq. constr.	81	900	3136	8100
WR-LP2					
	time (s)	0.003	0.23	1.62	11.28
	success	10/10	10/10	10/10	10/10
	# of optim. vars	162	1800	6272	16200
	# of eq. constr.	45	210	504	990
	# of ineq. constr.	324	3600	12544	32400

Table 3.5: Comparison of the presented algorithms in terms of computational time while dealing with KRNs having different sizes. *WR-LP2* algorithm outperforms all the other methods. All the compared methods are based on LPs as the constrained dense search is also implemented with LP for the graph-based method. The size of the generated LPs also appear in the table. *: the graph-based method calculates m^2 LPs with the given size.

constraints (see Fig. 3.7). This fact has a serious effect on the computational time of the solution of the corresponding LP problems causing that *WR-LP1* can not complete the computation within the predefined time limit in the case of larger networks.

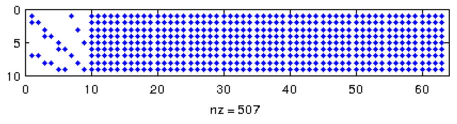
3.4 Finding dynamically equivalent realizations with mass conservation

In this Section, we introduce additional constraints into the optimization problem originally defined in [118] to ensure that the resulting realizations, besides other requirements, also fulfill the mass-conservation property. The resulting computation method of the realizations leads to an MILP problem. The presented results correspond to Thesis II. (see 4.3.3) and originally presented in [4].

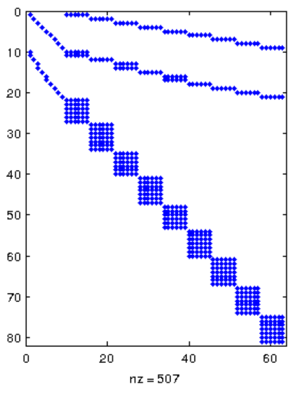
Recall that the linear constraints corresponding to the mass action dynamics and the Kirchhoff property can be originated from eq. (3.9) and (3.8) leading to the form defined in eqs. (3.18)-(3.19).

Considering eq. (3.44) in order to ensure the mass-conservation property, a reaction

DOI:10.15774/PPKE.ITK.2014.002

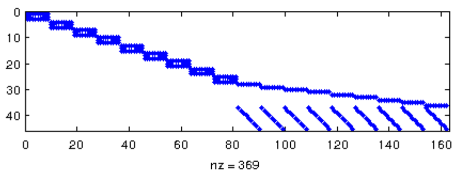


(a) Structure of the matrix describing the equality constraints in the LP problem. The size of the matrix is 9×63.

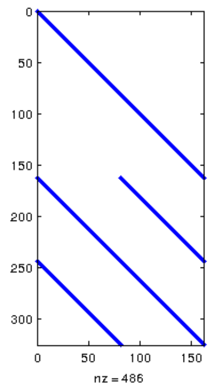


(b) Structure of the matrix describing the inequality constraints in the LP problem. The size of the matrix is 81×63.

Figure 3.7: Structure of the constraint set in case of the *WR-LP1* algorithm. Rows and columns represent constraints and variables, respectively. The original KRN contains 2 species and 9 complexes. Non-zero elements of the matrices are marked. It can be seen that that the equality constraints formulate a nearly full matrix.



(a) Structure of the matrix describing the equality constraints in the LP problem. The size of the matrix is 45×162.



(b) Structure of the matrix describing the inequality constraints in the LP problem. The size of the matrix is 324×162.

Figure 3.8: Structure of the constraint set in case of the *WR-LP2* algorithm. Rows and columns represent constraints and variables, respectively. The original KRN contains 2 species and 9 complexes. Non-zero elements of the matrices are marked. As one can note, the algorithm generates sparse constraint matrices with clear structure.

between complex i and j can be present if and only if $g \cdot \rho^{(i,j)} = 0$ where g is the vector of scaled molecular weights and $\rho^{(i,j)} = [Y]_{\cdot,i} - [Y]_{\cdot,j}$ is the *reaction vector* [10]. This constraint is formulated for each (i, j) pair $i, j = 1, \ldots, m,\ i \neq j$ in the following way:

$$g \cdot \rho^{(i,j)} = 0 \ \Rightarrow \ [A_k]_{i,j} + [A_k]_{j,i} \geq 0 \tag{3.68}$$

where $\Rightarrow$ stands for the logical implication operation. To keep track of which complexes are able to react with each other (while preserving the mass-conservation) we introduced an auxiliary binary valued decision variable $\delta_l,\ l = 1, \ldots, m \cdot (m-1)/2$ for each constraint in the following way:

$$\begin{aligned} g \cdot \rho^{(i,j)} = 0 \quad &\Rightarrow \quad \delta_l = 0 \\ g \cdot \rho^{(i,j)} \neq 0 \quad &\Rightarrow \quad \delta_l = 1 \end{aligned} \tag{3.69}$$

$$\delta_l = 0 \quad \Rightarrow \quad [A_k]_{i,j} + [A_k]_{j,i} \geq 0 \tag{3.70}$$

$$\delta_l = 1 \quad \Rightarrow \quad [A_k]_{i,j} + [A_k]_{j,i} = 0 \tag{3.71}$$

where eq. (3.70)-(3.71) control the reaction graph structure: if reaction $C_i \to C_j$ is forbidden then $[A_k]_{i,j}$ is forced to be 0, otherwise it has a non-negative value. After proper reformulation, these constraints can be incorporated into the MILP problem.

The proper bounds for the elements of g is introduced to ensure that g is a strictly positive vector. Also, if the problem in question dictates it, it is possible to exclude isomers (i.e. species with the same scaled molecular weights) by adding the following constraint: $g(i) \neq g(j),\ i, j = 1, \ldots, n,\ i \neq j$.

Finally, a KRN which fulfills all the constraints is obtained by evaluating arbitrary objective function with respect to the above listed constraints, where the optimization vector consists of the elements of matrix A_k and vector g as real-valued variables, and $m \cdot (m-1)/2$ binary valued variables obtained from eqs. (3.68)-(3.71). It should be noted that specific objective functions can lead to realizations having further special properties, e.g. minimal/maximal number of reactions etc.

The size of the emerging MILP problem is mainly determined by the number of appearing auxiliary binary variables which has the magnitude $\mathcal{O}(m^2)$.

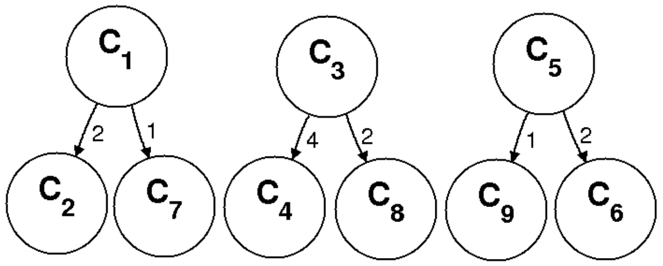
Example. Let us consider the following kinetic ODEs:

$$\begin{aligned} \dot{x}_1 &= -2x_1^2x_2 + 4x_2^2 - 2x_1^4 \\ \dot{x}_2 &= x_1^2x_2 - 2x_2^2 + x_1^4 \end{aligned} \tag{3.72}$$

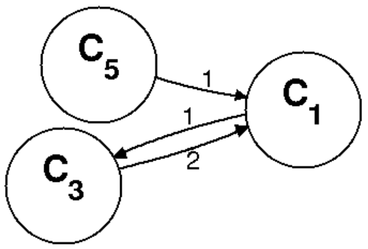
The so-called "canonical" reaction graph of eq. (3.72) can be formulated with the help of the algorithm described in [62]. The resulting reaction network is depicted in Fig. 3.9a where the complex composition matrix is the following:

$$Y = \begin{bmatrix} 2 & 1 & 0 & 1 & 4 & 3 & 2 & 0 & 4 \\ 1 & 1 & 2 & 2 & 0 & 0 & 2 & 1 & 1 \end{bmatrix} \tag{3.73}$$

The non-zero off-diagonal elements of the matrix A_k describing the reaction graph are the following: $A_k(2,1) = 2,\ A_k(7,1) = 1,\ A_k(4,3) = 4,\ A_k(8,3) = 2,\ A_k(6,5) = 2,\ A_k(9,5) = 1$. Based on eq. (3.5) the algebraic structure of the system can be characterized by the following



(a) Canonical realization of eq. (3.72).



(b) A mass conservative realization of eq. (3.72).

Figure 3.9: The canonical and an alternative, dynamically equivalent mass conservative realization of the dynamical system described in (3.72) In both subfigures, node numbers refer to the columns of Y from eq. (3.73). Elements of A_k and A_k^{mc} appear as edge weights on the proper subfigures.

matrix:

$$Y \cdot A_k = \begin{bmatrix} -2 & 0 & 4 & 0 & -2 & 0 & 0 & 0 & 0 \\ 1 & 0 & -2 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

It can be seen that this realization is not mass conservative because in each linkage class at least one semi-positive reaction vector can be found.

By applying the proposed algorithm with the maximization of the cost function, a dynamically equivalent reaction network was found, which fulfills the mass-conservation property. The resulting reaction graph is presented in Fig. 3.9b. Isolated complexes (i.e. those complexes which do not react with any other) were omitted. By looking at the reaction vectors in the reaction graph and the obtained vector of scaled molecular weights $g = [1\ 2]$, the presence of the mass-conservation property can be seen. The resulting reaction graph is described by matrix A_k^{mc} in which the non-zero off-diagonal elements are the following: $A_k^{mc}(3,1) = 1$, $A_k^{mc}(1,3) = 2$, $A_k^{mc}(1,5) = 1$. For this system the complex composition matrix Y is still the one from eq. (3.73). It can be seen that $Y \cdot A_k = Y \cdot A_k^{mc}$. Therefore the two systems are dynamically equivalent based on eq. (3.11).

3.5 Summary

In this Chapter a set of new algorithms were proposed to compute alternative realizations of KRNs with prescribed structural/dynamical properties. By analyzing the properties of the system model and the classical MILP-based description, simplified algorithms were developed which have polynomial complexity. Also, it has been shown that new, physically meaningful phenomena (such as mass conservation) can be incorporated into an optimization-based framework applied to the computation of alternative realizations.

Two new algorithms were introduced to compute dynamically equivalent realizations of reaction networks having minimal and maximal number of reactions in Sections 3.2 and 3.3, respectively. The algorithms formulate the problems in an LP-framework, thus they are

DOI:10.15774/PPKE.ITK.2014.002

capable to deal with large-scale, biologically meaningful reaction networks. The algorithms replace the previously published MILP-based methods which have complexity issues if the size of the network is increased. The correctness of the new algorithms has been shown. They were compared to the classical, combinatorial optimization based techniques with respect to the solution time in case of several reaction networks having different sizes. The algorithms can be easily applied in a parallel framework, too.

Also, linear programming based methods were have developed and analyzed to compute dynamically equivalent weakly reversible realizations of kinetic systems. Similarly to a result published in [122], it was shown that the dense dynamically equivalent weakly reversible realization structure of a kinetic system contains all other possible dynamically equivalent weakly reversible structures as proper subgraphs if the complex set is fixed. Based on the author's results on the LP-based computation of dense realizations, a previously published graph-theory-based method was re-implemented without integer variables. In addition, two new computation methods were also proposed, having polynomial time complexity, too. The implemented methods were tested on examples taken from the literature, and then they were compared from the point of view of computational performance on reaction networks of increasing size. The numerical tests showed that the LP-based methods solve the problems correctly, while avoiding the complexity issue which emerges during the solution of the former MILP-based problems. It was also shown that the structure of the constraint set in the LP problems has serious impact on the solution time of the problem.

During the development of the above mentioned method, it has been concluded that an interesting research direction would be the inclusion of certain conservation laws into the LP-structure. These requirements can be any structural or parametrical properties which can be formulated as linear constraints. Such developments would support the elimination of physically infeasible reaction networks from the set of mathematically possible solutions.

This leads us to the formulation of the third method introduced in this Chapter. This MILP-based method is able to compute dynamically equivalent realizations of KRNs with mass conservation property. The correctness of the proposed method has been shown with the help of examples taken from the literature. By extending the proposed method a large set of network properties could be investigated, too, e.g. linear conjugate KRNs with mass conservation, structural/parametrical uniqueness of mass conservative networks etc. Also, by improving the proposed methods, controller design methods can be implemented to find stabilizing controllers for dynamical systems represented with KRNs [116].

The results described in this Chapter are summarized in Thesis I. and II. (see Subsection 4.3.3 and 4.3.3, respectively). The corresponding publications are [1, 2, 4].

DOI:10.15774/PPKE.ITK.2014.002

Chapter 4

Efficient scheduling of railway networks using optimization

The increasing load on transportation networks, especially on railway networks motivate us to develop new network controlling methods. With the help of them, the throughput of the railway network can be kept on the maximal level in case of disturbances, too. In this work, we focus on the case when the disturbances are delays, no track blocking or loss of trains are considered. As it is known, delays can quickly propagate all over a railway network due to the nearly impossible overtake of the delayed train [33]. Hence, to recover a network from a delay as soon as possible, the rescheduling of the trains is inevitable.

In this Chapter, we address the problem of the rescheduling of a railway network in case of delayed operation. This problem is also known from the literature as the dispatching problem of railway networks (see e.g. [26, 32]), where a train dispatcher tries to react to delays during operations.

The scheduling problem in railway networks is investigated in a model predictive control framework, formulated as an MILP problem. Using dynamic traffic management we develop and study techniques that minimize the total delay over the prediction horizon by changing the order of the trains in an optimal way. The formulation of the optimization problem is described in Section 4.1.

By exploiting the structure of the problem we show that a problem-specific reordering of the constraints in the model results in a significant speedup in the solution time compared to the built-in methods of the MILP solvers. Moreover, with proper reformulation of the constraints the dependence between the events in the network and the control variables becomes easy to analyse. The proposed formulation enables the future development of a problem-specific solution method to replace the general purpose MILP solvers in the solution process. The advantages of the resulting model formulations are detailed and the computational efforts needed to find the optimal solution are compared using each forms of the model. The simulation results confirm the effectiveness of the proposed control technique

DOI:10.15774/PPKE.ITK.2014.002

in case of complex delay scenarios, too. Furthermore, we show that the formulated model is applicable to find the most delay-sensitive parts of the network. The algorithm is applied on the model of the Dutch railway network.

The results presented in the current Chapter are summarized in Thesis III., see Subsection 4.3.3 and originally appeared in [5, 3].

4.1 Optimization-based control of railway networks to minimize total delays

Delay management is a topic having crucial importance in case of railway networks. Due to the fact that overtaking is nearly impossible in railway networks, the delay propagation is very fast and a small disturbance in the prescribed schedule can cause large delays all over the network.

Recently, several methods were proposed to help the recovering of the network from a delay scenario. Robust timetables were designed [58], rescheduling methods and connection breaking rules were introduced to minimize the effect of the disturbance on the network [69]. The model-based control of the railway networks give us the opportunity to apply proper rescheduling actions and improve the robustness of the network against delays.

A railway network can be modeled using the so-called microscopic approach, where a lot of infrastructural elements are contained in the model [97]. These elements include all tracks, points, signals, stop boards, speed indicators, platforms, and other elements required for mapping a rail network in deep details. However, due to the extremely large size of these models, long computation times can appear for large and highly utilized networks. The macroscopic approach used in this thesis gives us the opportunity to handle large size (country-wide) networks by omitting some details from the network description [57]. In this thesis, only stations and tracks are modeled together with the trains running on them.

