MATCH
Communications in Mathematical and in Computer Chemistry

MATCH Commun. Math. Comput. Chem. **77** (2017) 299-320

ISSN 0340 - 6253

A New Efficient Algorithm for Determining All Structurally Different Realizations of Kinetic Systems

Bernadett Ács[1], Gábor Szederkényi[1,2], Dávid Csercsik[1]

[1]*Pázmány Péter Catholic University, Faculty of Information Technology and Bionics, Práter u. 50/a, H-1083 Budapest, Hungary*

[2]*Systems and Control Laboratory, Institute for Computer Science and Control (MTA SZTAKI) of the Hungarian Academy of Sciences, Kende u. 13-17, H-1111 Budapest, Hungary*

`acs.bernadett@itk.ppke.hu, szederkenyi@itk.ppke.hu, csercsik@itk.ppke.hu`

(Received August 15, 2016)

Abstract

In this paper we present a novel algorithm for computing all possible reaction graph structures representing linearly conjugate realizations of a polynomial kinetic system assuming a fixed set of complexes. The computation is based on the repeated application of linear programming steps. The correctness of the method is formally proved. The approach is compared to the only solution known from the literature using two examples, and it is shown that the number of optimization steps and the overall execution time are significantly lower in the case of the proposed new method.

1 Introduction

Kinetic systems form a general class of nonlinear models with a wide range of application possibilities, such as the quantitative description of complex processes in (bio)chemistry, pharmacokinetics, compartmental systems, population and disease dynamics, economic or transportation problems [5, 41]. Therefore, as it was already suggested in [14], we consider the theory of kinetic systems as a general modelling framework for nonlinear and especially for nonnegative dynamics. Thus, thermodynamically realizable chemical reaction networks (CRNs) become the special case of general kinetic models. Using the abstraction of chemical complexes and reactions, a weighted directed graph called the reaction graph can be assigned to any realization of a kinetic dynamics showing the 'transformation' of complexes into each other [19].

Graph theory is a versatile tool for capturing and understanding the interactions between building blocks of natural or technical systems [2,15]. The emergence of chemical graph theory was primarily motivated by the study of the connections between molecular structure and chemical properties [16]. In this context, several connectivity indices have been proposed and analysed (see, e.g. [23, 33]) which have also inspired the solution of numerous interesting and deep mathematical problems. It has been clearly shown in the literature that different graph constructions corresponding to *dynamical models* often play a key role in the analysis of fundamental system properties or even in design tasks [28, 29, 38].

One of the most important aims of chemical reaction network theory (CRNT) is to establish relations between the reaction graph structure and the qualitative dynamical behaviour of kinetic systems, preferably without precisely knowing the rate coefficients (i.e. the edge weights of the reaction graph) [19]. This research line has brought numerous significant results such as the Deficiency Zero and Deficiency One Theorems [4,14]. In [6] algebraic and graph-structure-based conditions were given for the stability analysis of kinetic systems. A directed multigraph obtained from the Jacobian matrix of the kinetic ODE model was used for establishing conditions on the qualitative network dynamics in [27]. The so-called species-reaction graph was introduced and applied for the analysis of multiple equilibria in [8]. From a general systems theoretic point of view, it is important to mention recent fundamental results on the proof of the Global Attractor Conjecture saying that any complex balanced reaction network is globally stable [1, 7] . This advantageous property is also strongly related to the network structure, since complex balance implies weak reversibility [18, 19]. A procedure is given in [31] for the systematic reduction of reaction networks which is based on a special directed graph relating the species to be eliminated. By applying suitable model transformations, a wide class of dynamical systems can be written in kinetic form [11,32], therefore we might be able to study their properties through CRN representation.

However, as it was already shown through an example in [19], the solution of the inverse problem is generally non-unique in the sense that several different network structures and parametrizations may give rise to exactly the same kinetic differential equations. This phenomenon is called macro-equivalence, confoundability or dynamical equivalence in the literature, and it was first analysed theoretically in [9]. In [20], a procedure was described to symbolically compute a possible network structure (called the canonical network) for any kinetic system given in the form of polynomial differential equations. The notion of dynamical equivalence was extended by the introduction of linear conjugacy in [21] by allowing a positive diagonal transformation between the solutions of linearly conjugate networks. Using the computationally advantageous structure of polynomial kinetic models, optimization-based numerical methods were proposed for the computation of dynamically equivalent or linearly conjugate reaction networks with preferred properties in e.g. [22, 24, 30, 34, 36, 39]. Besides the proposed numerical methods, the previously mentioned publications also illustrate through the presented examples that important properties such as weak reversibility, deficiency, complex or detailed balance may vary among the different linearly conjugate structures, even if the set of complexes is fixed.

The first provably correct solution known by the authors for the complete generation of reaction graph structures linearly conjugate to a given dynamics was published in [40]. Due to the possible large number of network structures, it is of interest to construct new solutions for this problem preferably with more advantageous computational properties. The purpose of this paper is to propose such a novel algorithm.

2 Notations and computational background

In this section we summarize the definitions of kinetic systems and their models. These are all standard notations, therefore they are similar to other works in this topic such as [13, 41], or the related work [40]. We apply the following general notations in this paper:

$\mathbb{R}$	the set of real numbers,
$\mathbb{R}_+$	the set of nonnegative real numbers,
$\mathbb{N}$	the set of natural numbers including 0,
$H^{n\times m}$	the set of matrices having entries from a set H with n rows and m columns,
$[M]_{ij}$	the entry of a matrix M with row index i and column index j.

2.1 Algebraic and dynamical description

Nonnegative polynomial systems can be defined in the following general form:

$$\dot{x} = M \cdot \varphi(x), \tag{1}$$

where $x : \mathbb{R} \to \mathbb{R}^n_+$ represents a nonnegative function, the entries of matrix $M \in \mathbb{R}^{n\times p}$ are the coefficients of the monomials defined by the mapping $\varphi : \mathbb{R}^n_+ \to \mathbb{R}^p_+$, and the coordinate functions are of the form $\varphi_i(x) = \sum_{j=1}^n x_j^{\beta_{ij}}$ with $\beta_{ij} \in \mathbb{N}$ for all $i \in \{1,\dots,p\}$ and $j \in \{1,\dots,n\}$. We remark that it is very easy to check the invariance of the nonnegative orthant with respect to the dynamics (1) using the signs of the entries of matrix M [17].

