RUTCOR
RESEARCH
REPORT

Effective solution of linear Diophantine equation systems with an application in chemistry

Dávid Papp[a] Béla Vizvári[b]

RRR 28-2004, September, 2004

RUTCOR
Rutgers Center for
Operations Research
Rutgers University
640 Bartholomew Road
Piscataway, New Jersey
08854-8003
Telephone: 732-445-3804
Telefax: 732-445-5472
Email: rrr@rutcor.rutgers.edu
http://rutcor.rutgers.edu/~rrr

[a]Dept. of Mathematical Analysis, Budapest University of Technology and Economics, pappd@cs.bme.hu

[b]Dept. of OR, Eötvös Loránd University of Budapest (ELTE), H-1117 Budapest, Pázmány Péter s. 1/C, visitor of RUTCOR, bvizvari@rutcor.rutgers.edu

Rutcor Research Report
RRR 28-2004, September, 2004

Effective solution of linear Diophantine equation systems with an application in chemistry

Dávid Papp   Béla Vizvári

Abstract. Systematic methods of the solution of linear Diophantine systems of equations and their application in decomposing complex chemical reactions are presented. The earlier presented Contejean–Devie algorithm is improved. A new, linear programming based enumerative algorithm is described, which is applicable to large systems with large solutions. *Mathematica* implementations are tested and compared in important chemical examples.

Acknowledgements: The research was partially carried out when the second author visited RUTCOR and DIMACS. DP was partially supported by the Hungarian National Scientific Research Foundation, No. T037491. The authors are grateful to János Tóth for his helpful comments on the previous versions of the paper. DP thanks Daniel Lichtblau and Yifan Hu for their help in the *Mathematica* implementation.

1 Introduction

In chemistry reactions or *steps* are called *elementary* if they start from at most two reactant species. Very often only *overall* reactions are known, where the number of reactants is greater than 2. It is a general view that only elementary reactions take place in nature and the overall reactions are aggregated results of some elementary reactions. The chemical equation of an overall reaction gives a higher-level description only, since it does not contain any information on the elementary reactions of the chemical mechanism. The aim of decomposing overall reactions is to systematically calculate the sets of elementary steps that may add up to the given reaction. One of the most successfully investigated class of reactions is the combustion processes of hydrocarbons [13], with the well-known application in petrol chemistry.

In chemistry the different kind of materials involved in a chemical reaction including ions and electrons are referred to as *species*.

First the set of species that might be produced in the mechanism must be selected – not only the products but every possible intermediate species as well. (A species is called *intermediate* if it is not included in the overall reaction.) Then every elementary step that might take place among these species must be determined. The latter task is usually considered as a purely chemical problem. In fact, as it is discussed in section 4, this part of the problem can also be automated, moreover, with a well-established algorithmic approach significantly higher number of elementary reactions can be produced than it would be possible by chemical "instinct" only.

The third step is the decomposition of the overall reaction into the elementary steps in as many ways as possible. The total number of decompositions can be so high that not all of them can be generated. The last step is a filtering process to rule out the chemically infeasible decompositions. This last step is based on a graph-indexing technique. But the previous steps require the solution of linear systems of Diophantine equations over nonnegative integers.

The algorithmic problems encountered during the decompositioning process are usually **NP**-hard if not of exponential space complexity. Linear Diophantine equations are discussed by many authors of the field (see e.g. [3, 8, 11, 15]), but the algorithms described in these works are mostly designed to decide the solvability of a system (over integers), without generating its solutions. In the present paper not only an optimal solution is required, but all nonnegative solutions or all minimal nonnegative solutions are of interest from the chemical point of view.

Previous decompositions of chemical processes are mostly based on *ad hoc* solutions: highly relying on special properties of the overall reaction in question, and the used thermodynamical data usually are not available for inorganic reactions. Our goal was to design and implement mathematical algorithms that support the automated decomposition of reactions even if no thermodynamical data is available. Certainly, these data, if available, can be taken into account during the analysis of a complex reaction right after the decomposition process.

The structure of the paper is this: the problem of decomposing overall reactions if formulated in section 2. Our investigation of linear Diophantine systems is presented in section 3. Decomposition generation and filtering is discussed in section 5. Sections 4.1 and 5.2 show an example for the decomposition of a complex inorganic reaction of high importance, namely the permanganate/oxalic acid reaction.

2 Overall and elementary reactions

The solution starts with the selection of the species participating in the mechanism. This information must come from chemical experiments and measurements. Here only at most a few dozens of species are taken into consideration instead of listing hundreds (or thousands) of elementary reactions that are essential for a valid chemical model.

Elementary reactions are chemical reactions with at most two reactant species. Before the decomposition of the given overall reaction takes place, all possible elementary steps must be determined that might take place during the reaction. The usual approach to this problem is that it essentially needs the knowledge of chemical experts, thus it is not a mathematical problem. This approach leads to the fact that sometimes the number of elementary steps hardly exceeds the number of the species (see e.g. [12]).