In this Section we show how to design a feedback controller that is able to predict the future behavior of the railway network with a cyclic timetable and reorder the trains using the same track in an optimal way to minimize the total secondary delay in the network [21]. The control problem is formulated as a MILP problem [25] in a model predictive framework using the *switching control* methodology. In this approach each operation mode of the network is modeled and can be selected by choosing the appropriate combination of control variables. In this particular setup control variables are binary valued. During the optimization this set of binary control variables is determined while minimizing the given cost function, namely the total secondary delay over the prediction horizon. The solution of the emerging MILP problems can lead to computational difficulties since integer programming is known to be NP-hard (see Subsection 2.3.2 and [93]).

4.1.1 Basics of the model formulation

The general structure of the control architecture can be seen in Fig. 4.1. The railway network is affected by delays shifting the departure and arrival times away from their nominal

DOI:10.15774/PPKE.ITK.2014.002

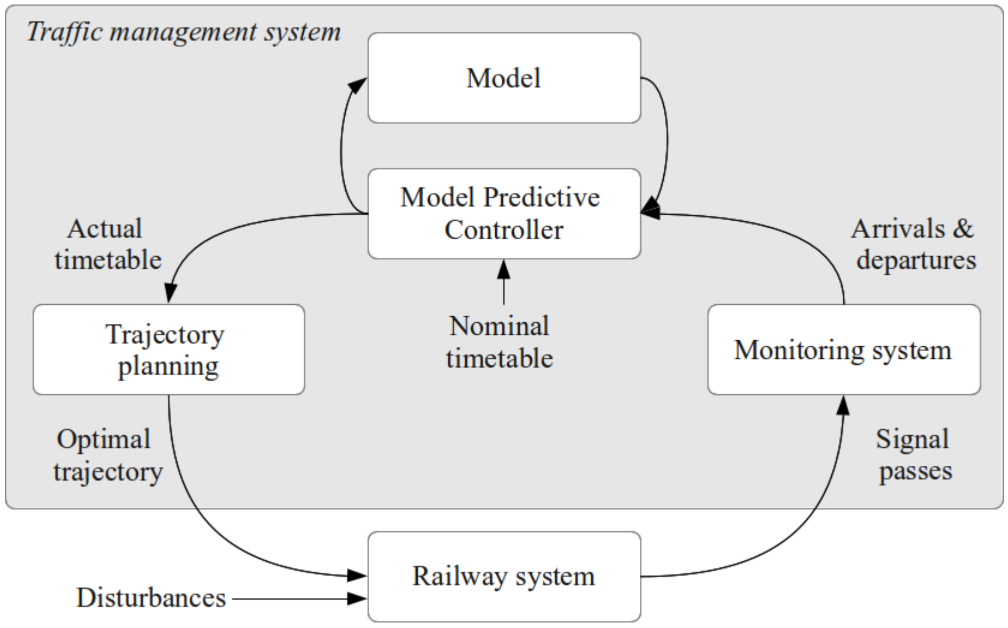


Figure 4.1: Structure of the railway traffic management system. In this work, the algorithm used in the *Model Predictive Controller* is detailed.

values. The controller reads the departure and arrival times of each train from the monitoring system and - by considering the nominal timetable and the network model - an actual timetable is generated, which can be used to generate optimal trajectories and routes for the trains. The calculation of a new actual timetable can be completed every time when an event (departure or arrival) is captured by the monitoring system.

Let us consider a railway network scheduled by a periodic timetable having cycle time T_c. The timetable is repeated every T_c minutes. In one cycle all trains departing in one period of the nominal timetable are modeled. Periodic timetables ensure the stability of the schedule in such a way that if a delay of a given train exceeds the cycle time, then that specific train is omitted from the schedule and the train from the next cycle will run instead of it. This means that the delays are upper bounded with the value of T_c.

Generally, the network has three main operation mode. During *nominal operation* trains depart and arrive according to the original timetable following a pre-determined route. In this mode of operation, the order of the trains on the tracks is defined by the timetable itself. In *perturbed operation* the timetable is corrupted by an emerging delay, causing the deviation of the network from the nominal operation. Every change from the nominal schedule can be associated with a perturbed mode. In order to recover the network from delays, control actions are taken to formulate a schedule which can decrease the effect of the delays on the network. This mode of operation is called *controlled operation*. In this work, control actions are narrowed down to the reordering of the trains on a track and breaking up connections. The aim of the controlled operation is to find a schedule which has minimal deviation from the nominal timetable. The measure of the deviation from the nominal timetable is the sum

DOI:10.15774/PPKE.ITK.2014.002

of the secondary delays. Note, that early arrivals are not considered in the delay calculation. The model presented in [20, 21] is further extended in this work to handle double tracks with a controlled track selection, and joint trains with controlled breakup if needed.

The operation of the railway network is described as a set of train runs. Train runs always start and end in a (virtual) station meaning that departures and arrivals denote the beginning and the end of the runs. Virtual stations are formulated where a track ends but no station is present, e.g. at crossings and junctions. Note, that virtual stations also have infinite capacity because there is no dwelling time at these type of stations, meaning that trains do not stop at virtual stations, they serve only as place for possible track change operation. A set of train runs can formulate a line if the same physical train moves along them. In the remainder of this thesis we will simply refer to a 'train run' as a 'train'.

In this work we consider a model which can handle two parallel tracks between stations, where each track has a dedicated direction. If between two stations only a single track is present, overtaking is only possible at stations. It should also be noted that we consider the capacity of the stations as infinite meaning that each station can host an unlimited number of trains at the same time. Also, we assumed that the order of departures from stations can be arbitrarily chosen. Although it should be noted, that limiting the capacity of the stations to a finite value and the exclusion of certain order of train departures can be formulated as linear constraints. Hence, the inclusion of these constraints can be easily completed using the proposed framework.

4.1.2 Constraint set formulation

In the following, we will define the constraint set describing the dependency between the events in the railway network. All the constraints are linear inequality constraints containing continuous and integer-valued variables.

The departure and arrival times of the trains are defined by the following constraints. A train is allowed to depart as soon as all of the corresponding constraints are satisfied.

- *Timetable constraint:* the timetable constraint should be satisfied which means that a departure $d_i(k)$ may not occur before its scheduled departure time

$$d_i(k) \geq r_i^d(k) \tag{4.1}$$

where $r_i^d(k)$ is the scheduled departure time for the ith train in the kth cycle. A similar constraint is formulated for the arrival time $a_i(k)$:

$$a_i(k) \geq r_i^a(k). \tag{4.2}$$

This constraint formulation enforces the trains to accomplish the given run with a nominal running time which is usually greater than the minimal running time. Note that the arrival constraints can be omitted by setting $r_i^a(k)$ to $-\infty$.

DOI:10.15774/PPKE.ITK.2014.002

- *Running time constraint:* let $t_i(k)$ be defined as the minimum running time of the ith train. Then the running time constraint is formulated as follows:

$$a_i(k) \geq d_i(k-\delta_{ii}) + t_i(k) \tag{4.3}$$

  where $\delta_{ii} = 0$ if the departure time and the arrival time are in the same cycle and $\delta_{ii} = 1$ if there is a cycle difference between departure and arrival.

- *Dwell time constraint (or continuity constraint):* in the model formulation, a physical train moving across the network through several stations is divided into several train runs, one between each two successing (virtual) stations. Dwell time constraints are formulated to order these train runs after each other enabling the model to simulate the move of the physical train. Let us denote a train run and its preceding train run by i and p_i, respectively. Let $s_{p_i}(k)$ be the minimum dwell time between arrival of the train run p_i and the departure of train run i, then the dwell time constraint which has to be satisfied is the follows:

$$d_i(k) \geq a_{p_i}(k-\delta_{ip_i}) + s_{p_i}(k) \tag{4.4}$$

  where $\delta_{ip_i} = 0$ if in cycle (k) train p_i arrives and proceeds as train i in the same cycle, and $\delta_{ip_i} = 1$ otherwise.

- *Headway constraints:* let us collect those trains which are moving over the same track in the same direction as train i and are scheduled before train i in the nominal operation mode into the set $\mathcal{H}_i(k)$. Let us define h^d_{ij} and h^a_{ij} as the minimum headway times for departure and arrival, respectively, between train j and train i where $j \in \mathcal{H}_i(k)$. For each train j the headway constraints are defined for both departure and arrival:

$$\begin{gathered} d_i(k) \geq d_j(k-\delta_{ij}) + h^d_{ij} + u_{ij}(k)\beta,\ \forall j \in \mathcal{H}_i(k) \\ d_j(k-\delta_{ij}) \geq d_i(k) + h^d_{ji} + (1-u_{ij}(k))\beta,\ \forall j \in \mathcal{H}_i(k) \end{gathered} \tag{4.5}$$

$$\begin{gathered} a_i(k) \geq a_j(k-\delta_{ij}) + h^a_{ij} + u_{ij}(k)\beta,\ \forall j \in \mathcal{H}_i(k) \\ a_j(k-\delta_{ij}) \geq a_i(k) + h^a_{ij} + (1-u_{ij}(k))\beta,\ \forall j \in \mathcal{H}_i(k) \end{gathered} \tag{4.6}$$

  where β is a large negative number ($\beta \ll 0$) and $u_{ij}(k)$ is a binary control variable used to determine the order of the trains through manipulating the values in the constraints. Note that for $u_{ij}(k) = 0$, the first constraints of ineq. (4.5) and ineq. (4.6) will be active, and for $u_{ij}(k) = 1$, the second constraints of ineq. (4.5) and ineq. (4.6) will be active. Note, that the constant β is incorporated into the right hand side of the constraints as a lower bound in order to implement the capability to select the proper constraints corresponding to the value of the given control variable. If β is active in a constraint (meaning it is multiplied with a non-zero valued control variable), that

DOI:10.15774/PPKE.ITK.2014.002

specific constraint will become meaningless because a stricter constraint will be present in the system.

- *Meeting constraints:* similarly to the headway constraints, let us collect those trains which are moving over the same track in the opposite direction as train i, and are scheduled before train i in the nominal operation mode into the set $\mathcal{M}_i(k)$. Let us define w_{ij} as the minimum separation time between arrival of train j and departure of train i where $j \in \mathcal{M}_i(k)$. For each train j the separation constraint is defined as

$$\begin{aligned} d_i(k) &\geq a_j(k-\delta_{ij}) + w_{ij} + u_{ij}(k)\beta,\ \forall j \in \mathcal{M}_i(k) \\ d_j(k-\delta_{ij}) &\geq a_i(k) + w_{ji} + (1-u_{ij}(k))\beta,\ \forall j \in \mathcal{M}_i(k) \end{aligned} \tag{4.7}$$

  where the roles of β and $u_{ij}(k)$ are similar to the roles they have for the headway constraints.

- *Double tracks:* there are parts of the railway network where two parallel tracks are present in the same direction. In this case not only the order of the trains can be changed by the controller but they can be reallocated to an another track. The headway constraints introduced between trains running on different tracks should be deactivated but they should be active if the trains are allocated onto the same track. Let us denote the set of trains that move over a double track in the same direction as train i, and scheduled before i in the nominal operation as $\mathcal{D}_i(k)$. Now the headway constraints are modified in the following way to handle double tracks:

$$\begin{aligned} d_i(k) &\geq d_j(k-\delta_{ij}) + h^d_{ij} + u_{ij}(k)\beta + u^t_{ij}(k)\beta,\ \forall j \in \mathcal{D}_i(k) \\ d_j(k-\delta_{ij}) &\geq d_i(k) + h^d_{ji} + (1-u_{ij}(k))\beta + u^t_{ij}(k)\beta,\ \forall j \in \mathcal{D}_i(k) \end{aligned} \tag{4.8}$$

$$\begin{aligned} a_i(k) &\geq a_j(k-\delta_{ij}) + h^a_{ij} + u_{ij}(k) + u^t_{ij}(k)\beta,\ \forall j \in \mathcal{D}_i(k) \\ a_j(k-\delta_{ij}) &\geq a_i(k) + h^a_{ij} + (1-u_{ij}(k))\beta + u^t_{ij}(k)\beta,\ \forall j \in \mathcal{D}_i(k) \end{aligned} \tag{4.9}$$

  where u^t_i and u^t_j are binary control variables denoting which track is selected for train i and j, respectively, and $u^t_{ij} = u^t_i \oplus u^t_j$ where $\oplus$ stands for the classical `xor` operation (logical *exclusive OR* operation). All other notations are the same as in the case of the *headway constraints.*

- *Joint trains:* in a railway schedule it can occur that two trains are coupled, which means that they are running together through some part of the network. These trains are called joint trains. If one of the trains is heavily delayed then the controller should be able to break up the coupling connection and run the two train parts separately. We can assume that if train i and j are coupled, they are in the same cycle which means that $\delta_{ij} = 0$ in all cases.

  In case of a joint train the headway constraints in between the two train parts should be deactivated while in case of splitting up the coupling they should be reactivated.

Let us denote the set of trains that move on the same track in the same direction as train i, and are scheduled before it in the nominal operation as $\mathcal{J}_i(k)$. To handle joint trains, the headway constrains are formulated as follows:

$$\begin{gathered} d_i(k) \geq d_j(k) + u^c_{ij}(k)\beta,\ \forall j \in \mathcal{J}_i(k) \\ d_i(k) \geq d_j(k) + (1-u^c_{ij}(k))\beta + h^d_{ij} + u_{ij}(k)\beta,\ \forall j \in \mathcal{J}_i(k) \\ d_j(k) \geq d_i(k) + u^c_{ij}(k)\beta,\ \forall j \in \mathcal{J}_i(k) \\ d_j(k) \geq d_i(k) + (1-u^c_{ij}(k))\beta + h^d_{ij} + (1-u_{ij}(k))\beta,\ \forall j \in \mathcal{J}_i(k) \end{gathered} \tag{4.10}$$

$$\begin{gathered} a_i(k) \geq a_j(k) + u^c_{ij}(k)\beta,\ \forall j \in \mathcal{J}_i(k) \\ a_i(k) \geq a_j(k) + (1-u^c_{ij}(k))\beta + h^a_{ij} + u_{ij}(k)\beta,\ \forall j \in \mathcal{J}_i(k) \\ a_j(k) \geq a_i(k) + u^c_{ij}(k)\beta,\ \forall j \in \mathcal{J}_i(k) \\ a_j(k) \geq a_i(k) + (1-u^c_{ij}(k))\beta + h^a_{ij} + (1-u_{ij}(k))\beta,\ \forall j \in \mathcal{J}_i(k) \end{gathered} \tag{4.11}$$

where u^c_{ij} is a binary control variable which equals to 0 if train i and j are coupled and 1 otherwise. All the other notations are the same as in case of the *headway constraints*. It should be noted that for the logical variables $u^t_i, u^t_j, u^t_i, u^c_{ij}$ the following statement should hold: $\neg(u^t_i\bar{u}^t_j\bar{u}^c_{ij} \vee \bar{u}^t_i u^t_j \bar{u}^c_{ij})$ for all joint train i for all the trains in the corresponding $\mathcal{J}_i(k)$. This can be turned to linear constraints as follows:

$$\begin{aligned} (1-u^t_i) + u^t_j + u^c_{ij} > 0 \\ u^t_i + (1-u^t_j) + u^c_{ij} > 0 \end{aligned} \tag{4.12}$$

These constraints ensure that no infeasible control input combination can appear, namely that the two parts of a joint train would be allocated to different parallel tracks.