We define chemical reaction networks in the following form as it is common in the literature [41].

Definition 2.1. *Chemical reaction networks can be defined by three sets* (see, e.g. [13,14]).

species: $\mathcal{S} = \{X_i \mid i \in \{1,\dots,n\}\}$

complexes: $\mathcal{C} = \{C_j = \sum\limits_{i=1}^{n} \alpha_{ji} X_i \mid \alpha_{ji} \in \mathbb{N},\ j \in \{1,\dots,m\},\ i \in \{1,\dots,n\}\}$

reactions: $\mathcal{R} \subseteq \{(C_i, C_j) \mid C_i, C_j \in \mathcal{C}\}$

The ordered pair (C_i, C_j) for $i, j \in \{1,\dots m\}$, $i \neq j$ represents the reaction $C_i \to C_j$, which corresponds to a ***reaction rate coefficient*** *$k_{ij} \in \mathbb{R}^+$. The reaction is present in the reaction network if and only if k_{ij} is strictly positive.*

The nonnegative integer stoichiometric coefficients α_{ij} are contained by the **complex composition matrix** $Y \in \mathbb{N}^{n\times m}$ as follows

$$[Y]_{ij} = \alpha_{ji} \qquad i \in \{1,\dots,n\},\ j \in \{1,\dots,m\}.$$

The reactions are represented through the reaction rate coefficients by the **Kirchhoff matrix** $A_k \in \mathbb{R}^{m\times m}$ of the CRN, which is a Metzler-matrix with column conservation property.

$$[A_k]_{ij} = \begin{cases} k_{ji} & \text{if } i \neq j \\ -\sum\limits_{l=1, l\neq i}^{m} k_{il} & \text{if } i = j \end{cases}$$

The state vector $x : \mathbb{R} \to \mathbb{R}^n_+$ represents the time varying concentrations of the species, i.e. $x_i(t) = [X_i](t)$ for $i = 1,\dots,n$ and $t \geq 0$. From now on, the argument t of x will be suppressed. Assuming mass-action kinetics, the ODEs of kinetic models take the form [13]:

$$\dot{x} = Y \cdot A_k \cdot \psi^Y(x) \tag{2}$$

where $\psi^Y : \mathbb{R}^n_+ \to \mathbb{R}^m_+$ is the monomial function of the CRN defined by the complex composition matrix. Its coordinate functions are $\psi^Y_j(x) = \prod\limits_{i=1}^{n} x_i^{Y_{ij}}$ for all $j \in \{1,\dots,m\}$.

It can be seen that the structural and dynamical properties of a CRN are uniquely defined by the matrices Y and A_k, therefore we refer to a reaction network by the corresponding matrix pair (Y, A_k). The equations characterizing the dynamics of a reaction network are in the form of a polynomial system, but naturally, not every polynomial system is kinetic.

Definition 2.2. *A polynomial system of the form* (1) *defined by a function* $x : \mathbb{R} \to \mathbb{R}^n_+$, *a coefficient matrix* $M \in \mathbb{R}^{n \times p}$ *and a monomial function* $\varphi : \mathbb{R}^n_+ \to \mathbb{R}^p_+$ *is called* ***kinetic*** *if and only if there exists a chemical reaction network* (Y, A_k) *so that the following equation holds:*

$$M \cdot \varphi(x) = Y \cdot A_k \cdot \psi^Y(x). \tag{3}$$

For the kinetic property of a general polynomial system, necessary and sufficient conditions were first given in [20]. If the polynomial system (1) is kinetic and the CRN (Y, A_k) fulfills Equation (3), then this CRN is called a **dynamically equivalent realization** of the kinetic system (1). In general, a kinetic system has several dynamically equivalent realizations, reaction networks with different sets of reactions and complexes can be governed by the same dynamics [9, 19, 34].

The notion of dynamical equivalence can be extended to the case when the state variables are subject to a positive definite diagonal linear transformation [21]. In such a case the kinetic property of the system is preserved [12]. The transformation is given by the positive definite diagonal matrix $T \in \mathbb{R}^{n \times n}$, and $\bar{x} = T^{-1} \cdot x$ denotes the transformed state vector (i.e. $x = T \cdot \bar{x}$). By applying the diagonal transformation to the polynomial system (1) we get

$$\dot{\bar{x}} = T^{-1} \cdot \dot{x} = T^{-1} \cdot M \cdot \varphi(x) = T^{-1} \cdot M \cdot \varphi(T \cdot \bar{x}) = T^{-1} \cdot M \cdot \Phi_T \cdot \varphi(\bar{x}), \tag{4}$$

where $\Phi_T \in \mathbb{R}^{n \times n}$ is a positive definite diagonal matrix, the diagonal entries of which are $[\Phi_T]_{ii} = \varphi_i(T \cdot \mathbf{1})$ for $i \in \{1, \dots, n\}$. The vector $\mathbf{1} \in \mathbb{R}^n$ is a column vector with all coordinates equal to 1. Using Equation (4) we can define linear conjugacy.

Definition 2.3. *A reaction network* (Y, A_k) *is called a* ***linearly conjugate realization*** *of the kinetic system* (1) *if there exists a positive definite diagonal matrix* $T \in \mathbb{R}^{n \times n}$ *so that*

$$Y \cdot A_k \cdot \psi^Y(x) = T^{-1} \cdot M \cdot \Phi_T \cdot \varphi(x). \tag{5}$$

It can be seen that dynamical equivalence is a special case of linear conjugacy, where the matrix T, and consequently the matrices T^{-1} and Φ_T as well are identity matrices. It is also visible that the monomial functions φ and ψ^Y in Equations (3) and (5) might be different. A suitable set of complexes for a kinetic system can be determined by applying the method described in [20], and the description of the polynomial system (1) can be modified so that the monomial functions φ and ψ^Y are equivalent, and p is equal to m [39]. Assuming this (i.e. $\psi^Y = \varphi$), Equation (5) can be written as

$$Y \cdot A_k \cdot \psi^Y(x) = T^{-1} \cdot M \cdot \Phi_T \cdot \psi^Y(x). \tag{6}$$

It follows from the property of polynomials that for Equation (6) to hold, the corresponding monomial coefficients must be equal. Using this fact and the notation

$$A_b = A_k \cdot {\Phi_T}^{-1}, \tag{7}$$

we can rewrite Equation (5) as:

$$Y \cdot A_b = T^{-1} \cdot M \tag{8}$$

The matrix $A_b \in \mathbb{R}^{m \times m}$ is obtained by scaling the columns of A_k by positive scalars, therefore it has the Kirchhoff property, too, and the two matrices represent the same set of reactions in the CRN.