In order to simplify the understanding of the above described constraints, we collected the notations of the introduced control variables into the following table:

Notation	Meaning
$u_{ij}(k)$	controls the order of the ith and jth train in the kth cycle
$u^c_{ij}(k)$	connects the ith and jth trains if they are coupled trains
$u^t_i(k)$	controls the track-selection of the ith train in case of double tracks

The above defined constraints defines the time-evolution of the system during operation, thus eqs. (4.1)-(4.12) can be considered as a dynamical model of the system.

Note that in case of nominal operation all δ_{ij} values are equal to zero or one. In case of perturbed operation other values are also possible. For sake of simplicity, we will only consider the case when $\delta_{ij} = \{0, 1\}$.

4.1.3 MILP problem formulation

In order to reformulate the above presented model as an MILP problem which were detailed in Section 2.3, the following steps should be taken.

DOI:10.15774/PPKE.ITK.2014.002

Let us consider a network having n train runs (or shortly trains) as it has been defined before, and define the following vectors for the kth cycle:

$$\bar{x}(k) = \begin{bmatrix} d_i(k) \\ \vdots \\ d_n(k) \\ a_1(k) \\ \vdots \\ a_n(k) \end{bmatrix} \in \mathbb{R}^{2n}; \quad \bar{r}(k) = \begin{bmatrix} r_1^d(k) \\ \vdots \\ r_n^d(k) \\ r_1^a(k) \\ \vdots \\ r_n^a(k) \end{bmatrix} \in \mathbb{R}^{2n}. \tag{4.13}$$

Besides of these, the elements of $u_{ij}(k)$ and $u_{ij}^c(k)$ with $j \in \{\mathcal{H}_i(k), \mathcal{M}_i(k), \mathcal{D}_i(k), \mathcal{J}_i(k)\}$, $i = 1...n$ and variables $u_i^t(k)$, $u_j^t(k)$ for $i, j = 1...n$ are collected in one column vector $\bar{u}(k) \in \mathbb{R}^m$ where m is the total number of control variables in the model for one cycle. Due to the fact that only two events (departure and arrival) is connected to a given train run, but it is controlled by several control variables (e.g. to control the order of the trains running on the same track in the same direction in the given cycle), in case of real-life network models the following relation holds: $m \gg 2n$.

The exact state equation of the model can be defined in a description form using the tools from max-plus algebra, as it is detailed in [21]. In our model the time index k runs for the cycles, and the state variable $\bar{x}(k)$ contains the occurrence times of the lower level events within one cycle. Therefore, there are two time evolutions described in two time scales, the optimization is concerned with the time evolution of the upper (cycle) time scale. The lower level dynamics itself is a discrete event system with pure time delays.

The goal of the model predictive controller is to minimize the sum of the delays over the prediction horizon, which can be calculated as the sum of the deviation of the current model from the reference model. Hence, the objective function (or performance index) can be formulated as

$$J(k) = \sum_{j=0}^{N_p} \Big(\sum_{i=1}^{n} \sigma_i \Big(\bar{x}_i(k+j) - \bar{r}_i(k+j) \Big) + \sum_{l=1}^{m} \rho_l \bar{u}_l(k+j) \Big) \tag{4.14}$$

where N_p is the prediction horizon, and vectors $\sigma \in \mathbb{R}^n$ and $\rho \in \mathbb{R}^n$ are containing non-negative weighting scalars. The objective function eq. (4.14) can be interpreted as follows: the first term penalizes the predicted departure and arrival delays while the second term is related to the changes in the order of the trains during cycle $k + j$. Note that due to constraints (4.1)-(4.2) there holds $\bar{x}_i(k+j) - \bar{r}_i(k+j) \geq 0$, $\forall j$ and so $J(k) \geq 0$. During this work, vectors σ and ρ are considered as uniform valued constants. Although they give the possibility to classify the trains and penalize the delays or schedule modifications of given train types on different levels.

DOI:10.15774/PPKE.ITK.2014.002

By defining the extended vectors

$$\begin{aligned} \tilde{x}(k) &= \begin{bmatrix} \bar{x}(k) \\ \bar{x}(k+1) \\ \vdots \\ \bar{x}(k+N_p) \end{bmatrix} \in \mathbb{R}^{2nN_p} \\ \tilde{u}(k) &= \begin{bmatrix} \bar{u}(k) \\ \bar{u}(k+1) \\ \vdots \\ \bar{u}(k+N_p) \end{bmatrix} \in \mathbb{R}^{mN_p} \end{aligned} \tag{4.15}$$

the objective function is expressed as:

$$J(k) = \begin{bmatrix} c_x & c_u \end{bmatrix}^T \begin{bmatrix} \tilde{x}(k) \\ \tilde{u}(k) \end{bmatrix} \tag{4.16}$$

where c_x contains σ and c_u contains ρ. Also the constraints (4.1)-(4.7) defining the events in the network can be formulated as a constraint matrix, by using the vectors $\tilde{x}$ and $\tilde{u}$:

$$\begin{bmatrix} A_x & A_u \end{bmatrix} \begin{bmatrix} \tilde{x}(k) \\ \tilde{u}(k) \end{bmatrix} \leq b(k) \tag{4.17}$$

where $k = 1, \ldots, N_p$ and the following holds: $[A_x]_{ij} \in \{-1, 0, 1\}$ and $[A_u]_{ij} \in \{-\beta, 0, \beta\}$, $\forall i, j$. The matrix $A = [\ A_x \quad A_u\ ]$ is called the *constraint matrix*. Vector $b(k)$ is determined as follows:

$$b(k) = b_0 + b_1\, x(k-1) + \theta(k) \tag{4.18}$$

where $[b_0]_i \in \{0, \beta\}$, $[b_1]_i \in \{0, 1\}$, $x(k-1)$ is the past value, and $\theta(k)$ is a vector containing the schedule times $r_d(k)$, $r_a(k)$, the minimum running time $t(k)$, the minimum dwell times $s(k)$, the minimum heading times $h(k)$ and the minimum separation times $w(k)$.

Considering the definition of an MILP problem from Section 2.3, we have now all the ingredients to recast the model predictive control problem into a MILP structure, where eq. (4.16) defines the objective function and ineq. (4.17) defines the constraints.

Note that for every delay scenario a feasible solution can be found. But, not every possible control combination results in a feasible timetable.

4.1.4 Introducing delays into the model

Let us consider a *delay scenario* defined by the parameter vector θ containing containing all the deviations from values in the original schedule described by θ_0 (see eq. (4.18)). With other words, a delay scenario is a specific realization of delays in the network. The parameter set of the delayed model is obtained as the element-wise sum of the two vectors: $\theta + \theta_0$. Numerical details about the θ_0 vectors used during simulations can be found in Section 4.3.

DOI:10.15774/PPKE.ITK.2014.002

To be able to simulate the perturbed operation, delays should be introduced into the model. Practically this is done by increasing the values corresponding to the dwell time or running time in case of departure or arrival delay, respectively. Let us detail an example in case of an arrival delay. Consider train j scheduled as a train run in the kth cycle by variables d_j and a_j referring to the departure and arrival time, respectively. The correspondence between the two values are given by ineq. (4.3) as follows: $a_j \geq d_j + t$ where t stands for the running time defined for the given train run. Considering ineq. (4.17) this will be reformulated as

$$a_j - d_j \geq t \tag{4.19}$$

where variables d_j and a_j are the elements of vector $\tilde{x}$, their coefficients (signs) are incorporated into A_x and t is the element of b corresponding to the specific constraint. Let us assume that t' delay is detected during train run j causing an increased running time. In this case ineq. (4.19) is modified as follows:

$$a_j - d_j \geq t + t' \tag{4.20}$$

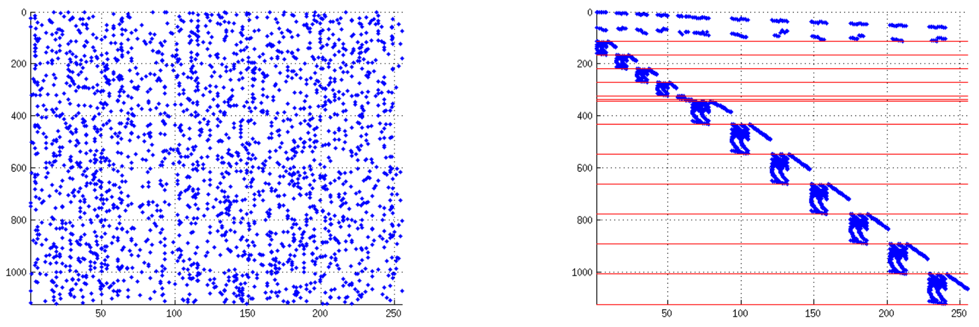
by introducing t' into θ and through it into the new b vector. Obtaining a new b vector triggers the re-computation of the MILP task resulting a new, feasible schedule which now incorporates the increased running time of train j and proper control actions if they are needed.

In case of dependent model formulation, the model corresponding to the given delay scenario can be computed by building up the whole constraint set from scratch. In case of the independent model the constraints depend only the on the structure of the network: once the constraint set is generated for a given railway network, the delay scenarios can be incorporated into the right-hand side of the constraints, namely into vector b. This means that in case of the independent model the regeneration of the whole model is unnecessary until the structure of the network is unchanged, only the parameter vector describing the delay scenario should be plugged into the model.

4.2 New solution methods of the railway scheduling problem to increase solution efficiency

In Subsection 4.1.2 a detailed description of the constraint set is presented which describes the correlations between the departure and arrival events in the network, considering the given references and the control inputs. In the following, two different transformation methods are proposed which lead to simplified constraint structures in the MILP model. The new constraint structures form the basis of the new efficient solution methods for the underlying MILP problem. The first method reorders the constraints so the the solution speed of the emerging MILP problem significantly improves. The second method reformulates the

DOI:10.15774/PPKE.ITK.2014.002



(a) Randomly ordered constraint matrix.

(b) Constraint matrix after the track-based ordering. Borders of the blocks are marked.

Figure 4.2: Structure of the constraint matrix where the variables are on the x-axis and the equations are on the y-axis. Non-zero elements of the matrices are marked. As it can be seen in Fig. 4.2b. the result of the track-based ordering is a clear block-angular structure.

constraints in a way that the dependence of the delays and the applied control actions can be easily computed. This formulation gives the opportunity to develop a problem-specific solution method in order to replace the general MILP solvers in the solution process.

4.2.1 Track-based transformation of the problem matrix

From the constraint formulations it can be seen that the order of the trains on two different tracks are independent from each other. To exploit this property, the reordering of the constraint matrix is completed on a per-track basis: all the constraints and control variables corresponding to a given track has been collected into one block [5]. The constraints in the different blocks are independent of each other and the blocks themselves (each corresponds to a track), and they are connected via the continuity constraints.

Let us recall that in Subsection 2.3.2.3 it is detailed that the computational effort needed to solve a given MILP problem strongly depends on the structure of the constraint matrix [51]. By investigating the corresponding methods it can be seen that a constraint matrix A with a clear block-angular structure has the most advantageous properties in terms of the complexity of the solution process. In case of the constraint set described in Subsection 4.1.2 and considering the results of [5], a proper reordering of the rows and columns of the constraint matrix can lead to a block-angular structure.

In Fig. 4.2 one can see the result of the track-based reordering method. The formulated blocks can be seen clearly. It should be noted that there could be other reordering methods which result a constraint set with block-angular structure, but the track-based method seems to be a quite straightforward approach. The increase in solution speed gained by the proposed reordering is detailed in Subsection 4.3.2.1. Note that the proposed reordering method still

DOI:10.15774/PPKE.ITK.2014.002

decreases the solution time while the preprocessor of the solver is also used.

4.2.2 Transformation of individual constraints

Let us refer to the constraints which depend on two different continuous variables as *dependent constraints* while those which depend only on one continuous variable as *independent constraints*. Note that both the dependent and the independent constraints can depend on multiple control variables.

The proposed algorithm transforms the dependent constraints into independent ones leading to a quite clear problem structure. The main advantages of the new structure are the following: firstly, the constraint matrix of a model containing purely independent constraints is independent of the given delay scenario. Hence, the formulation of the MILP problem corresponds to a given delay scenario requires only the computation of the vector $b(k)$, while the constraint matrix A remains the same, instead of computing new A matrix for every delay scenario as it is needed in case of dependent constraints are present. Note that this holds until the structure of the railway network remains unchanged (no blocked track etc.). Secondly, in case of independent constraints one specific event in the system - out of the binary control variables - has a direct dependence on only one other event. This fact gives the opportunity to investigate deeper the model structure which was not possible in case of the presence of dependent constraints.

We will exploit the special properties of the constraint set described in Subsection 4.1.2. Namely, the coefficients corresponding to the continuous variables can have only $\{-1, 0, 1\}$ values, and for each constraint only one 1 and one -1 value can be present. Furthermore, it is known that if a 1 appears, a -1 value is also present in that given constraint.

As it is mentioned before, the proposed algorithm transforms the dependent constraints into independent ones. This means that the A_x matrix from ineq. (4.17) is transformed to a matrix that will only contain $\{0, -1\}$ values and only one non-zero coefficient for each constraint.

The proposed model transformation method - which will be introduced next - is based on an iterative substitution of the constraints into each other resulting in new constraints. The procedure continues until the expected constraint structure is achieved. A procedural description of the proposed method is presented in Sec. Appendix C. To ease the understanding of the procedure, an illustrative example is also presented.

4.2.2.1 Reformulation of the model for the substitution

The model described in ineq. (4.17) is reshaped into an other form in order to ease the understanding the proposed procedure. Let us use the following notations: x_i refers to an event in the network represented by a continuous variable. The sign of x_i is related to the corresponding elements of A_x. r_j is a reference time of the jth event. Elements of τ are referring to numerical values calculated from the running times, dwell times, headway times and separation times and related to the elements of $b(k)$. u_l is a binary control variable and β is a large negative value appearing in A_u. $\bar{1}_m$ refers to a full-one vector having m rows,

DOI:10.15774/PPKE.ITK.2014.002

matrices denoted by letter E have a block structure while matrices denoted by letter A contain non-zero elements without specific pattern. Operation $\otimes$ refers to the *Kroenecker*-product.

The constraint set of the given model is ordered in two different ways resulting different model formulations. These formulations can be used implement the substitution of the constraints to formulate the iterative method. In the following, we define the two model formulations and the substitution procedure.

Model formulation 1 Let us order the constraints in the model based on the variables on their left-hand side. Now the following structure can be obtained:

$$\begin{bmatrix} E_1 \\ \vdots \\ E_n \end{bmatrix} x \geq \begin{bmatrix} A_1 \\ \vdots \\ A_n \end{bmatrix} x + \begin{bmatrix} B_1 \\ \vdots \\ B_n \end{bmatrix} u + \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} \tag{4.21}$$

which can be read as

$$E\,x \geq A\,x + B\,u + c \tag{4.22}$$

where m_i is the number of constraints in the model having x_i in their left-hand side, $E_i \in \{0,1\}^{m_i \times n}$, $A_i \in \{0,1\}^{m_i \times n}$, $B_i \in \{-\beta, 0, \beta\}^{m_i \times n}$, $c_i \in \mathbb{R}^{m_i}$ and $i = 1...n$.

It should be noted that this model formulation is related to the original model, it will not change during the substitution process.

Model formulation 2 During the iteration, constraints emerged from the model are ordered based on the right-side of the inequality constraints. Let us describe the model formulation in the $\ell = 0$ case where ℓ is the iteration counter. Now, the sub-matrices are denoted as follows:

$$\begin{bmatrix} \bar{A}_0^\ell \\ \bar{A}_1^\ell \\ \vdots \\ \bar{A}_n^\ell \end{bmatrix} x \geq \begin{bmatrix} \bar{E}_0^\ell \\ \bar{E}_1^\ell \\ \vdots \\ \bar{E}_n^\ell \end{bmatrix} x + \begin{bmatrix} \bar{B}_0^\ell \\ \bar{B}_1^\ell \\ \vdots \\ \bar{B}_n^\ell \end{bmatrix} u + \begin{bmatrix} \bar{c}_0^\ell \\ \bar{c}_1^\ell \\ \vdots \\ \bar{c}_n^\ell \end{bmatrix} \tag{4.23}$$

which can be rewritten as

$$\bar{A}^\ell x \geq \bar{E}^\ell x + \bar{B}^\ell u + \bar{c}^\ell \tag{4.24}$$

where $\bar{A}_i^\ell \in \{0,1\}^{\bar{m}_i^\ell \times n}$, $\bar{E}_i^\ell \in \{0,1\}^{\bar{m}_i^\ell \times n}$, $\bar{B}_i^\ell \in \{-\beta, 0, \beta\}^{\bar{m}_i^\ell \times n}$, $\bar{c}_i^\ell \in \mathbb{R}^{\bar{m}_i^\ell}$, $i = 0...n$. Note that in the ℓth iteration step $\bar{m}_0^\ell$ and $\bar{m}_i^\ell$ equals to the number of the independent and dependent constraints corresponding to x_i, respectively.

Let us note that this kind of model formulation groups the independent constraints together leading to the appearance of the sub-matrices indexed by 0 (namely A_0^ℓ, E_0^ℓ, B_0^ℓ and c_0^ℓ). In *Model formulation 1*, these constraints are incorporated into the E_i, A_i, B_i, c_i ($i = 1...n$) matrices, too. Also, it can be seen that a permutation matrix P exists for which

DOI:10.15774/PPKE.ITK.2014.002

the following holds: $P \cdot E = \bar{A}^\ell$, $P \cdot A = \bar{E}^\ell$, $P \cdot B = \bar{B}^\ell$ and $P \cdot C = \bar{C}^\ell$.

4.2.2.2 Substitution procedure

Now using the two forms of the model presented previously, an iterative substitution process is proposed to achieve a model structure containing only independent constraints.

It is known that for $i > 0$:

$$\bar{1}_{m_i} \otimes \bar{E}_i^\ell = \bar{1}_{\bar{m}_i^\ell} \otimes E_i$$

Use *Model formulation 2*:

$$\begin{bmatrix} \bar{A}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{A}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{A}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{A}_3^\ell \end{bmatrix} x \geq \begin{bmatrix} \bar{E}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{E}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{E}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{E}_3^\ell \end{bmatrix} x + \begin{bmatrix} \bar{B}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{B}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{B}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{B}_3^\ell \end{bmatrix} u + \begin{bmatrix} \bar{c}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{c}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{c}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{c}_3^\ell \end{bmatrix}$$

By using ineq. (4.22) we find

$$\begin{bmatrix} \bar{E}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{E}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{E}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{E}_3^\ell \end{bmatrix} x + \begin{bmatrix} \bar{B}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{B}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{B}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{B}_3^\ell \end{bmatrix} u + \begin{bmatrix} \bar{c}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{c}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{c}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{c}_3^\ell \end{bmatrix}$$

$$= \begin{bmatrix} \bar{E}_0^\ell \\ \bar{1}_{\bar{m}_1^\ell} \otimes E_1 \\ \bar{1}_{\bar{m}_2^\ell} \otimes E_2 \\ \bar{1}_{\bar{m}_3^\ell} \otimes E_3 \end{bmatrix} x + \begin{bmatrix} \bar{B}_0^\ell \\ \bar{1}_{m_1^\ell} \otimes \bar{B}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{B}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{B}_3^\ell \end{bmatrix} u + \begin{bmatrix} \bar{c}_0^\ell \\ \bar{1}_{m_1^\ell} \otimes \bar{c}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{c}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{c}_3^\ell \end{bmatrix}$$

$$= \begin{bmatrix} \bar{E}_0\, x \\ \bar{1}_{\bar{m}_1^\ell} \otimes E_1\, x \\ \bar{1}_{\bar{m}_2^\ell} \otimes E_2\, x \\ \bar{1}_{\bar{m}_3^\ell} \otimes E_3\, x \end{bmatrix} + \begin{bmatrix} \bar{B}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{B}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{B}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{B}_3^\ell \end{bmatrix} u + \begin{bmatrix} \bar{c}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{c}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{c}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{c}_3^\ell \end{bmatrix}$$

$$\geq \begin{bmatrix} \bar{E}_0^\ell\, x \\ (\bar{1}_{\bar{m}_1^\ell} \otimes A_1)\, x + (\bar{1}_{\bar{m}_1^\ell} \otimes B_1)\, u + (\bar{1}_{\bar{m}_1^\ell} \otimes c_1) \\ (\bar{1}_{\bar{m}_2^\ell} \otimes A_2)\, x + (\bar{1}_{\bar{m}_2^\ell} \otimes B_2)\, u + (\bar{1}_{\bar{m}_2^\ell} \otimes c_2) \\ (\bar{1}_{\bar{m}_3^\ell} \otimes A_3)\, x + (\bar{1}_{\bar{m}_3^\ell} \otimes B_3)\, u + (\bar{1}_{\bar{m}_3^\ell} \otimes c_3) \end{bmatrix} + \begin{bmatrix} \bar{1}_{m_0} \otimes \bar{B}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{B}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{B}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{B}_3^\ell \end{bmatrix} u + \begin{bmatrix} \bar{1}_{m_0} \otimes \bar{c}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{c}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{c}_2^\ell \\ \bar{1}_{m_3} \otimes \bar{c}_3^\ell \end{bmatrix}$$

$$= \begin{bmatrix} \bar{E}_0^\ell \\ (\bar{1}_{\bar{m}_1^\ell} \otimes A_1) \\ (\bar{1}_{\bar{m}_1^\ell} \otimes A_2) \\ (\bar{1}_{\bar{m}_1^\ell} \otimes A_3) \end{bmatrix} x + \begin{bmatrix} \bar{B}_0^\ell \\ (\bar{1}_{m_1} \otimes \bar{B}_1^\ell) + (\bar{1}_{\bar{m}_1^\ell} \otimes B_1) \\ (\bar{1}_{m_2} \otimes \bar{B}_2^\ell) + (\bar{1}_{\bar{m}_2^\ell} \otimes B_2) \\ (\bar{1}_{m_3} \otimes \bar{B}_3^\ell) + (\bar{1}_{\bar{m}_3^\ell} \otimes B_3) \end{bmatrix} u + \begin{bmatrix} \bar{c}_0^\ell \\ (\bar{1}_{m_1} \otimes \bar{c}_1^\ell) + (\bar{1}_{\bar{m}_1^\ell} \otimes c_1) \\ (\bar{1}_{m_2} \otimes \bar{c}_2^\ell) + (\bar{1}_{\bar{m}_2^\ell} \otimes c_2) \\ (\bar{1}_{m_3} \otimes \bar{c}_3^\ell) + (\bar{1}_{\bar{m}_3^\ell} \otimes c_3) \end{bmatrix}$$

DOI:10.15774/PPKE.ITK.2014.002

which results in the new inequality:

$$\hat{A}^\ell\, x \geq \hat{E}^\ell\, x + \hat{B}^\ell\, u + \hat{c}^\ell \tag{4.25}$$

where

$$\hat{A}^\ell = P \begin{bmatrix} \bar{A}_0^\ell \\ \bar{1}_{m_1} \otimes \bar{A}_1^\ell \\ \bar{1}_{m_2} \otimes \bar{A}_2 \\ \bar{1}_{m_3} \otimes \bar{A}_3^\ell \end{bmatrix} \qquad \hat{E}^\ell = P \begin{bmatrix} \bar{E}_0^\ell \\ (\bar{1}_{\bar{m}_1^\ell} \otimes A_1) \\ (\bar{1}_{\bar{m}_1^\ell} \otimes A_2) \\ (\bar{1}_{\bar{m}_1^\ell} \otimes A_3) \end{bmatrix}$$

$$\hat{B}^\ell = P \begin{bmatrix} \bar{B}_0^\ell \\ (\bar{1}_{m_1} \otimes \bar{B}_1^\ell) + (\bar{1}_{\bar{m}_1^\ell} \otimes B_1) \\ (\bar{1}_{m_2} \otimes \bar{B}_2^\ell) + (\bar{1}_{\bar{m}_2^\ell} \otimes B_2) \\ (\bar{1}_{m_3} \otimes \bar{B}_3^\ell) + (\bar{1}_{\bar{m}_3^\ell} \otimes B_3) \end{bmatrix} \qquad \hat{c}^\ell = P \begin{bmatrix} \bar{c}_0^\ell \\ (\bar{1}_{m_1} \otimes \bar{c}_1^\ell) + (\bar{1}_{\bar{m}_1^\ell} \otimes c_1) \\ (\bar{1}_{m_2} \otimes \bar{c}_2^\ell) + (\bar{1}_{\bar{m}_2^\ell} \otimes c_2) \\ (\bar{1}_{m_3} \otimes \bar{c}_3^\ell) + (\bar{1}_{\bar{m}_3^\ell} \otimes c_3) \end{bmatrix}$$

where P is again a proper permutation matrix such that $\bar{E}^\ell$ has the correct structure.

After computing ineq. (4.25), those inequalities, which contain terms $u_i\beta$ and $(1 - u_i)\beta$ for the same i should be removed from the constraint set. Furthermore, the constraints with the following form

$$x_i \geq x_i + B_i\, u + c_i$$

can be replaced with the following inequalities:

$$B_i\, u + c_i \leq 0.$$

Some of these inequalities can be removed immediately, because they lead to a control combination corresponding to an infeasible series of events. This is caused by the fact that in case of n control variables the number of possible combinations (2^n) is larger than the possible valid control input combinations. A simple example is the following: consider three trains running on the same track, ordered by three control variables. The number of possible control combinations is $2^3 = 8$ but there are only 6 feasible train orderings. For example, an infeasible train order appears if the first train is scheduled before the second, the second is scheduled before the third and the third is scheduled before the first. To overcome on this problem, the proposed algorithm filters out those constraints which contain infeasible control combinations. Note, that the reachability set of the system is not changed because only infeasible control combinations are removed.

Now an iteration step can be formulated as follows. In the ℓth step we consider ineq. (4.24) and ineq. (4.22) and obtain ineq. (4.25). Initialize the next step by letting $\bar{A}^{\ell+1} = \hat{A}^\ell$, $\bar{B}^{\ell+1} = \hat{B}^\ell$ and $\bar{c}^{\ell+1} = \hat{c}^\ell$. It can be seen that for each iteration step the following inequality holds: $row(\bar{A}_0^\ell) \leq row(\bar{A}_0^{\ell+1})$, where row stands for the number of rows in the given matrix. For each event there exists at least one independent constraint in the starting model (e.g.

DOI:10.15774/PPKE.ITK.2014.002

the timetable constraint), hence the process will finish with the elimination of the dependent constraints in finite steps: $\exists \ell : \bar{m}_i^\ell = 0$ where $i > 0$ and $\ell \leq n$.

As it is shown, the main purpose of this algorithm is to transform the originally dependent constraints into independent ones. It differs from other matrix decomposition or factoring methods because of its capability to handle the strongly over-determined constraint set with the restricted set of allowed operations on the constraints.

4.2.2.3 Example

Let us consider a small model containing three events. x_i denotes the time of a specific event while r_i denotes the reference time of the given event. τ_{ij} is a headway time defined between events x_i and x_j. Control variables are denoted by u_k and β is a large negative number.

$$\begin{aligned}
x_1 &\geq r_1 \\
x_2 &\geq r_2 \\
x_3 &\geq r_3 \\
x_2 &\geq x_1 + \tau_{21} + (1 - u_1)\beta \\
x_3 &\geq x_1 + \tau_{31} + (1 - u_3)\beta \\
x_1 &\geq x_2 + \tau_{12} + u_1\beta \\
x_3 &\geq x_2 + \tau_{32} + (1 - u_2)\beta \\
x_1 &\geq x_3 + \tau_{13} + u_3\beta \\
x_2 &\geq x_3 + \tau_{23} + u_2\beta
\end{aligned} \tag{4.26}$$

This constraint set will be reformulated with the proposed transformation and the resulting new constraint set is presented, too.

Model formulation 1:

$$\left[\begin{array}{ccc} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ \hline 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{array}\right] x \geq \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \hline 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ \hline 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{array}\right] x + \beta \left[\begin{array}{ccc} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ \hline 0 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 0 \end{array}\right] u + \left[\begin{array}{c} r_1 \\ \tau_{12} \\ \tau_{13} \\ \hline r_2 \\ \tau_{21} + \beta \\ \tau_{23} \\ \hline r_3 \\ \tau_{31} + \beta \\ \tau_{32} + \beta \end{array}\right]$$

Model formulation 2:

$$\left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \hline 0 & 1 & 0 \\ 0 & 0 & 1 \\ \hline 1 & 0 & 0 \\ 0 & 0 & 1 \\ \hline 1 & 0 & 0 \\ 0 & 1 & 0 \end{array}\right] x \geq \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline 1 & 0 & 0 \\ 1 & 0 & 0 \\ \hline 0 & 1 & 0 \\ 0 & 1 & 0 \\ \hline 0 & 0 & 1 \\ 0 & 0 & 1 \end{array}\right] x + \beta \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ \hline -1 & 0 & 0 \\ 0 & 0 & -1 \\ \hline 1 & 0 & 0 \\ 0 & -1 & 0 \\ \hline 0 & 0 & 1 \\ 0 & 1 & 0 \end{array}\right] u + \left[\begin{array}{c} r_1 \\ r_2 \\ r_3 \\ \hline \tau_{21} + \beta \\ \tau_{31} + \beta \\ \hline \tau_{12} \\ \tau_{32} \\ \hline \tau_{13} \\ \tau_{13} \end{array}\right]$$

Using the above introduced transformation procedure, the constraint set defined in ineq. (4.26) turned into the following set:

$$\begin{aligned}
x_1 &\geq r_1 \\
x_2 &\geq r_1 + \tau_{21} + (1-u_1)\beta \\
x_2 &\geq r_1 + \tau_{23} + \tau_{31} + u_2\beta + (1-u_3)\beta \\
x_3 &\geq r_1 + \tau_{31} + (1-u_3)\beta \\
x_3 &\geq r_1 + \tau_{32} + \tau_{21} + (1-u_1)\beta + (1-u_2)\beta \\
x_2 &\geq r_2 \\
x_1 &\geq r_2 + \tau_{12} + u_1\beta \\
x_1 &\geq r_2 + \tau_{13} + \tau_{32} + u_3\beta + (1-u_2)\beta \\
x_3 &\geq r_2 + \tau_{32} + (1-u_2)\beta \\
x_3 &\geq r_2 + \tau_{31} + \tau_{12} + u_1\beta + (1-u_3)\beta \\
x_3 &\geq r_3 \\
x_1 &\geq r_3 + \tau_{13} + u_3\beta \\
x_1 &\geq r_3 + \tau_{12} + \tau_{23} + u_1\beta + u_2\beta \\
x_2 &\geq r_3 + \tau_{23} + u_2\beta \\
x_2 &\geq r_3 + \tau_{13} + \tau_{21} + (1-u_1)\beta + u_3\beta \\
0 &> u_1\beta + u_2\beta + (1-u_3)\beta \\
0 &> (1-u_1)\beta + (1-u_2)\beta + u_3\beta
\end{aligned} \tag{4.27}$$

As it can be seen, ineq. (4.27) contains only independent constraints: on the right-hand side only control variables, reference times and headway times can be found (denoted by u_i, r_i and τ_{ij}, respectively).