Since, according to our assumptions the complex composition matrix Y is fixed, linearly conjugate realizations can be characterized by the corresponding matrices T and A_k. However, in Equation (8) the matrices T^{-1} and A_b describe the CRN. The two pairs of matrices can uniquely be transformed into each other, therefore, with some abuse of notation we also refer to linearly conjugate realizations as the matrix pair (T^{-1}, A_b).

2.2 Graph representation

Chemical reaction networks can be represented as edge-weighted directed graphs [13].

Definition 2.4. *The graph* $G(V, E, w)$ *describing the CRN with weight function* $w : E(G) \to \mathbb{R}_+$ *is called* ***Feinberg-Horn-Jackson graph****, or* ***weighted reaction graph*** *if*

$V(G) = \mathcal{C}$	*the* ***vertices*** *correspond to the complexes,*
$E(G) = \mathcal{R}$	*the* ***directed edges*** *represent the reactions,*
$w((C_i, C_j)) = k_{ij}$	*the* ***weights*** *are the reaction rate coefficients.*

There is a directed edge from vertex C_i *to vertex* C_j *if and only if the reaction* $C_i \to C_j$ *takes place in the CRN. No loops or multiple edges (with identical directions) are allowed in the graph.*

It is possible that there are two realizations of a kinetic system, where the sets of reactions are the same, but the reaction rates are different. We call such realizations **structurally identical**, since the reaction graphs representing them have identical sets of edges, consequently they have the same structure. Our aim is to determine all the possible reaction graph structures, therefore in this paper we simply refer to unweighted reaction graphs as **reaction graphs**. During the computations we calculate the matrix A_b in each step, but the values of its entries need not to be stored. It is enough to record the presence or lack of reactions indicated by the non-zero or zero entries of A_b, respectively. This allows us to represent reaction graphs in the form of binary strings. Of course, if the available storage space allows, A_b (or A_k) and the transformation parameter T may also be recorded during the computation.

The correctness of the algorithm presented in this paper relies on a special property of the so called dense realizations which is summarized below, based on [39].

Definition 2.5. *A realization of a CRN is called a* ***dense realization*** *if the maximum number of reactions take place.*

Dense realizations can be defined in case of dynamically equivalent and linearly conjugate realizations, even if certain constraints need to be fulfilled by all the considered reaction networks.

Let $\mathcal{G}$ be a set of directed graphs with labelled vertices. A directed graph is called the **super-structure** with respect to the set $\mathcal{G}$ if it contains every graph in the set as a subgraph and it is minimal under inclusion. If there are two graphs that contain every graph in $\mathcal{G}$, then the intersection of these graphs also contains all graphs in $\mathcal{G}$. Consequently, in case of any set $\mathcal{G}$ the super-structure is unique, since it is the intersection of all graphs that contain every graph in $\mathcal{G}$.

Proposition 2.6. [39] *Among all the realizations linearly conjugate to a given kinetic system on a fixed set of complexes and fulfilling a finite set of additional linear constraints, the dense realization with the prescribed properties determines a super-structure.*

2.3 Linear programming based computational model

Linearly conjugate realizations of a kinetic system with a fixed set of complexes can be computed using a linear optimization model. It was described in Section 2.1 that in case of linearly conjugate realizations the linear constraint in Equation (8) must be fulfilled, where the known parameters are the matrices Y and M, and the decision variables are the off-diagonal entries of A_b and the diagonal entries of T^{-1}. The zero column-sums of A_b can be given as the following linear constraint

$$[A_b]_{ii} = -\sum_{\substack{j=1\\ j\neq i}}^{m} [A_b]_{ji} \qquad i \in \{1,\dots,m\}. \tag{9}$$

While the sign constraints for the decision variables are

$$[A_b]_{ij} \geq 0 \qquad\qquad i,j \in \{1,\dots,m\},\ i \neq j \tag{10}$$

$$[T^{-1}]_{ii} > 0 \qquad\qquad i \in \{1,\dots,n\}. \tag{11}$$

In summary, Equation (8) ensures linear conjugacy, and Equations (9), (10) and (11) guarantee that the matrices A_b and T^{-1} meet their definitions. In our method, additional linear constraints are applied only for the exclusion of some set $\mathcal{H} \subset \mathcal{R}$ of reactions, so they can be written as

$$[A_b]_{ji} = 0 \qquad (C_i, C_j) \in \mathcal{H} \tag{12}$$

It is important to mention that the boundedness property of the optimization variables can be ensured so that the set of possible reaction graphs remains the same as in the original problem, as it was proven in [39]. For convenience, we recall this result.

Proposition 2.7. [39] *For any linearly conjugate realization* (T^{-1}, A_b) *of a kinetic system there is another linearly conjugate realization* $((T')^{-1}, A'_b)$ *with all variables smaller than the given upper bound(s) so that the two realizations are structurally identical.*

During the computation, we need to determine constrained dense linearly conjugate realizations repeatedly. For this, we use a reliable and numerically stable polynomial-time method presented in [39], which applies linear programming steps.

3 Description of the proposed algorithm

In this section we introduce an algorithm for determining all possible reaction graph structures representing linearly conjugate realizations of a kinetic system with a fixed set of complexes.

The inputs of the algorithm are the matrices Y and M, which determine the set of complexes and the dynamics of the kinetic system, respectively. As outputs we compute the structures of linearly conjugate realizations, represented as binary sequences. During the computation the matrices T^{-1} and A_b encoding the properties of the realizations are calculated, as it was described in Section 2.3, but the algorithm stores only the sequences.

For simplicity, we refer both to a realization and the corresponding binary sequence as R, and $R[i]$ indicates the ith coordinate of the sequence. The reaction graph representing the realization R and the corresponding set of edges is denoted by G_R and $E(G_R)$, respectively. Since the dense realization has a special role, it has a particular notation as D.

According to the superstructure property, described in Proposition 2.6, during the binary sequence representation we need to consider only the set $E(G_D)$ of edges in the dense realization. It can be computed using a polynomial-time method, as it was mentioned in Section 2.3. This is the first step of our method.