4.2.2.4 Partial transformation of the model

The advantageous property of the model achieved by the procedure described in Subsection 4.2.2 - namely that all constraints are independent constraints - has the drawback that the number of the constraints in the model is increased dramatically. This has also effect on the running time of the transformation.

To overcome this, the algorithm is modified to process the constraint set on an iterative way. In one iteration a subset of events are selected and only those constraints are processed which correspond to these events in the network. By limiting the size of this subset, the computational requirements of an iteration and the resulting model size can be limited, too.

This kind of modification also means that the final model will contain some dependent constraints. These could not be translated into independent ones because some of the constraints needed for the substitution were out of the processed constraint set having limited size. Hence, the resulting model can be regarded as a partially independent model.

The partial transformation can preserve the advantageous properties of the model having only independent constraints (e.g. the dependencies between the events can keep a clear structure) while it overcomes the problems emerging if the constraint set is enlarged. It should be noted that the ratio of the remaining dependent constraints can be controlled by changing the size of the subset containing the events processed in one iteration step.

Computational results and comparisons between the original and transformed models can be found in Subsection 4.3.2.

4.2.2.5 Structure of the transformed constraint set

The constraint set composed in Subsection 4.1.2 can be transformed into a new constraint matrix by using the previously detailed algorithm. The resulting problem containing independent constraints shows a clear connection between the events in the network through solely the control variables.

As an example, a constraint set containing 23278 dependent constraints and 4769 variables (from which 1930 are continuous variables, the others are binary control variables) is converted to contain independent constraints. The number of the constraints increased to 520194. The structure of the constraint set can be seen in Fig. 4.3. The running time of the transformation algorithm was around 4300 seconds. Note, that this is an off-line transformation meaning that it should be completed once for each model until the model structure (the structure of the railway network and the routes and number of trains in it) is unchanged. An estimation for the number of the constraints in the independent model is possible but would take serious computational efforts, hence it strongly depends on the structure of the dependent model.

4.3 A case study

Simulations were done using the model of the Dutch railway system (see Fig. 4.4), with a

DOI:10.15774/PPKE.ITK.2014.002

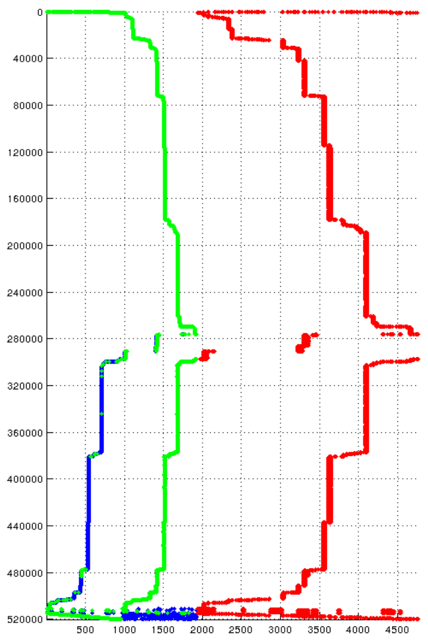


Figure 4.3: Structure of the constraint matrix containing independent constraints. Blue and green marks are for positive and negative values in the matrix, respectively. Red marks are for values corresponding to control variables. In this particular case the track-based ordering has not been used. The total size of the matrix is 520194 rows and 4769 columns.

DOI:10.15774/PPKE.ITK.2014.002

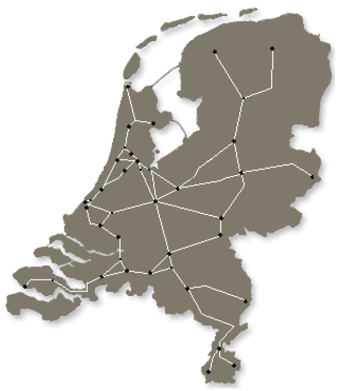


Figure 4.4: The Dutch railway network.

timetable having cycle time $T_c = 60$ *min*. The network model consists of 66 stations and 191 tracks. In the timetable the international, interregional and most of the local trains are included, but freight trains and some smaller lines that are not connected to the main network are excluded. For the simulations $N_p = 2$ is selected, so the prediction is done for 2 hours in the future. A new schedule is generated if any of the trains have more than 2 minutes of delay. The generated problem originally has 54090 constraints and 13136 variables from which 3860 are continuous variables (departure and arrival times) and the remaining 9276 are binary control variables. The time constants in the model are set as follows: headway times $h^a_{ij} = h^d_{ij} = 3\,min$, dwell time $s_{p_i}(k) = 2\,min$ and separation time $w_{ij} = 3\,min$.

The computations were completed on a personal computer equipped with a Pentium IV CPU having 4 cores, running on 3.4GHz. Model generation and transformations were done in MATLAB. To solve the generated MILP problem, the Gurobi 5.50 [59] solver was used with enabled multi-threaded solution capability. It should be mentioned that during the computations the default parameter settings of the solver were used in order to ensure the comparability of the results.

The delay scenarios were introduced into the model as it is described in Subsection 4.1.4. Each delay scenario contains several, individual initial delays, representing the time-shifting of an event (departure or arrival) from its reference time. The sum of the introduced initial delays by a given scenario is referred as the *total initial delay*. It is shown in [134] that the distribution of delays appearing in a train network follows a Weibull distribution. In our case the delays were generated randomly, according to a Weibull-distribution having shape parameter 0.8 and scale parameter 5. The maximum amount of an individual delay event was set to 10 *min* and the average introduced delay to 3 *min*. During the scenario generation a predefined percentage of the trains were selected to be delayed: the delay values were added to their departure and arrival data. In general, using the previously parametrized Weibull distribution, delaying 7% of the events ends up in a realistic delay scenario considering the absolute values and distribution of the delays. This means that 260 events were delayed over the prediction horizon, leading to 130 delayed event in an hour. Note that according to the

DOI:10.15774/PPKE.ITK.2014.002

infrastructure manager company of the Dutch Railways, in the first nine months of 2012, 10.5% of the trains in the Netherlands had 3 minutes or more delay.

The aim of the controller is to minimize the sum of the secondary delays over the prediction horizon. This is calculated as the sum of the deviations from the predefined timetable and referred as *total secondary delay* in case of a given delay scenario.

4.3.1 Performance analysis of the proposed control technique

The performance of the proposed control technique is showed by comparing the open loop and closed loop (uncontrolled and controlled, respectively) behavior of the railway network. The simulations were completed on 45 scenarios. In case of these scenarios, the average of the total initial delay was 3448.2 min with $s = 146.1$ min, where s stands for standard deviation.

The open-loop simulation is done by setting all control inputs to zero. This means that neither the order of the trains, nor the selected tracks or the coupled runs were changed. By simulating the operation of the network over the prediction horizon, the total secondary delay appeared during the uncontrolled behavior can be computed. The obtained values are the following: 4264.2 min total secondary delay in average, with $s = 549.7$ min.

In controlled mode the previously introduced control method was applied on the railway network. Again, at the end of the prediction horizon the difference between the actual timetable and the reference schedule is calculated as the total secondary delay. It is turned out that by applying the proposed control method the average total secondary delay has been reduced to 2164.2 min with $s = 238.6$ min.

The simulations showed the effectiveness of the proposed control technique as it can be seen in Fig. 4.5. The controller can decrease the the total secondary delay with approximately 48.9% in average compared to the uncontrolled case. In average, 48 control actions were applied.

Note, that considering a given delay scenario, selecting longer prediction horizons will increase the difference between the sum of the total secondary delays of the network in uncontrolled and controlled modes. Although, the computational time needed for the optimization process will increase, too. It should be also noted, that from a practical point of view and considering the frequency of unexpected events in the network, it does not worth to deal with prediction horizons longer than 3 hours.

4.3.2 Comparison of time consumption of solutions in case of different model formulations

Due to the large size of the emerging MILP problems in case of a busy railway network and long prediction horizon, time consumption of the rescheduling become a crucial problem. In the following, the model formulations detailed in Subsection 4.2.1 and 4.2.2 are compared in terms of the solution time.

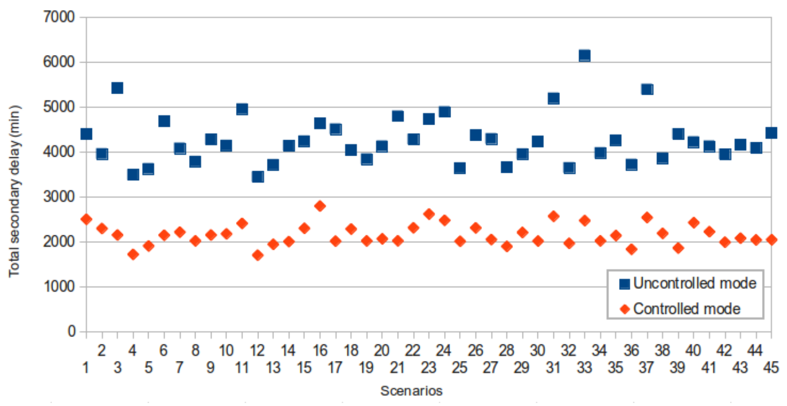


Figure 4.5: Effect of the proposed control technique while minimizing the total secondary delays. The optimal reordering of the trains can significantly reduce the total delay over the control horizon.

4.3.2.1 Effect of constraint reordering on the solution time

It is known that proper restructuring of an MILP problem can have serious impact on the solution time [51]. In the present work a track-based ordering is proposed (see Sec. 4.2.1) which results in a clear block-angular structure in the constraint matrices. The following results verify that even in the presence of the solver's preprocessor serious gains can be achieved using this specific ordering of the constraints.

During the generation of the constraint matrices, two different methods were compared. First the columns and rows of the formulated matrices were generated in a random order resulting an unordered matrix structure. In the other case we built up the matrices of the same problem with track-based ordering.

The solution times in both cases for several different scenario can be seen in Fig. 4.6. for 30 different scenarios. The average solution time was 53.763 sec ($s = 43.194$ sec) without and 29.674 sec ($s = 27.703$ sec) with reordering which means an average speedup ratio 1.813. It should be noted that according to the simulations larger problem sizes lead to larger speedup ratio. This emphasizes the importance of the structured problem formulation based on problem-specific knowledge. Note, that this reordering method outperforms all the preprocessing methods implemented in the applied solver meaning that the built-in methods were not as effective in the formulation of the block-diagonal structure as the proposed method was.

4.3.2.2 Effect of constraint reformulation on the solution time

In Subsection 4.2.2 a model transformation method is proposed. Now the effect of this transformation on the solution time is detailed. As it was mentioned before, the reformulation of the constraints from dependent to independent ones enlarges the problem matrix by increasing the number of constraints. This naturally causes some overhead during the

DOI:10.15774/PPKE.ITK.2014.002

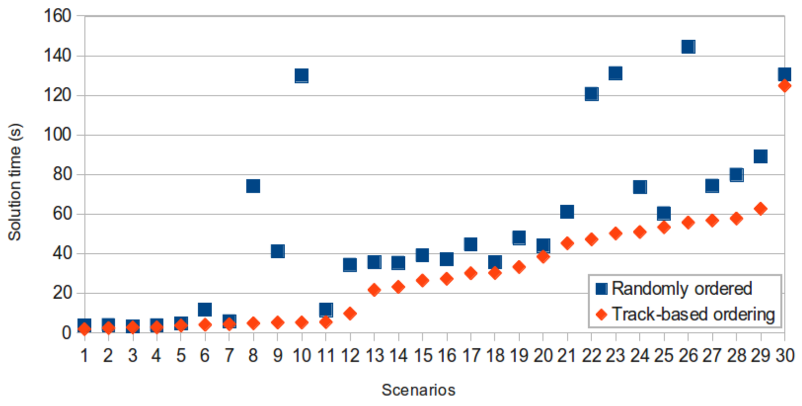


Figure 4.6: Solution times in case of a random constraint matrix structure and in case of the track-based reordering. The simulation results show that even in the presence of the preprocessor of the solver proper reordering can yield to significant speedup. Scenarios are plotted in an increasing order w.r.t. the solution time in case of track-based ordering.

optimization process. Hence, we expected that in case of a (partially) transformed model the solution will be slower than the original model.

In order to test the proposed methods in scenarios which can be considered as realistic ones, the following computations were completed. The dependent and independent models of 50 different, randomly generated scenarios were solved. In each scenario 7% of the trains were delayed which means that the average introduced total initial delay was 2790 min ($s = 108.8$ min). The average solution time in case of the dependent model was 853.91 sec ($s = 975.91$ sec) while in case of the independent model 1676.06 sec ($s = 2292.92$ sec) was achieved. The resulting solution times can be seen in Fig. 4.7. Due to the larger size of the independent model, in several cases it is slower to solve it than the dependent one. On the other hand, in case of some delay scenarios the dependent model can be solved slowly and the solution of the independent model (even its larger size) can gain on it. It should be also noted, that the solution speed of the independent model strongly depends on the number of redundant or unnecessary constraints which can be eliminated from the model by the solver's preprocessor, which is influenced by the values effected by the given delay scenario.

Further analysis were completed to investigate the changes in the solution time if the delay scenarios become more complex. 45 different, randomly generated scenarios were created while the percentage of the delayed trains was enabled to grow up to 15% (affecting 260 events per hour). This means that in average 3394 min ($s = 1702$ min) total initial delay was added to the models. From the 45 scenarios, 36 and 39 were solved successfully in case of the dependent and independent model formulation, respectively. In case of the remaining ones the solution process was run out of memory. The dependence between the introduced total initial delay and the solution times can be seen in Fig. 4.8. To ease the understanding of the results, trend lines were inserted into the diagram using power regression. The comparison of the solution times in case of the dependent and independent model formulation can be

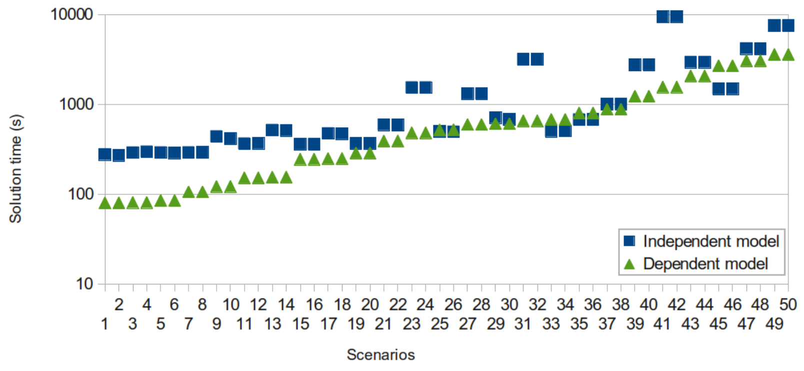


Figure 4.7: Solution times in case of models containing dependent and independent constraints. Scenarios are plotted in an increasing order w.r.t. the solution time in case of dependent models.

seen in Fig. 4.9. Note, that as the complexity of the delay scenarios are increasing (denoted by the increasing amount of primary delays) the effectiveness of the solver's preprocessor decreases. This means that less size reduction can be achieved in case of the independent model, resulting larger problem sizes and increased solution times.

It can be said that despite the fact that the independent model contains a lot more constraints than the dependent one, the solution times are in the same range in case of complex scenarios, too.

The main advantage of the formulated model is that it expresses the dependence of the events and the control variables in a very clear and simple way (see ineq. (4.27)): every continuous variable depends only on precomputable scalars and binary control variables. This also means that the dependence between the delays and the control variables can be directly formulated, by substracting the reference times from the time of the events, thus the effect of a given control variable on the evolution of the delays can be directly computed. By considering these, a problem-specific solution method can be developed to replace the general methods of the MILP solvers, which could have serious impact on the performance of the solution process. The development of this solver is out of the scope of this work but seems to be a very promising future work.

4.3.3 Sensitivity analysis based on single delays

It is known that in a railway network the change of the departure or arrival time of one train can a have small effect, but the change of the parameters of other trains can lead to an immense effect on the whole system. In the following it is shown that the proposed model structure is capable of showing the most delay-sensitive parts of the network.

To investigate the effect of a given train's delay on the network, the following setup was used. Two different cases were examined introducing 5 and 10 minutes of individual initial delay to a given train, respectively. Note that in case of this special setup the introduced total

DOI:10.15774/PPKE.ITK.2014.002

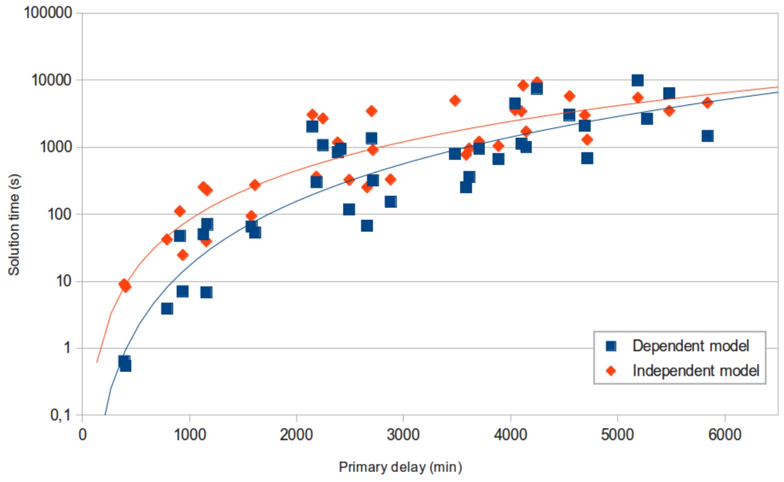


Figure 4.8: Dependence between the introduced total initial delays and the solution times in case of dependent and independent model formulation. The power regression lines has the following parameters and correlation coefficients. In case of the dependent model: $5 \cdot 10^{-9}x^{3.18}$, $R^2 = 0.831$, in case of the independent model: $4.155 \cdot 10^{-6}x^{2.43}$, $R^2 = 0.809$.

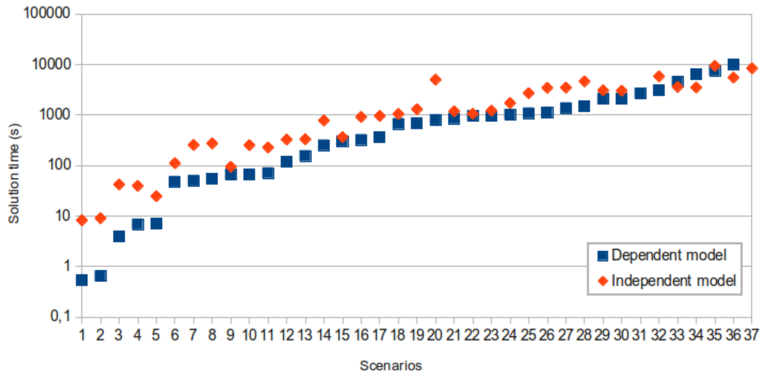


Figure 4.9: Comparison of the solution times of the dependent and independent model formulation in case of several different delay scenarios. Scenarios are plotted in an increasing order w.r.t. the solution time in case of dependent models. As it is shown in Fig. 4.8, this has a strong correlation with the total initial delay introduced into the given scenario.

DOI:10.15774/PPKE.ITK.2014.002

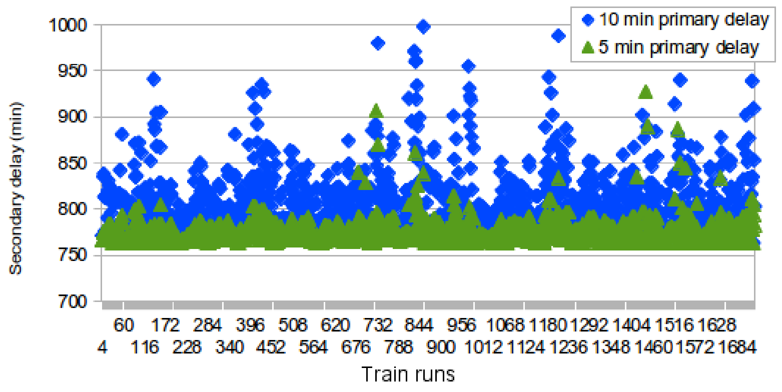


Figure 4.10: Result of delay-sensitivity tests in case of uncontrolled mode. Both the scenarios having 5 and 10 minutes initial delay are presented.

initial delay is equal to the individual delay of the given train. The whole network model with $N_p = 2$ was generated and simulated over the prediction horizon. Both uncontrolled case and controlled case were evaluated. This procedure was iterated over all trains in the network.

This setup enables us to analyze the effect of delay on a given train. If the selected train is a bottle-neck node in the network, then the introduced initial delays will end up in high secondary delay values, showing that many other trains are effected by the delay of this specific train.

The generated results can be seen in Fig. 4.10 and in Fig. 4.11. In case of the 5 min initial delay, the average of secondary delay in uncontrolled mode is 778.37 min ($s = 11.95\ min$) while in controlled mode 413.96 min ($s = 10.37\ min$) is achieved. In case of 10 min initial delay, the average of the secondary delays is 800.99 min ($s = 39.65\ min$) and 430.49 min ($s = 27.08\ min$) in uncontrolled and controlled case, respectively. The results are concurring with the expectations, namely that the effect of a relatively small delay (5 min) can be handled more effectively than the effect of a larger delay (10 min) because of the larger scale of delay propagation. These results also confirm that the proposed control technique can effectively reduce the amount of the delay in the network.

It should be noted that this algorithm works with the train runs in the model. As it is introduced in Subsection 4.1.1 a train run is a representation of a specific train over a given track. Handling train runs brings the advantage that besides being able to analyze the delay-sensitivity of a train we can also determine the most delay-sensitive track in the train's route.

4.4 Summary

In this Chapter a railway scheduling problem is formulated as an MILP and solved in an MPC architecture. The model is based on the one presented in [21]. The aim of the control method is to minimize the total delay of the trains over the prediction horizon. Due to the

DOI:10.15774/PPKE.ITK.2014.002

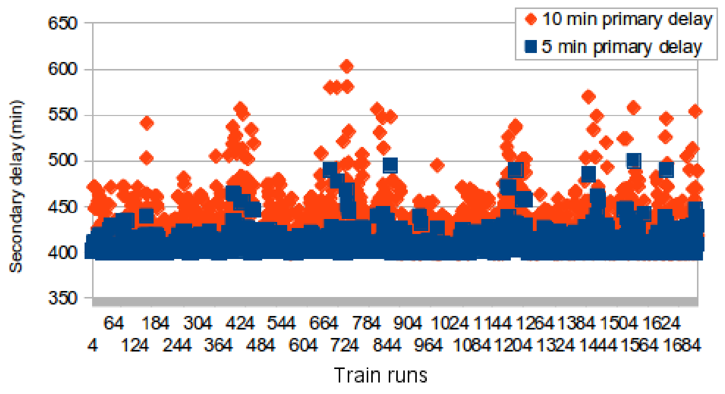


Figure 4.11: Result of delay-sensitivity tests in case of controlled mode. Both the scenarios having 5 and 10 minutes initial delay are represented. Delay values are clearly smaller than in Fig. 4.10.

computational complexity of the emerging MILP problem and the need of an algorithm which can solve the rescheduling in a reasonable time limit, it was necessary to speed up the solution with proper handling of the original problem. Hence, the obtained constraint set has been reformulated on two different ways, resulting in different model structures. The time consumption of the solutions have been compared to each other in case of both model forms, having different delay scenarios. The performance of the proposed control technique to minimize total secondary delays has been analyzed.

The main contributions presented in this Chapter and summarized in Thesis II. are the following: using a track-based ordering a significant speedup can be achieved during the solution process. By replacing the dependent constraints with independent ones, a much simpler constraint structure has been obtained which gives the opportunity of the deeper analysis of the dependencies between events and control actions in the network. The effectiveness of the proposed control technique is shown by extensive simulations: an average of 48.9% reduction can be achieved in the sum of delays. The proposed method is capable to simulate the effect of individual initial delays which enables us to determine the most critical parts of the railway network in terms of delays.

The general approach - namely the formulation of a scheduling problem as a MILP problem in an MPC framework - could be utilized in other control problems, e.g. the optimal control of complex nonlinear systems by using the piecewise affine approximation of the nonlinear system model.

DOI:10.15774/PPKE.ITK.2014.002

Chapter 5

Conclusions

In this thesis, optimization based methods were used to analyze and control networked systems in large scale, with complex nonlinear dynamics. New methods were presented both in the topic of the structural analysis of the Kinetic Reaction Networks and the solution of the scheduling problems of traffic networks.

The main effort has been focused on the analysis of the structure of the investigated problems. It has been shown that a proper problem representation can be obtained by exploiting the special structure of problem itself, which leads to the simplification of the emerging optimization problems (e.g. see Figs. 3.7-3.8 and Fig. 4.2 and their interpretation). In some cases LP problems can be formulated to substitute the original MILP problems and in other cases the MILP structure is transformed resulting in computational tasks that can be solved in reasonable time. Also, it has been shown that the methods can be applied on a parallel architecture.

5.1 New scientific contributions of the work (thesis points)

The new scientific results presented in this work are summarized in this Section. They are arranged in three thesis points as follows.

Thesis I. Numerically efficient algorithms to find sparse and dense realizations of kinetic reaction networks.

I have proposed two algorithms both based on linear programming (LP) having polynomial time complexity to compute dynamically equivalent alternative realizations of a kinetic reaction network (KRN). I have showed that with the help of the proposed methods alternative realizations of large scale, biologically motivated KRNs can be computed, too. The algorithms are compared with the mixed-integer linear programming (MILP) based algorithm available in the literature and the correctness of the solutions is shown. I have also concluded that the introduced new methods outperform the MILP-based solution in terms of the time consumption of the solution. (Section 3.2)

Corresponding publications: [1, 6]

DOI:10.15774/PPKE.ITK.2014.002

Thesis I.a

I have proposed an LP-based algorithm to compute dynamically equivalent realizations of a KRN containing minimal number of reactions. The so-called sparse realization of the reaction network is computed via the column-wise L1-norm minimization of the off-diagonal elements of the Kirchhoff-matrix. (Section 3.2.1)

Thesis I.b

I have proposed an LP-based algorithm to compute a dynamically equivalent realization of a KRN containing maximal number of reactions which is proven to contain all possible realizations of the reaction graph as a subgraph. The method to compute the so-called dense realization of the reaction network is based on the relaxation of the MILP-based method known from the literature: the column-wise sum of the introduced real-valued auxiliary variables corresponding to the off-diagonal elements of the Kirchhoff-matrix has been maximized. (Section 3.2.2)

Thesis II. New methods to compute weakly reversible and mass conserving realizations of kinetic reaction networks.

I have proposed new methods to compute dynamically equivalent and linearly conjugate alternative realizations of a kinetic reaction network while constraints in terms of the structural properties of the reaction graph and/or dynamical properties of the described system are present, too.

Corresponding publications: [2, 4]

Thesis II.a.

A new, linear programming-based method with polynomial time complexity is proposed to compute linearly conjugate, weakly reversible realizations of a kinetic reaction network (KRN). I have compared the method to other linear programming- and mixed-integer linear programming-based algorithms from the literature and it is shown that it outperforms all the others in terms of computational time, hence the algorithm is capable to handle large scale KRNs, too. (Section 3.3)

Thesis II.b.

I have proposed a mixed-integer linear programming-based algorithm to compute dynamically equivalent realizations of a kinetic reaction network with mass-conservation property.

DOI:10.15774/PPKE.ITK.2014.002

The correctness of the results was shown through examples taken from the literature. (Section 3.4)

Thesis III. New solution methods of scheduling problems in traffic networks.

I have proposed a model formulation method for model-predictive controllers which aim to deal with the scheduling problem of railway networks in case of delayed operation. The controller reorders the trains in order to minimize the total delay in the network over the prediction horizon. The model is described with the help of linear constraints. Thus, the controlling problem is formulated as a mixed-integer linear programming (MILP) problem. I have showed the effectiveness of the proposed control technique and a method is proposed for the sensitivity analysis of the model in case of single delays. (Chapter 4)

Corresponding publications: [5, 3]

Thesis III.a.

I have proposed a reordering method of the constraint matrix that can speed up the solution of the MILP problem in the presence of the solver's preprocessor, too. The method is based on the track-based reordering of the constraint matrix. The track-based reordering means that the constraints corresponding to a given track are collected into one block resulting that the constraint matrix of the emerging MILP problem has a clear block-angular structure. (Section 4.2.1)

Thesis III.b.

I have proposed an algorithm to reformulate the constraints in order to achieve a more simple model formulation. The resulting model shows a clear and simple correspondence between the continuous variables describing the schedule of the events in the network and the binary control variables. Considering these, further analysis of the internal relations of the network model can be done, while problem-specific solution methods can be developed instead of the application of the general-purpose MILP solvers. (Section 4.2.2)

5.2 Utilization of the presented results, further work

In this thesis the topic of the analysis and control of complex, nonlinear dynamical systems with networked structure has been investigated through two open problems. Namely, in Chapter 3 a set of new or improved methods were presented to compute dynamically equivalent or linearly conjugate KRNs with additional structural or dynamical constraints.

DOI:10.15774/PPKE.ITK.2014.002

The introduced algorithms give us the opportunity to analyze large scale, biologically relevant networks, too. In Chapter 4 new model formulations were proposed for the dynamical railway scheduling problem in case of delayed operation. Moreover, a model predictive controller-based framework was formulated to generate timetables corresponding to delay scenarios in order to minimize the total secondary delay in the network. New timetables are obtained from the nominal one by swapping trains, splitting joint trains or move them onto alternative parallel tracks. It has been shown, that by applying the presented model formulations, in one hand, the solution speed of the emerging optimization problems can be increased. On the other hand, with proper reformulation, a very clean model structure can be obtained which can serve as a basis of the development of problem-specific solvers.

There are several fields where the results of the presented work can be further developed. The implementation of the applied methods on many-core computing devices can lead to massive improvement in solution time which further extends the applicability of the introduced methods. It should be noted, that each presented algorithm can be embedded into a parallel framework: during the problem formulation parallelly solvable problems can be generated or the solution process of the underlying optimization problem (e.g. the tree-exploration phase of the MILP solution) can be parallelized.

A possible further work can be the application of problem-specific decomposition methods on the given optimization problems. As an example, a topographical decomposition method can be mentioned in case of the railway networks: if the original, large scale railway network would be split into several, weakly connected parts, the MILP problem having huge size would also be split into smaller problems connected to each other via boundary conditions. The emerged set of smaller size MILPs can be solved parallelly, leading to decreased solution time.

From a modeling point of view, both topics offer several possible directions of development: in case of KRNs the search for realizations with further structural and dynamical constraints can be implemented, such as realizations with minimal/zero deficiency, computing all possible dynamically equivalent realizations etc. Also a controller design method with static or dynamic extension of the network could be possible using the presented framework.