It is possible that there are so-called **core reactions**, which are present in every realization of the kinetic system (see [3]), therefore these also do not require representation. The set E_c of edges corresponding to the core reactions can also be determined in polynomial time, see e.g. [35, 37]. It is not absolutely necessary doing this computation step, however it might save some computation time and space by reducing the length of binary sequences.

It is clear that the reaction graph of any realization can be uniquely characterized by a binary sequence of length

$$q = |E(G_D) \setminus E_c|. \tag{13}$$

Next we need to define an ordering of the non-core edges of the dense realization. The ordering can be chosen arbitrarily, it has no effect on the working of the algorithm. Let e_i denote the ith edge, then the sequence R can be defined as follows:

$$e \in E(G_R) \Longleftrightarrow \begin{cases} \exists i \ e = e_i, \ R[i] = 1 \\ \quad \text{or} \\ e \in E_c \end{cases} \tag{14}$$

Since the initial substrings of the sequences have an essential role in our method, we define special equivalence relations according to them. For all $k \in \{1, \ldots, q\}$ we say $R =_k R'$ if and only if for all $i \in \{1, \ldots, k\}$ $R[i] = R'[i]$ holds. The equivalence class of the relation $=_k$ containing the sequence R is denoted by $C_k(R)$. (We have to note that in general there are several possible representing elements of the equivalence classes.)

During the algorithm, as predefined data structures, we apply $q + 2$ pieces of stacks, indexed from 0 to $q + 1$. The stack with index k is referred to as $S(k)$. Graph structures represented by their sequences are stored in these stacks, following the rule: The sequence R may be in stack $S(k)$ only if R represents the dense realization in $C_k(R)$. However, it is not true that for a given k and R there is a point during the running of the algorithm, when the dense realization in $C_k(R)$ is stored in $S(k)$.

According to the above description the stacks $S(0)$, $S(q)$ and $S(q+1)$ have no immediate use, but actually these have a technical role in the algorithm. We start the computation at stack $S(0)$, process the stacks in increasing order of indices, and stop at stack $S(q)$ or $S(q+1)$. After completing the computation, every sequence R is stored in these two stacks. If the coordinate $R[q]$ is 1 then R is in $S(q)$, otherwise it is in $S(q+1)$.

At the beginning all stacks are empty, but during the running of the algorithm we put in and pop out sequences from them. The command 'push R into $S(k)$' puts the sequence R into the stack $S(k)$, and the command 'pop $S(k)$' removes a sequence out from $S(k)$ and returns it. The number of sequences in $S(k)$ is denoted by size.$S(k)$.

Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be sets of reactions part of the dense realization so that $\mathcal{R}_1 \cap \mathcal{R}_2 = \emptyset$. The property that all reactions of $\mathcal{R}_1$ take place in the reaction network but none of the elements of $\mathcal{R}_2$ is present can be represented by linear constraints. It can be added to the computation model as constraints that the entries of the matrix A_k corresponding to $\mathcal{R}_1$ and $\mathcal{R}_2$ should be strictly positive and zero, respectively. The non-strict inequalities cannot be added directly the LP model, however, the proof of Proposition 2.6 can be applied without modifications, since it was not supposed there that all the halfplanes should be closed. Therefore, a sequence $R \in S(k)$ defines a superstructure among the sequences in $C_k(R)$. More formally,

$$[R \in S(k),\ W \in C_k(R)\ ] \Longrightarrow [\ \forall j \in \{1, \ldots, q\}\ W[j] \leq R[j]\ ]. \tag{15}$$

In the algorithm we also use a binary array of size 2^q called *Exist*, where the returned sequences get stored. The indices of the fields are the sequences as binary numbers. At the beginning of the algorithm the value of each field is zero, and after the computation the value of the field $Exist[R]$ is one if and only if there is a linearly conjugate realization represented by the sequence R. The application of this array is not essential for the running of the algorithm, since according to Proposition 3.2 no realization is returned twice, and the sequences are collected in $S(q)$ and $S(q+1)$ at the end of the run. However, we can store the results for convenience in such an array.

Within the algorithm we repeatedly apply the following two subroutines:

FindLinConjWithZeros(R, k, i) computes the dense linearly conjugate realization where the reactions corresponding to the set J of indices is forbidden (i.e., the corresponding elements of matrix A_b are constrained to be zero in the computations). The inputs of the procedure define the set J as the union of two sets: $J = J_1 \cup J_2$, $J_1 = \{j \mid R[j] = 0,\ j \leq k\}$, $J_2 = \{j \mid k+1 \leq j \leq i\}$.
If there is no such realization of the kinetic system, then the procedure returns -1, otherwise it generates the sequence W describing the computed realization. Then the sequences W and R get compared to each other, and the procedure returns W only if $W =_k R$ holds, otherwise it returns -1. (The comparison step is necessary since there might be more zero coordinates corresponding to the indices 1 to k of W than required.)

FindNextOne(R, k) returns the smallest index i for which $i > k$ and $R[i] = 1$ hold. If there is no such index, i.e. $R[j]$ is zero for all $j > k$, then it returns $q + 1$.

Based on the above description, we can now give the pseudocode of the algorithm.

Algorithm 1 Compute all linearly conjugate graph structures
Inputs: Y, M, D, q
Output: *Exist*

```
push $D$ into $S(0)$
$Exist[D] := 1$
for $k = 0$ to $q - 1$ do
    while size.$S(k) > 0$ do
        $R :=$ pop $S(k)$
        $i :=$ FindNextOne$(R, k)$
        push $R$ into $S(i)$
        while $i \leq q$ do
            $W :=$ FindLinConjWithZeros$(R, k, i)$
            if $W < 0$ then
                BREAK
            else
                $i :=$ FindNextOne$(W, i)$
                push $W$ into $S(i)$
                $Exist[W] := 1$
            end if
        end while
    end while
end for
```

In the following paragraphs we give a brief explanation on the operation of the method. At the beginning of the computation we put the only known realization D in stack $S(0)$

and also save it in array $Exist$. Then by applying a for loop we pass through the stacks in increasing order of indices. For each stack we pop out sequences as long as it is not empty. In the general step sequence R is popped out from the actual stack $S(k)$.

If the index $i = \text{FindNextOne}(R, k)$ is smaller than $q+1$, then by the definitions it follows that R defines a superstructure not only in $C_k(R)$ but in $C_i(R)$ as well. Therefore we put R into stack $S(i)$ in order to save it for further examination. If there is no index i greater than k for which $R[i] = 1$ holds, then i computed by $\text{FindNextOne}(R, k)$ equals $q+1$. In this case R defines a superstructure in $C_q(R)$, but for the sake of simplicity we put it into $S[i] = S[q+1]$ and finish the examination of sequence R. This step is the reason why all the sequences with last coordinate equal to zero are all stored in stack $S(q+1)$ after the computation.