In the railway scheduling topic further extensions of the model describing the events in the railway network would be useful, introducing capacity limits of the stations, priorizing trains based on their types, handling track blocking etc. Moreover, it would be interesting to modify the framework to control passenger delays instead of train delays leading to a more passenger-friendly rescheduling method.

In general, it could be said that the optimization based control of analysis and control of complex dynamical systems having networked structures can be an interesting research topic in the future, too, leading to the deeper understanding of the behavior of large-scale systems incorporating many interconnected units.

DOI:10.15774/PPKE.ITK.2014.002

The Author's Publications

[1] **J. Rudan**, G. Szederkényi, and K. M. Hangos, "Efficiently computing alternative structures of large biochemical reaction networks using linear programming," *MATCH Commun. Math. Comput. Chem.*, vol. 71, pp. 71–92, 2014.

[2] **J. Rudan**, G. Szederkényi, K. M. Hangos, and T. Péni, "Polynomial time algorithms to determine weakly reversible realizations of chemical reaction networks," *Journal of Mathematical Chemistry*, pp. 1–19, 2014.

[3] B. Kersbergen, **J. Rudan**, T. van den Boom, and B. D. Schutter, "Railway traffic management using switching max-plus-linear systems," *Discrete Event Dynamic Systems*, submitted.

[4] **J. Rudan**, G. Szederkényi, and K. M. Hangos, "Computing dynamically equivalent realizations of biochemical reaction networks with mass conservation," in *11th international Conference of Numerical Analysis and Applied Mathematics 2013: ICNAAM 2013*, vol. 1558, pp. 2356–2359, AIP Conference Proceedings, 2013. ISBN: 978-0-7354-1184-5.

[5] **J. Rudan**, B. Kersbergen, T. van den Boom, and K. M. Hangos, "Performance analysis of MILP based model predictive control algorithms for dynamic railway scheduling," in *European Control Conference (ECC2013), July 17-19 2013, Zurich*, pp. 4562–4567, 2013.

[6] **J. Rudan**, G. Szederkényi, and K. M. Hangos, "Effectively computing dynamically equivalent structures of large biochemical reaction networks," in *International Conference on Bioinformatics and Computational Biology – BIOCOMP 2012, Bulgaria, Varna*, p. 80, 2012.

[7] **J. Rudan**, K. M. Hangos, and G. Szederkényi, "Analysis and supervisory control of a pressurizer using coloured Petri nets," in *9th European Workshop on Advanced Control and Diagnosis - ACD 2011*, pp. 1–6, 2011.

[8] G. Szederkényi, G. Lipták, **J. Rudan**, and K. M. Hangos, "Optimization-based design of kinetic feedbacks for nonnegative polynomial systems," in *IEEE 9th International Conference of Computational Cybernetics, July 8-10, Tihany, Hungary*, pp. 67–72, 2013. ISBN: 978-1-4799-0063-3.

DOI:10.15774/PPKE.ITK.2014.002

References

[9] I. F. S. G. A. Capone, G. Carello and F. Malucelli, "Solving a resource allocation problem in wireless mesh networks: A comparison between a CP-based and a classical column generation," *Networks*, vol. 55, pp. 221–233, 2010.

[10] S. Akle, O. Dalal, R. M. Fleming, M. Saunders, N. Taheri, and Y. Ye, "Existence of positive steady states for mass conserving and mass-action chemical reaction networks with a single terminal-linkage class," *arXiv preprint arXiv:1105.2359*, 2011.

[11] D. F. Anderson, "Boundedness of trajectories for weakly reversible, single linkage class reaction systems," *Journal of Mathematical Chemistry*, vol. 49, pp. 1–16, 2011. DOI: 10.1007/s10910-011-9886-4.

[12] D. Angeli, "A tutorial on chemical network dynamics," *European Journal of Control*, vol. 15, pp. 398–406, 2009.

[13] R. Aris and R. Mah, "Independence of chemical reactions," *Industrial & Engineering Chemistry Fundamentals*, vol. 2, no. 2, pp. 90–94, 1963.

[14] A.-L. Barabási and J. Frangos, *Linked: The New Science Of Networks Science Of Networks*. Basic Books, 2002.

[15] A. Barrat, M. Barthelemy, and A. Vespignani, *Dynamical processes on complex networks*, vol. 574. Cambridge University Press Cambridge, 2008.

[16] M. Bergner, A. Caprara, A. Ceselli, F. Furini, M. E. Lübbecke, E. Malaguti, and E. Traversi, "Automatic dantzig-wolfe reformulation of mixed integer programs."

[17] L. Bertacco, M. Fischetti, and A. Lodi, "A feasibility pump heuristic for general mixed-integer problems," Tech. Rep. Technical Report OR-05-5, University of Padova, Italy, 2005.

[18] T. Berthold, "Primal heuristics for mixed integer programs," Master's thesis, Technischen Universitat Berlin, 2006.

[19] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang, "Complex networks: Structure and dynamics," *Physics reports*, vol. 424, no. 4, pp. 175–308, 2006.

DOI:10.15774/PPKE.ITK.2014.002

[20] T. van den Boom and B. D. Schutter, “On a model predictive control algorithm for dynamic railway network management,” in *2nd International Seminar on Railway Operations Modelling and Analysis (Rail-Hannover2007)*, 2007.

[21] T. J. van den Boom, N. Weiss, W. Leune, R. M. Goverde, and B. D. Schutter, “A permutation-based algorithm to optimally reschedule trains in a railway traffic network,” in *IFAC World Congress*, 2011.

[22] B. Boros, “On the existence of the positive steady states of weakly reversible deficiency-one mass action systems,” *Mathematical Biosciences*, vol. 245, pp. 157–170, 2013.

[23] S. Boyd, L. El-Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in Systems and Control Theory.* SIAM Books, Philadelphia, PA, 1994.

[24] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press, 2004.

[25] S. P. Bradley, A. C. Hax, and T. L. Magnanti, *Applied Mathematical Programming.* Addison-Wesley, 1977.

[26] A. Caprara, M. Fischetti, and P. Toth, “Modeling and solving the train timetabling problem,” *Operations Research*, vol. 50, no. 5, pp. 851–861, 2002.

[27] V. Chellaboina, S. P. Bhat, W. M. Haddad, and D. S. Bernstein, “Modeling and analysis of mass-action kinetics – nonnegativity, realizability, reducibility, and semistability,” *IEEE Control Systems Magazine*, vol. 29, pp. 60–78, 2009.

[28] D.-S. Chen, R. G. Batson, and Y. Dang, *Applied integer programming: modeling and solution.* John Wiley & Sons, 2011.

[29] W. W. Chen, B. Schoeberl, P. J. Jasper, M. Niepel, U. B. Nielsen, D. A. Lauffenburger, and P. K. Sorger, “Input-output behavior of erbb signaling pathways as revealed by a mass action model trained against dynamic data,” *Molecular Systems Biology*, vol. 5, Jan. 2009.

[30] J. W. Chinneck, *Practical Optimization: A Gentle Introduction.* Carleton University, 2009.

[31] L. O. Chua and T. Roska, “The cnn paradigm,” *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, vol. 40, no. 3, pp. 147–156, 1993.

[32] F. Corman, “Real-time railway traffic management: Dispatching in complex, large and busy railway networks,” tech. rep., TRAIL Research School, 2010.

[33] A. D’Ariano and M. Pranzo, “An advanced real-time train dispatching system for minimizing the propagation of delays in a dispatching area under severe disturbances,” *Networks and Spatial Economics*, vol. 9, no. 1, pp. 63–84, 2009.

DOI:10.15774/PPKE.ITK.2014.002

[34] G. B. Dantzig, *Linear Programming and Extensions.* Princeton University Press, 1963.

[35] G. B. Dantzig and P. Wolfe, “Decomposition principle for linear programs,” *Operations research*, vol. 8, no. 1, pp. 101–111, 1960.

[36] E. Davison, “Connectability and structural controllability of composite systems,” *Automatica*, vol. 13, no. 2, pp. 109 – 123, 1977.

[37] J. Deng, C. Jones, M. Feinberg, and A. Nachman, “On the steady states of weakly reversible chemical reaction networks.” http://arxiv.org/abs/1111.2386, Nov. 2011.

[38] D. L. Donoho, “For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution,” *Comm. Pure Appl. Math*, vol. 59, pp. 797–829, 2004.

[39] D. L. Donoho and J. Tanner, “Sparse nonnegative solutions of underdetermined linear equations by linear programming,” in *Proceedings of the National Academy of Sciences*, pp. 9446–9451, 2005.

[40] D. L. Donoho and J. Tanner, “Sparse nonnegative solution of underdetermined linear equations by linear programming,” *Proc. of the National Academy of Sciences of the USA (PNAS)*, vol. 102, no. 27, pp. 9446–9451, 2005.

[41] H. Ehrmantraut and E. Rabinowitch, “Kinetics of hill reaction,” *Archives of biochemistry and biophysics*, vol. 38, no. 1, pp. 67–84, 1952.

[42] B. Elspas, “The theory of autonomous linear sequential networks,” *Circuit Theory, IRE Transactions on*, vol. 6, no. 1, pp. 45–60, 1959.

[43] P. Érdi and J. Tóth, *Mathematical Models of Chemical Reactions. Theory and Applications of Deterministic and Stochastic Models.* Manchester, Princeton: Manchester University Press, Princeton University Press, 1989.

[44] K. P. Eswaran and R. E. Tarjan, “Augmentation problems.,” *SIAM J. Comput.*, vol. 5, pp. 653–665, 1976.

[45] G. Farkas, “Kinetic lumping schemes,” *Chemical Engineering Science*, vol. 54, pp. 3909–3915, 1999.

[46] M. Feinberg, “Chemical reaction network structure and the stability of complex isothermal reactors - II. Multiple steady states for networks of deficiency one,” *Chemical Engineering Science*, vol. 43, pp. 1–25, 1988.

[47] M. Feinberg, “Chemical reaction network structure and the stability of complex isothermal reactors - I. The deficiency zero and deficiency one theorems,” *Chemical Engineering Science*, vol. 42 (10), pp. 2229–2268, 1987.

DOI:10.15774/PPKE.ITK.2014.002

[48] M. Feinberg, *Lectures on chemical reaction networks.* Notes of lectures given at the Mathematics Research Center, University of Wisconsin, 1979.

[49] F. Friedler, K. Tarján, Y. W. Huang, and L. T. Fan, "Graph-theoretic approach to process synthesis: polynomial algorithm for maximal structure generation," *Computer and Chemical Engineering*, vol. 17, pp. 929–942, 1993.

[50] F. Friedler, K. Tarján, Y. W. Huang, and L. T. Fan, "Graph-theoretic approach to process synthesis: axioms and theorems," *Chemical Engineering Science*, vol. 47, pp. 1973–1988, 1992.

[51] M. Galati, *Decomposition Methods for Integer Linear Programming.* PhD thesis, Lehigh University, 2010.

[52] M. V. Galati and T. K. Ralphs, "DIP: A framework for decomposition in integer programming," tech. rep., COR@L Laboratory, Lehigh University, 2012.

[53] C. E. Garcia, D. M. Prett, and M. Morari, "Model predictive control: theory and practice - a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[54] J. C. Geromel, J. Bernussou, and P. Peres, "Decentralized control through parameter space optimization," *Automatica*, vol. 30, no. 10, pp. 1565–1578, 1994.

[55] R. E. Gomory, "An algorithm for integer solutions to linear programs," *Recent advances in mathematical programming*, vol. 64, pp. 260–302, 1963.

[56] R. E. Gomory *et al.*, "Outline of an algorithm for integer solutions to linear programs," *Bulletin of the American Mathematical Society*, vol. 64, no. 5, pp. 275–278, 1958.

[57] R. M. Goverde, "A delay propagation algorithm for large-scale railway traffic networks," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 3, pp. 269 – 287, 2010. 11th {IFAC} Symposium: The Role of Control.

[58] R. M. Goverde, "Railway timetable stability analysis using max-plus system theory," *Transportation Research Part B: Methodological*, vol. 41, no. 2, pp. 179–201, 2007.

[59] I. Gurobi Optimization, "Gurobi optimizer reference manual," 2013.

[60] W. M. Haddad, V. Chellaboina, and Q. Hui, *Nonnegative and Compartmental Dynamical Systems.* Princeton University Press, 2010.

[61] K. M. Hangos and G. Szederkényi, "The effect of conservation on the dynamics of chemical reaction networks," in *IFAC Workshop on Thermodynamic Foundations of Mathematical Systems Theory, TFMST'13*, 2013.

DOI:10.15774/PPKE.ITK.2014.002

[62] V. Hárs and J. Tóth, "On the inverse problem of reaction kinetics," in *Qualitative Theory of Differential Equations* (M. Farkas and L. Hatvani, eds.), vol. 30 of *Coll. Math. Soc. J. Bolyai*, pp. 363–379, North-Holland, Amsterdam, 1981.

[63] L. H. Hartwell, J. J. Hopfield, S. Leibler, and A. W. Murray, "From molecular to modular cell biology," *Nature*, vol. 402, pp. C47–C52, 1999.

[64] S. Haykin, *Neural networks: a comprehensive foundation.* Prentice Hall PTR, 1994.

[65] B. Heidergott and R. d. Vries, "Towards a (max,+) control theory for public transportation networks," *Discrete Event Dynamic Systems*, vol. 11, pp. 371–398, Oct. 2001.

[66] L. B. Holder and D. J. Cook, "Graph-based data mining.," *Encyclopedia of data warehousing and mining*, vol. 2, pp. 943–949, 2009.

[67] F. Horn and R. Jackson, "General mass action kinetics," *Archive for Rational Mechanics and Analysis*, vol. 47, pp. 81–116, 1972.

[68] O. C. Imer, S. Yüksel, and T. Başar, "Optimal control of lti systems over unreliable communication links," *Automatica*, vol. 42, no. 9, pp. 1429–1439, 2006.

[69] S. M. Z. Iqbal, H. Grahn, and J. T. Krasemann, "A comparative evaluation of re-scheduling strategies for train dispatching during disturbances," *Proc. of the 13th Int'l Conf. on Design and Operation in Railway Engineering (Computers in Railways XIII)*, pp. 567–579, 2012.

[70] S. Iwnicki, *Handbook of railway vehicle dynamics.* CRC Press, 2006.

[71] N. Jamshidi and B. O. Palsson, "Mass action stoichiometric simulation models: Incorporating kinetics and regulation into stoichiometric models," *Biophysical Journal*, vol. 98, no. 2, pp. 175 – 185, 2010.

[72] P. A. Jensen and J. F. Bard, *Operations Research Models and Methods.* Wiley, 2003.

[73] M. D. Johnston and D. Siegel, "Linear conjugacy of chemical reaction networks," *Journal of Mathematical Chemistry*, vol. 49, pp. 1263–1282, 2011.

[74] M. D. Johnston, D. Siegel, and G. Szederkényi, "A linear programming approach to weak reversibility and linear conjugacy of chemical reaction networks," *Journal of Mathematical Chemistry*, vol. 50, pp. 274–288, 2012.

[75] M. D. Johnston, D. Siegel, and G. Szederkényi, "Dynamical equivalence and linear conjugacy of chemical reaction networks: new results and methods," *MATCH Commun. Math. Comput. Chem.*, vol. 68, pp. 443–468, 2012.

DOI:10.15774/PPKE.ITK.2014.002

[76] S. Jonnalagadda, B. Balagurunathan, and R. Srinivasan, “Graph theory augmented math programming approach to identify minimal reaction sets in metabolic networks,” *Computers & Chemical Engineering*, vol. 35, no. 11, pp. 2366 – 2377, 2011.