If i is smaller than $q+1$, by applying the loop in line 8 we examine the possible zero gaps starting from index $k+1$ in the sequences of $C_k(R)$. Since the values corresponding to the indices between k and i are all known to be zero, at the first attempt the largest index of the gap should be i.

If there is no suitable realization, i.e. the procedure $\text{FindLinConjWithZeros}(R, k, i)$ returns -1, then there cannot be any realization with $i-k$ or more consecutive zeros following the coordinate k in $C_k(R)$. In this case we stop the examination of R and take another sequence (if there is any).

If a real sequence W is returned by the procedure, the index $i = \text{FindNextOne}(W, k)$ is computed (which is equal to $\text{FindNextOne}(W, i)$ in the case of the actual value of the index i). Since W is a dense realization in $C_i(W)$, we put it in $S(i)$ and save it in array $Exist$ as well. Then, if i is smaller than $q+1$, we start looking for a realization having at least one more consecutive zeros following the coordinate k than W has.

The computation stops when there are no more sequences in stacks with indices smaller than q.

Using the previous description of the algorithm, we can now give formal results about its main properties.

Proposition 3.1. *For any kinetic system all the possible reaction graph structures representing linearly conjugate realizations can be computed by* **Algorithm 1** *after finitely many steps.*

Proof. Let us assume by contradiction that there is a linearly conjugate realization represented by the sequence V which is not returned by the algorithm. Let us take another sequence R, which was returned by the algorithm, it was in stack $S(p)$ at some point during the computation, $V =_p R$ holds and p is the greatest such number. There must be such a sequence, since the dense realization D and $p = 0$ meet the conditions. If $p = q$ or $q + 1$, then V is equal to R, consequently it is returned by the algorithm, so we can assume that $p \leq q - 1$ holds.

There is a point during the computation when sequence R is popped out from stack $S(p)$. By applying the notations of the algorithm let us assume, that $\mathrm{FindNextOne}(R, p)$ is i and $\mathrm{FindNextOne}(V, p)$ is j. It follows from the super-structure property of R that $i \leq j$ holds.

If j is equal to i then $V =_i R$ holds. But at some point of the computation sequence R is in stack $S[i]$, therefore p is not maximal, which is a contradiction.

If $i < j$ holds then the procedure $\mathrm{FindLinConjWithZeros}(R, p, i)$ is applied first at the examination of sequence R. Since the realization V fulfils the constraints, the procedure must return a sequence W_1. W_1 represents the dense realization in $C_i(W_1)$, but $V \in C_i(W_1)$ and $V[j] = 1$ hold, therefore $W_1[j]$ is also 1 and $\mathrm{FindNextOne}(W_1, p) = j_1 \leq j$ must be true.

If $j_1 = j$ holds, then $W_1 =_j V$ and W_1 is in stack $S(j)$ at some point of the computation. This means that p is not maximal, which is a contradiction.

If $j_1 < j$ holds, then we apply the procedure $\mathrm{FindLinConjWithZeros}(R, p, j_1)$. Since the realization represented by sequence V fulfils the constraints, it returns a sequence W_2 for which $\mathrm{FindNextOne}(W_2, p) = j_2 \leq j$ holds. If j_2 is equal to j, then we have a contradiction as before, otherwise we can continue the computation similarly. As a result we either get a contradiction for p not being maximal or get an infinite increasing sequence of integers which has an upper bound, and it is again a contradiction. This means that there cannot be any sequence V which represents a linearly conjugate realization of the kinetic system but is not returned by the algorithm.

The total computation time can be well characterized by the number of the calls of the subroutine FindLinConjWithZeros(). We do such an optimization step concerning a realization only when it is in a stack, at most as many times as the number of not fixed coordinates. In stack $S(k)$ there might be at most 2^k different sequences, therefore a very rough upper bound on the number of optimization steps is $\sum\limits_{k=0}^{q} (q - k)2^k$. ■

Proposition 3.2. *Within the computation described by* **Algorithm 1** *no realization is returned twice by the procedure* FindLinConjWithZeros.

Proof. Assume by contradiction that there is a sequence W which is computed twice during the algorithm, i.e. there are sequences R_1 and R_2, and integers k_1, k_2, i_1 and i_2 so that the objects in similar positions are not all identical, and the following holds:

$$W = \text{FindLinConjWithZeros}(R_1, k_1, i_1) = \text{FindLinConjWithZeros}(R_2, k_2, i_2) \qquad (16)$$

It can be assumed that $k_1 \leq k_2$ holds, and according to this value we distinguish two cases.

First let us assume that k_1 and k_2 are equal. It comes from the working of the algorithm that $R_1 =_{k_1} W$ and since R_1 is in stack $S(k_1)$ at some point of the computation, it defines a super-structure in $C_{k_1}(W)$. Similarly follows that R_2 defines a superstructure in $C_{k_2}(W) = C_{k_1}(W)$, and on account of the uniqueness of the super-structure the sequences R_1 and R_2 must be identical. There must be some difference among the inputs, therefore we can assume that $i_1 < i_2$ holds. Both computation steps are done when sequence R_1 is popped out from stack $S(k_1)$, but the smaller index is applied first. We get $W = \text{FindLinConjWithZeros}(R_1, k_1, i_1)$ and $j = \text{FindNextOne}(W, k_1)$. If j is q or $q + 1$ then we finish the examination of sequence R_1 from the coordinate k_1, and the procedure $\text{FindLinConjWithZeros}(R_1, k_1, i_2)$ is not applied. For a smaller j the procedure $\text{FindLinConjWithZeros}(R_1, k_1, j)$ returns either -1 or a proper sequence V. In the first case the examination is finished, otherwise the sequence V cannot be equal to W, since $W[j] = 1$ holds but $V[j]$ is zero. This property is fulfilled in case of every larger value of the index j, therefore $i_1 < i_2 < j$ must hold, however in this case during the computation the procedure $\text{FindLinConjWithZeros}(R_1, k_1, i_2)$ is not applied. Consequently, if $k_1 = k_2$ holds, then R_1 is identical to R_2 and assuming $i_1 < i_2$ in all cases the procedure $\text{FindLinConjWithZeros}(R_1, k_1, i_2)$ is not called.

Now we assume that k_1 is smaller than k_2. From the definitions we get that $R_1 =_{k_1} W$ holds, at some point of the computation R_1 is in $S(k_1)$ and therefore $R_1[k_1]$ is 1. We get similarly that $R_2 =_{k_2} W$ and $R_2[k_2] = 1$ hold. Consequently, R_1 and R_2 are in the equivalence class $C_{k_1}(R_1) = C_{k_1}(W)$, where R_1 represents the dense realization. From this it follows that $R_1[k_1] = R_2[k_1] = R_1[k_2] = R_2[k_2] = 1$ holds, and then $W[k_1] = W[k_2] = 1$ is also true.

As we assumed procedure $\text{FindLinConjWithZeros}(R_1, k_1, i_1)$ returns W. But since there are consecutive zero coordinates from index $k_1 + 1$ to i_1 in W and $W[k_2] = 1$, for i_1 the inequality $k_1 < i_1 < k_2$ must hold and $j = \text{FindNextOne}(W, k_1)$ can be at most k_2. Since $W =_{k_2} R_2$ holds, then $W =_j R_2$ is also true. According to the algorithm W is in $S(j)$ at some point of the computation, therefore the reaction graph G_{R_2} must be a subgraph of graph G_W. It leads to contradiction, since as results the procedure returns a realization with less coordinates equal to 1, therefore from the result FindLinConjWithZeros$(R_2, k_2, i_2) = W$ should follow that G_W is a subgraph of G_{R_2}. ■

Remark 3.3. *It is possible that a realization of the given kinetic system is computed multiple times by the procedure* FindLinConjWithZeros()*. However it is returned only once, when it is in the required equivalence class. In all other cases the procedure returns -1. Basically, this is the property stated and proved as Proposition 3.2.*

4 Computation results and efficiency analysis

In this section we illustrate the operation of our algorithm on two examples taken from the literature. The obtained results are compared to the features of the single thread implementation of the solution published in [40]. All the computations were performed on a Lenovo D60 workstation with two 2.60GHz Xeon (E5-2650 v2) processors and 32 Gb RAM (DDR3 1600 MHz, 0.6ns). The algorithms were implemented in MATLAB [26] using the YALMIP modelling language [25]. It was checked and confirmed that the two compared algorithms computed exactly the same reaction graph structures both for Example 1 and Example 2.

4.1 Example 1

The kinetic system (17) examined in this section was originally published in [19], where it was represented by Equations (7-3) and (7-4) characterizing a dynamically equivalent realization (Y, A_k) of it.

$$\begin{aligned} \dot{x}_1 &= x_1^2 x_2 - 2\varepsilon x_1^3 - x_1 x_2^2 + 2\varepsilon x_2^3 \\ \dot{x}_2 &= -x_1^2 x_2 + 2\varepsilon x_1^3 + x_1 x_2^2 - 2\varepsilon x_2^3 \end{aligned} \tag{17}$$

The complexes in set $\mathcal{C}$ are built from two species X_1 and X_2 as follows: $C_1 = 2X_1 + X_2, C_2 = 3X_1, C_3 = X_1 + 2X_2, C_4 = 3X_2$. The structures of the complexes and the reaction rates are stored in the stoichiometric matrix Y and the Kirchhoff matrix A_k of the CRN as follows.

$$Y = \begin{bmatrix} 2 & 3 & 1 & 0 \\ 1 & 0 & 2 & 3 \end{bmatrix} \qquad A_k = \begin{bmatrix} -1 & 0 & 0 & \varepsilon \\ 1 & -\varepsilon & 0 & 0 \\ 0 & \varepsilon & -1 & 0 \\ 0 & 0 & 1 & -\varepsilon \end{bmatrix}$$

The reaction graph complemented with the edge weights can be seen in Figure 1.

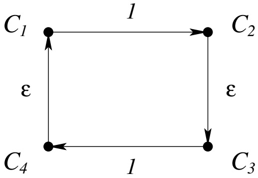

Figure 1. Reaction graph representing the CRN.

The coefficient matrix $M = Y \cdot A_k$ is given by

$$M = \begin{bmatrix} 1 & -2\varepsilon & -1 & 2\varepsilon \\ -1 & 2\varepsilon & 1 & -2\varepsilon \end{bmatrix}.$$

In the case of the parameter value $\varepsilon = 1/7$ our algorithms found 784 different reaction graph structures representing linearly conjugate realizations of this simple kinetic system. The distribution of possible different graph structures with given numbers of reactions is depicted in Fig. 2. As it is visible, the number of sparse structures, that have 4 directed edges is 9 in this case.

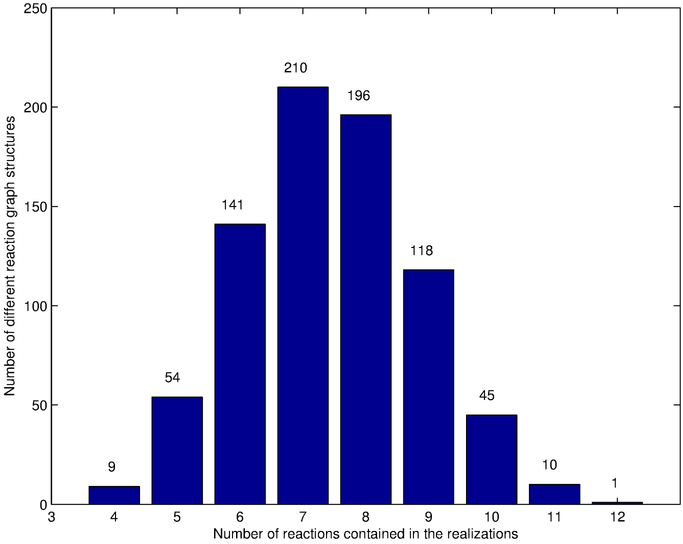

Figure 2. Number of different reaction graph structures with given numbers of directed edges in the case of Example 1

We obtained that the dense realization $(A_b^d, (T^d)^{-1})$ contains all the possible reactions, i.e. it can be represented by the complete directed graph. The matrices characterizing the dense realization are the following:

$$A_b^d = \begin{bmatrix} -50002500 & 714.29 & 12498750 & 0.00036 \\ 29170416.67 & -952.38 & 8333333.33 & 238.0955 \\ 12498750 & 0.00036 & -50002500.0025 & 714.29 \\ 8333333.33 & 238.095 & 29170416.67 & -952.39 \end{bmatrix} \qquad (T^d)^{-1} = \begin{bmatrix} 5000.005 & 0 \\ 0 & 5000.005 \end{bmatrix}$$