[77] N. Karmarkar, “A new polynomial-time algorithm for linear programming,” *Proceedings of the 16th Annual ACM Symposium on Theory of Computing*, pp. 302–11, 1984.

[78] C.-T. Lin, “Structural controllability,” *Automatic Control, IEEE Transactions on*, vol. 19, no. 3, pp. 201–208, 1974.

[79] Y.-Y. Liu, J.-J. Slotine, and A.-L. Barabási, “Controllability of complex networks,” *Nature*, vol. 473, no. 7346, pp. 167–173, 2011.

[80] L. Ljung, *System identification.* Springer, 1998.

[81] A. Lodi and J. T. Linderoth, *Encyclopedia for Operations Research and Management Science*, ch. MILP Software. Wiley, 2011.

[82] J. Löfberg, “YALMIP : A toolbox for modeling and optimization in MATLAB,” in *Proceedings of the CACSD Conference*, (Taipei, Taiwan), 2004.

[83] P. Makila and H. Toivonen, “Computational methods for parametric lq problems–a survey,” *Automatic Control, IEEE Transactions on*, vol. 32, no. 8, pp. 658–671, 1987.

[84] N. Megiddo, “On the complexity of linear programming,” *Advances in economic theory*, pp. 225–268, 1987.

[85] L. Michaelis and M. L. Menten, “Die kinetik der invertinwirkung,” *Biochem. z*, vol. 49, no. 333-369, p. 352, 1913.

[86] H. Minc, *Nonnegative Matrices.* J. Wiley, 1988.

[87] W. Mitkowski, “Dynamical properties of metzler systems,” *Technical Sciences*, vol. 56, no. 4, 2008.

[88] Y. E. Nesterov and A. S. Nemirovski, *Interior-Point Polynomial Algorithms in Convex Programming (Studies in Applied and Numerical Mathematics).* Society for Industrial Mathematics, 1994.

[89] M. Newman, A.-L. Barabasi, and D. J. Watts, *The structure and dynamics of networks.* Princeton University Press, 2006.

[90] M. E. Newman, “The structure and function of complex networks,” *SIAM review*, vol. 45, no. 2, pp. 167–256, 2003.

[91] B. A. Ogunnaike and W. H. Ray, *Process dynamics, modeling, and control*, vol. 9. Oxford University Press New York, 1994.

DOI:10.15774/PPKE.ITK.2014.002

[92] M. Padberg and G. Rinaldi, “A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems,” *SIAM Review*, vol. 33, pp. pp. 60–100, 1991.

[93] C. H. Papadimitriou, “On the complexity of integer programming,” *J. ACM*, vol. 28, pp. 765–768, Oct. 1981.

[94] L. C. Pimenta, V. Kumar, R. C. Mesquita, and G. A. Pereira, “Sensing and coverage for a network of heterogeneous robots,” in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pp. 3947–3952, IEEE, 2008.

[95] F. A. Potra and S. J. Wright, *Primal-dual interior-point methods.* Citeseer, 1997.

[96] C. J. Puccia and R. Levins, *Qualitative modeling of complex systems.* Harvard University Press Cambridge, 1985.

[97] E. Quaglietta, “A microscopic simulation model for supporting the design of railway systems: development and applications,” 2011.

[98] S. Raghavan, *The Next Wave in Computing, Optimization, and Decision Technologies*, ch. ‘A Note on Eswaran and Tarjan’s Algorithm for the Strong Connectivity Augmentation Problem’, pp. 19–26. Kluwer Academic Press, 2005.

[99] T. Ralphs and M. Galati, “Decomposition methods,” in *Encyclopedia of Operations Research and Management Science* (J. Cochran, ed.), Wiley, 2010.

[100] T. Ralphs and M. Galati, “Decomposition and dynamic cut generation in integer programming,” tech. rep., Lehigh University, 2003.

[101] T. K. Ralphs, *Parallel Combinatorial Optimization*, ch. Parallel Branch and Cut, pp. 53–101. Wiley, 2006.

[102] P. J. Ramadge and W. M. Wonham, “The control of discrete event systems,” *Proceedings of the IEEE*, vol. 77, no. 1, pp. 81–98, 1989.

[103] R. Raman and I. Grossmann, “Modelling and computational techniques for logic based integer programming,” *Computers and Chemical Engineering*, vol. 18, pp. 563–578, 1994.

[104] R. L. Rardin, *Optimization in operations research*, vol. 166. Prentice Hall New Jersey, 1998.

[105] J. Reeb, S. Leavengood, and O. S. U. E. Service, *Using the Simplex Method to Solve Linear Programming Maximization Problems.* EM (Oregon State University. Extension Service), Oregon State University Extension Service, 1998.

DOI:10.15774/PPKE.ITK.2014.002

[106] F. Rosenblatt, “The perceptron: a probabilistic model for information storage and organization in the brain.,” *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[107] T. Roska, “Cellular wave computers for nano-tera-scale technology - beyond boolean, spatial-temporal logic in million processor devices,” *Electronics letters*, vol. 43, no. 8, pp. 427–429, 2007.

[108] A. Ruszczyński, “An augmented lagrangian decomposition method for block diagonal linear programming problems,” *Operations Research Letters*, vol. 8, no. 5, pp. 287–294, 1989.

[109] N. Samardzija, L. D. Greller, and E. Wassermann, “Nonlinear chemical kinetic schemes derived from mechanical and electrical dynamical systems,” *Journal of Chemical Physics*, vol. 90 (4), pp. 2296–2304, 1989.

[110] A. Schöbel, “A model for the delay management problem based on mixed-integer-programming,” *Electr. Notes Theor. Comput. Sci.*, vol. 50, no. 1, pp. 1–10, 2001.

[111] A. Schrijver, *Theory of linear and integer programming.* John Wiley & Sons, 1998.

[112] D. Simon, *Optimal state estimation: Kalman, H infinity, and nonlinear approaches.* John Wiley & Sons, 2006.

[113] C. Song, S. Havlin, and H. A. Makse, “Self-similarity of complex networks,” *Nature*, vol. 433, no. 7024, pp. 392–395, 2005.

[114] E. Sontag, “Structure and stability of certain chemical networks and applications to the kinetic proofreading model of T-cell receptor signal transduction,” *IEEE Transactions on Automatic Control*, vol. 46, pp. 1028–1047, 2001.

[115] T. L. M. Stephen P. Bradley, Arnoldo C. Hax, *Applied Mathematical Programming.* Addison-Wesley, 1977.

[116] G. Szederk’enyi, G. Lipt’ak, J. Rudan, and K. Hangos, “Optimization-based design of kinetic feedbacks for nonnegative polynomial systems,” in *IEEE 9th International Conference of Computational Cybernetics, July 8-10, Tihany, Hungary*, pp. 67–72, 2013. ISBN: 978-1-4799-0063-3.

[117] G. Szederkényi, J. R. Banga, and A. A. Alonso, “CRNreals: a toolbox for distinguishability and identifiability analysis of biochemical reaction networks,” *Bioinformatics*, vol. 28, pp. 1549–1550, June 2012.

[118] G. Szederkényi, “Computing sparse and dense realizations of reaction kinetic systems,” *Journal of Mathematical Chemistry*, vol. 47, pp. 551–568, 2010.

DOI:10.15774/PPKE.ITK.2014.002

[119] G. Szederkényi, J. R. Banga, and A. A. Alonso, "Inference of complex biological networks: distinguishability issues and optimization-based solutions," *BMC Systems Biology*, vol. 5, p. 177, 2011.

[120] G. Szederkényi, K. Hangos, and A. Magyar, "On the time-reparametrization of quasi-polynomial systems," *Physics Letters A*, vol. 334, pp. 288–294, 2005.

[121] G. Szederkényi and K. M. Hangos, "Finding complex balanced and detailed balanced realizations of chemical reaction networks," *Journal of Mathematical Chemistry*, vol. 49, pp. 1163–1179, 2011.

[122] G. Szederkényi, K. M. Hangos, and T. Péni, "Maximal and minimal realizations of reaction kinetic systems: computation and properties," *MATCH Commun. Math. Comput. Chem.*, vol. 65, pp. 309–332, 2011.

[123] G. Szederkényi, K. M. Hangos, and Z. Tuza, "Finding weakly reversible realizations of chemical reaction networks using optimization," *MATCH Commun. Math. Comput. Chem.*, vol. 67, pp. 193–212, 2012.

[124] J. Törnquist and J. A. Persson, "N-tracked railway traffic re-scheduling during disturbances," *Transportation Research Part B: Methodological*, vol. 41, pp. 342–362, March 2007.

[125] K. Tamura, N. Tomii, and K. Sato, "An optimal rescheduling algorithm from passengers' viewpoint based on mixed integer programming formulation," in *5th International Seminar on Railway Operations Modelling and Analysis (IAROR), May 13-15 2013, Copenhagen*, 2013.

[126] A. Tarantola, *Inverse problem theory and methods for model parameter estimation.* siam, 2005.

[127] J. Tornquist Krasemann, "Design of an effective algorithm for fast response to the re-scheduling of railway traffic during disturbances," *Transportation Research Part C: Emerging Technologies*, 2011.

[128] Z. A. Tuza, G. Szederkényi, K. M. Hangos, and J. R. B. A. A. Alonso, "Computing all sparse kinetic structures for a Lorenz system using optimization methods," *International Journal of Bifurcation and Chaos*, vol. accepted, p. to appear, 2013.

[129] Z. Tuza, J. Rudan, and G. Szederkényi, "Developing an integrated software environment for mobile robot navigation and control," in *2010 International Conference on Indoor Positioning and Indoor Navigation, IPIN 2010; Zurich; 15 September 2010 through 17 September 2010*, p. no. 5647506 (on CD), 2010. ISBN: 978-142445864-6.

DOI:10.15774/PPKE.ITK.2014.002

[130] P. Varaiya, "Smart cars on smart roads: problems of control," *Automatic Control, IEEE Transactions on*, vol. 38, no. 2, pp. 195–207, 1993.

[131] T. Vicsek, *Fluctuations and scaling in biology.* Oxford University Press New York, 2001.

[132] T. Vicsek, M. F. Shlesinger, and M. Matsushita, *Fractals in Natural Sciences.* World Scientific, 1994.

[133] R. Watson, *Railway Scheduling. In: Handbook of Transport Systems and Traffic Control.* 2001.

[134] J. Yuan, "Capturing stochastic variations of train event times and process times with goodness-of-fit tests," tech. rep., Delft University of Technology, 2007.

[135] M. M. Zavlanos, A. A. Julius, S. P. Boyd, and G. J. Pappas, "Inferring stable genetic networks from steady-state data," *Automatica*, vol. 47, no. 6, pp. 1113–1122, 2011.

[136] N. A. H. Zuraida Alwadood, Adibah Shuib, "A review on quantitative models in railway rescheduling," in *International Journal of Scientific and Engineering Research*, vol. 3, 2012.

[137] *Cut Generation Library.*
https://projects.coin-or.org/Cgl.

[138] *CLP - Coin-or linear programming.*
https://projects.coin-or.org/Clp.

[139] *COmputational INfrastructure for Operations Research.*
http://coin-or.org.

[140] "IBM ILOG CPLEX Optimizer."
http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/.

[141] *BioModels Database - ErbB signalling pathway.*
http://www.ebi.ac.uk/compneur-srv/biomodels-main/publ-model.do?mid=BIOMD0000000255.

[142] *GLPK - GNU Linear Programming Toolkit.*
https://www.glpk.org.

[143] *Benchmarks for Optimization Software.*
http://plato.asu.edu/bench.html.

DOI:10.15774/PPKE.ITK.2014.002

Appendix

A. List of abbreviations

The abbreviations used in this thesis are listed below.

Notation	**Meaning**
KRN	Kinetic Reaction Network
CRN	Chemical Reaction Network
LP	Linear Programming
MILP	Mixed Integer Linear Programming
MPC	Model Predictive Control
WR	Weak Reversibility
LC	Linear Conjugacy

DOI:10.15774/PPKE.ITK.2014.002

B. List of notations

The most important notations used through the thesis are listed below.

Notation	**Meaning**
A	constraint matrix of an optimization problem
b	right-hand side of an optimization problem
c	cost function in an optimization problem
ϵ	small positive value ($\epsilon \ll 1$)
n	number of species in a given KRN
m	number of complexes in a given KRN
x_i	concentration of the ith specie
$\mathbf{x}$	state vector of a KRN containing all x_i, $i = 1, \ldots, n$
C_j	jth complex, $j = 1, ..., m$
$\alpha_{i,j}$	stoichiometric coefficient of the jth complex, $i = 1, \ldots, n$
$k_{j,l}$	reaction rate coefficient
Y	complex composition matrix $Y \in \mathbb{R}^{n \times m}$
A_k	Kirchhoff matrix of a KRN, $A_k \in \mathbb{R}^{m \times m}$
D	weighted, directed graph representing a KRN
V_d	set of vertices in the weighted, directed graph of a KRN
E_d	set of edges in the weighted, directed graph of a KRN
$\tilde{A}_k$	scaled Kirchhoff matrix of a KRN
$\mathbf{d}$	scaling vector for linear conjugacy, $\mathbf{d} \in \mathbb{R}^n$
T	diagonal transformation for linear conjugacy, $T = \text{diag}(\mathbf{d})$
$\mathbf{h}$	a strictly positive vector from $\ker(A_k)$
p_u	an upper bound
p_l	a lower bound
g	vector of scaled molecular weights
T_c	cycle time of a timetable of a railway network
N_p	prediction horizon in a model predictive framework
β	large negative number, e.g. $\beta = -200$
$d_i(k)$, $a_i(k)$	departure and arrival times of the ith train in the kth cycle
$r_i^d(k)$, $r_i^a(k)$	reference times for the departure and arrival of the given train
u	a binary valued control variable
$J(x)$	value of the objective function (performance index)

DOI:10.15774/PPKE.ITK.2014.002

C. Procedural description of the transformation method presented in Sec. 4.2.2

The substitution method presented in Subsec. 4.2.2.2 can be described in a procedural way as follows. Note, that $A_x(i, j)$ refers to the element of A_x in the ith row and jth column.

Procedure 1 for generating independent constraints

```
1. Pick the $l$th row from $A_x$ so that $A_x(l, j) = 1$.
2. Let $i$ is the position of a $-1$ in the $l$th row.
3. Search for all dependent constraint which has $-1$ in the $j$th column and collect
   them in set $\mathcal{S}$.
4. If set $\mathcal{S}$ is non-empty, than
   4.1. Let us iterate over the rows in $\mathcal{S}$ with index $m$.
   4.2. Select $m$th row from the set $\mathcal{S}$ having a 1 element at the position $k$
        and substitute it into the constraint in the $l$th row of $[A_x\, A_u]$. Set the
        corresponding element of $b$ to the proper value calculated from $b(m)$ and
        $b(l)$.
   4.3. If the new constraint emerged from the substitution contains infeasible
        control combination, remove it.
5. Else
   5.1. Select an independent constraint having a $-1$ in the $j$th column and
        substitute it into the $l$th constraint. Adjust the coefficients of $A_u$ and
        value of $b$ properly. Put the resulting constraint back to the constraint set.
6. Remove the $l$th row from $A_x$ and $A_u$.
7. If there is at least one 1 element in $A_x$ then
   7.1. If a 1 can be found in the $j$th column than jump to (1).
   7.2. If there is no 1 in the $j$th column than let us pick a new $j$ value and jump
        to (1).
8. If there is no any 1 element left in $A_x$ then let $\bar{A} = [A_x\, A_u]$ and $\bar{b} = b$ and
   terminate.
```