It is interesting to mention that there is only one realization where the reaction graph is not connected. In this case there are two linkage classes (i.e. connected components) and the realization is a sparse one at the same time. It is described by the matrices A_b^s and $(T^s)^{-1}$:

$$A_b^s = \begin{bmatrix} -5000.005 & 1428.573 & 0 & 0 \\ 5000.005 & -1428.573 & 0 & 0 \\ 0 & 0 & -5000.005 & 1428.573 \\ 0 & 0 & 5000.005 & -1428.573 \end{bmatrix} \qquad (T^s)^{-1} = \begin{bmatrix} 5000.005 & 0 \\ 0 & 5000.005 \end{bmatrix}$$

Table 1 shows the comparison results for the applied two algorithms. The explanation of the compared features in Tables 1 and 2 is the following:

1. The total running time of the algorithm from start to end in seconds.
2. The computation time spent for solving optimization (i.e. linear programming) problems in seconds, including the setup of constraints.
3. The number of different reaction graph structures corresponding to linearly conjugate realizations of the studied kinetic system found by the applied methods.
4. The total number of function calls for computing constrained dense realizations (FindLinConj-WithZeros() in the proposed algorithm, and FindLinConjWithoutEdge() in [40]).
5. The number of infeasible function calls for computing constrained dense realizations (see, item 4, too).
6. The number of valid reaction graph structures found that had already been computed previously.
7. The average time in seconds that is elapsed between displaying/storing two distinct consecutive reaction graph structures.

8. The maximal time interval in seconds that is elapsed between displaying/storing two distinct consecutive reaction graph structures.
9. The variance of time intervals elapsed between displaying/storing two distinct consecutive reaction graph structures.

Table 1. Comparison of the properties of the proposed algorithm with the method published in [40] for Example 1

feat. no.	**feature description**	**current solution**	**solution in [40]**
1.	total running time [s]	617.5386	2933.4868
2.	total optimization time [s]	616.1069	2923.1713
3.	no. of distinct valid structures found	784	784
4.	no. of constrained dense real. comps.	1096	5825
5.	no. of infeasible comp. steps	312	2240
6.	no. of structures found again	0	2801
7.	avg. computation interval [s]	0.788	3.7416
8.	max. computation interval [s]	14.97	12.0999
9.	variance of computation intervals [s^2]	0.30339	2.3894

4.2 Example 2

In this subsection we study a slightly bigger example, which was first introduced in [10] as example $A1$. The computation problem was investigated in [40], where the details on the features of the computed reaction graph structures can be found. Here we are primarily interested in the performance comparison of the approaches in the present paper and in [40].

In the original article [10] the kinetic system is given by the following realization of it, described by the matrices

$$Y = \begin{bmatrix} 0 & 1 & 0 & 2 & 2 & 3 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \qquad A_k = \begin{bmatrix} -k_1 & k_2 & 0 & 0 & 0 & 0 \\ 0 & -k_2 & k_3 & 0 & 0 & 0 \\ k_1 & 0 & -k_3 & k_4 & 0 & 0 \\ 0 & 0 & 0 & -k_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & -k_5 & 0 \\ 0 & 0 & 0 & 0 & k_5 & 0 \end{bmatrix}.$$

In the reaction network there are two species X_1 and X_2 and six complexes: $\mathcal{C} = \{C_1 = 0, C_2 = X_1, C_3 = X_2, C_4 = 2X_1, C_5 = 2X_1 + X_2, C_6 = 3X_1\}$. According to the definitions, the coefficient matrix M is given as

$$M = \begin{bmatrix} 0 & -k_2 & k_3 & -2k_4 & k_5 & 0 \\ k_1 & 0 & -k_3 & k_4 & -k_5 & 0 \end{bmatrix}.$$

The reaction rate coefficients applied during the computations were the same as in [10], namely: $k_1 = 1,\ k_2 = 1,\ k_3 = 0.05,\ k_4 = 0.1,\ k_5 = 0.1$. With these parameter values the dynamics of the system shows oscillatory behaviour. It is known from [40] that the dense realization of this kinetic system contains 19 reactions.

Table 2 shows the comparison results with [40] for this example. It can be seen that the advantage of our current solution over [40] increased slightly in the case of the studied larger example considering overall computation time. This fact mainly comes from Proposition 3.2 that results in a significantly lower number of constrained dense realization computation steps than [40]. However, as features 8 and 9 show, the solution in [40] may guarantee a 'more even' run with smaller variance in the computation intervals. The reason for this is the important property of the algorithm in [40] that polynomial time is elapsed between displaying any two consecutive realizations.

Table 2. Comparison of the properties of the proposed algorithm with the method published in [40] for Example 2

feat. no.	**feature description**	**current solution**	**solution in** [40]
1.	total running time [s]	23359.4359	139456.3050
2.	total optimization time [s]	23126.6604	135217.6015
3.	no. of distinct valid structures found	17160	17160
4.	no. of constrained dense real. comps.	39662	211265
5.	no. of infeasible comp. steps	22502	79304
6.	no. of structures found again	3820	114801
7.	avg. computation interval [s]	1.1698	8.1268
8.	max. computation interval [s]	813.7367	22.2412
9.	variance of computation intervals [s^2]	38.6325	7.8097

5 Conclusion

A novel method was proposed in this paper for computing all distinct reaction graph structures corresponding to linearly conjugate realizations of polynomial kinetic systems assuming a fixed set of complexes. The inputs of the method are the kinetic differential equations and the complexes, while the output is the set of computed reaction graphs. The computation is started from the dense realization, and the subsequent reaction graphs are computed using linear programming and are stored in a multi-level stack structure. The correctness of the algorithm was formally proved. The proposed approach was compared to the single-thread implementation of a previous solution published in [40] via two illustrative examples, and it was shown that the number of optimization steps, and therefore the overall computation time and the average computation intervals are significantly smaller in the case of the new algorithm. Future work will be focused on the parallel implementation of the method, as well as on the further structural analysis of the obtained large number of reaction graphs corresponding to the studied kinetic models.

Acknowledgments: This paper is dedicated to Prof. Árpád I. Csurgay on the occasion of his 80th birthday. This project was developed within the PhD program of the Roska Tamás Doctoral School of Sciences and Technology, Faculty of Information Technology and Bionics, Pázmány Péter Catholic University, Budapest. The authors acknowledge the support of the National Research, Development and Innovation Office NKFIH through grant no. NF104706. The first author acknowledges the support of the project PPKE KAP-1.1-15/052. The authors thank the anonymous reviewers for the constructive comments.

References

[1] D. F. Anderson, A proof of the global attractor conjecture in the single linkage class case, *SIAM J. Appl. Math.* **71** (2011) 1487–1508.

[2] J. Bang–Jensen, G. Gutin, *Digraphs: Theory, Algorithms and Applications*, Springer, Berlin, 2001.

[3] J. R. Banga, Optimization in computational systems biology, *BMC Syst. Biol.* **2** (2008) 47–54.

[4] B. Boros, On the existence of the positive steady states of weakly reversible deficiency–one mass action systems, *Math. Biosci.* **245** (2013) 157–170.

[5] V. Chellaboina, S. P. Bhat, W. M. Haddad, D. S. Bernstein, Modelling and analysis of mass-action kinetics – nonnegativity, realizability, reducibility and semistability, *IEEE Control Syst. Mag.* **29** (2009) 60–78.

[6] B. L. Clarke, *Stability of Complex Reaction Networks*, Wiley, Hoboken, 1980.

[7] G. Craciun, Toric differential inclusions and a proof of the global attractor conjecture, *arXiv:1501.02860* [math.DS] (2015).

[8] G. Craciun, M. Feinberg, Multiple equilibria in complex chemical reaction networks: II. The species–reaction graph, *SIAM J. Appl. Math.* **66** (2006) 1321–1338.

[9] G. Craciun, C. Pantea, Identifiability of chemical reaction networks, *J. Math. Chem.* **44** (2008) 244–259.

[10] A. Császár, L. Jicsinszky, T. Turányi, Generation of model reactions leading to limit cycle behaviour, *React. Kinet. Catal. L.* **18** (1981) 65–71.

[11] R. Díaz–Sierra, B. Hernández–Bermejo, V. Fairén, Graph–theoretic description of the interplay between non–linearity and connectivity in biological systems, *Math. Biosci.* **156** (1999) 229–253.

[12] G. Farkas, Kinetic lumping schemes, *Chem. Eng. Sci.* **54** (1999) 3909–3915.

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

[14] M. Feinberg, Chemical reaction network structure and the stability of complex isothermal reactors – I. The deficiency zero and deficiency one theorems, *Chem. Eng. Sci.* **42** (1987) 2229–2268.

[15] J. L. Gross, J. Yellen, *Graph Theory and its Applications*, CRC Press, 2006.

[16] I. Gutman, O. E. Polansky, *Mathematical Concepts in Organic Chemistry*, Springer, 1986.

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

[18] F. Horn, Necessary and sufficient conditions for complex balancing in chemical kinetics, *Arch. Ration. Mech. An.* **49** (1972) 172–186.

[19] F. Horn, R. Jackson, General mass action kinetics, *Arch. Ration. Mech. An.* **47** (1972) 81–116.

[20] V. Hárs, J. Tóth, On the inverse problem of reaction kinetics, *Coll. Math. Soc. J. B.* **30** (1981) 363–379.

[21] M. D. Johnston, D. Siegel, Linear conjugacy of chemical reaction networks, *J. Math. Chem.* **49** (2011) 1263–1282.

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

[23] X. Li, Y. Shi, A survey on the Randić index. *MATCH Commun. Math. Comput. Chem.* **59** (2008) 127–156.

[24] G. Lipták, G. Szederkényi, K. M. Hangos, Computing zero deficiency realizations of kinetic systems. *Syst. Control Lett.* **81** (2015) 24–30.

[25] J. Löfberg, YALMIP: A toolbox for modeling and optimization in MATLAB, *Proc. CACSD Conference*, Taipei, Taiwan, 2004.

[26] The Math Works, Inc., Natick, MA. *Matlab User's Guide*, 2000.

[27] M. Mincheva, G. Cracium, Multigraph conditions for multistability, oscillations and pattern formation in biochemical reaction networks, *Proc. IEEE* **96** (2008) 1281–1291.

[28] K. Murota, *Systems Analysis by Graphs and Matroids*, Springer, 1987.

[29] G. Osipenko, *Dynamical Systems, Graphs, and Algorithms*, Springer, 2007.

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

[31] M. Saez, C. Wiuf, E. Feliu, Graphical reduction of reaction networks by linear elimination of species, *J. Math. Biol.*, in press.

[32] N. Samardzija, L. D. Greller, E. Wassermann, Nonlinear chemical kinetic schemes derived from mechanical and electrical dynamical systems, *J. Chem. Phys.* **90** (1989) 2296–2304.

[33] L. Shi, Chemical indices, mean distance, and radius, *MATCH Commun. Math. Comput. Chem.* **75** (2016) 57–70.

[34] G. Szederkényi, Computing sparse and dense realizations of reaction kinetic systems, *J. Math. Chem.* **47** (2010) 551–568.

[35] G. Szederkényi, J. R. Banga, A. A. Alonso, Inference of complex biological networks: distinguishability issues and optimization–based solutions, *BMC Syst. Biol.* **5** (2011) #177 (15 pp.).

[36] G. Szederkényi, K. M. Hangos, Finding complex balanced and detailed balanced realizations of chemical reaction networks, *J. Math. Chem.* **49** (2011) 1163–1179.

[37] Z. A. Tuza, G. Szederkényi, Computing core reactions of uncertain polynomial kinetic systems, *23rd Mediterranean Conference on Control and Automation (MED), June 16-19, 2015. Torremolinos, Spain*, 2015, pp. 1187–1194.

[38] A. I. Vol'pert, Differential equations on graphs, *Math. USSR Sb+* **17** (1972) 571–582.

[39] B. Ács, G. Szederkényi, Z. A. Tuza, Zs. Tuza, Computing linearly conjugate weakly reversible kinetic structures using optimization and graph theory, *MATCH Commun. Math. Comput. Chem.* **74** (2015) 481–504.

[40] B. Ács, G. Szederkényi, Zs. Tuza, Z. A. Tuza, Computing all possible graph structures describing linearly conjugate realizations of kinetic systems, *Comput. Phys. Commun.* **204** (2016) 11–20.

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