One restriction what is used in general is the law of atomic and charge balance. [5, Chapter 3] Suppose that species are made of k different atoms. Assign a $k+1$ dimensional vector to each species where the first k components are the quantities of the different atomic constituents and the electric charge is the last component. This is the only component of the vectors, which might be negative. This atomic structure of the species is usually described the *atomic matrix*, an example of which is Table 1. Then the weighted sum of the vectors assigned to the species on the two sides of a reaction are equal. More formally, assume that in a chemical reaction $R_1, \dots, R_n$ are the reactants and $P_1, \dots, P_m$ are the products. Then the chemical reaction is usually described in the form:

$$\alpha_1 R_1 + \cdots + \alpha_n R_n \to \beta_1 P_1 + \cdots + \beta_m P_m,$$

where the constants $\alpha_1, \dots, \alpha_n, \beta_1, \dots, \beta_m$, the so called *stoichiometric coefficients*, are positive integers. Suppose that the reactants and the products are represented by the $k+1$ dimensional integer vectors $\mathbf{r}_1, \dots, \mathbf{r}_n$ and $\mathbf{p}_1, \dots, \mathbf{p}_m$, where the components are denoted by r_{ij}, and p_{ij}, respectively. Then the equation of the reaction in a linear algebraic form is

$$\sum_{j=1}^{n} \alpha_j \mathbf{r}_j = \sum_{i=1}^{m} \beta_i \mathbf{p}_i.$$

Using this notation, a reaction is elementary if $\sum_{i=1}^{n} \alpha_i \leq 2$.

The elementary reactions can be generated by expressing in every possible way the atomic vector of each species and the double of these vectors and the sum of the vectors of any two species as a linear combination of the atomic vectors of the other species with nonnegative integer coefficients. This is equivalent to the solution of several linear Diophantine equation system over nonnegative integers.

The generation of the decompositions of the overall reaction is similar to the previous step. Let s be the number of species. An s-dimensional vector is assigned to each reaction. The absolute value of the i-th $(1 \leq i \leq s)$ component of such a vector is the coefficient of the i-th species in the reaction. If a species is a reactant, then the component is negative, if it is a product then the component is positive. A similar vector is assigned to the overall reaction.

Then our task is to express in every possible way the vector of the overall reaction as the linear combination of the vectors assigned to the elementary steps with nonnegative integer coefficients.

Let t be the number of elementary reactions. Then more formally the generation of the decompositions is equivalent to the generation of the nonnegative integer solutions of a

$$\mathbf{\Gamma x} = \mathbf{w}$$

linear Diophantine equation system, where the $\mathbf{\Gamma}$ *stoichiometric matrix* is an $s \times t$ the matrix, whose columns are the stoichiometric coefficients of the elementary reactions and $\mathbf{w}$ is the vector of the overall reaction.

This equation differs significantly from those in the previous step determining the elementary reactions. It is considerably larger than the previous equations and typically has infinite number of solutions while the former equations do not.

3 Linear systems of Diophantine equations

The symbols $\mathbb{N}$, $\mathbb{Z}$ and $\mathbb{Q}_0^+$ denote the set of nonnegative integer, arbitrary integer and nonnegative rational numbers, respectively. Vectors and their co-ordinates are denoted by bold and italic letters, respectively. If $\mathbf{x} \in \mathbb{Z}^n$, $\|\mathbf{x}\|_1 := \sum_{i=1}^n |x_i|$ is its length, $\|\mathbf{x}\|_\infty := \max |x_i|$ is its height, and $\|\mathbf{x}\|$ is its Euclidean norm. The scalar product of $\mathbf{v}$ and $\mathbf{w}$ is denoted by $\langle \mathbf{v}, \mathbf{w} \rangle$. We use the relation $\mathbf{v} \geq \mathbf{w}$ if and only if $\forall i : v_i \geq w_i$, and with this notation $\mathbf{v} \gneqq \mathbf{w}$ if and only if $\mathbf{v} \geq \mathbf{w}$ but $\mathbf{v} \neq \mathbf{w}$. The null vector of $\mathbb{Z}^n$ will be denoted by $\mathbf{0}^n$ or simply $\mathbf{0}$, if no ambiguity arises. The elements of the canonical basis of $\mathbb{Z}^n$ are $\mathbf{e}^{(1)}, \ldots, \mathbf{e}^{(n)}$.

The problem to solve is the following. Let $\mathbf{b} \in \mathbb{Z}^d$ and $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n\}$ $(\mathbf{v}_i \in \mathbb{Z}^d)$ be fixed vectors. We have to find the n-tuples $(a_1, a_2, \ldots, a_n) \in \mathbb{N}^n$ such that

$$\sum_{i=1}^{n} a_i \mathbf{v}_i = \mathbf{b}. \tag{1}$$

holds. The set of solutions will be denoted by $\mathcal{M}$, the set of minimal solutions with respect to $\gneqq$ is $\mathcal{M}'$. If the system (1) is inhomogeneous, i.e. $\mathbf{b} \neq \mathbf{0}$, then the set of minimal solutions of the corresponding homogeneous system will be denoted by $\mathcal{M}_0$.

It is worth noting that the problem is $\mathbf{NP}$-hard, even if $d = 1$, because then it is equivalent to the $\mathbf{NP}$-hard subset sum problem [9, section 9.4]. Even the decision if such a system has at least one integer solution is $\mathbf{NP}$-complete.

Algorithms of generating the nonnegative solutions of (1) do not appear even in the state-of-the-art works on Diophantine equations, such as [11, 15]. The aim of the integer programming methods is to find an optimal solution, i.e. even these methods do not generate all feasible solutions. Until recently the widely used mathematical software packages like *Mathematica* or *Maple* also lacked of functions that can solve such equations. (In version 5, though, *Mathematica* has extended its function `Reduce` so it solves Diophantine equations over naturals as well as integers. This function, however, performs poorly with large systems.)

A naive algorithm, e.g. a greedy-type depth-first search or breadth-first search procedure, cannot be applied here. First of all, the set of solutions may be infinite because of the presence of both positive and negative coefficients. Secondly, such a procedure might not terminate even if the set of solutions is finite. Another problem is that the applicability of the procedure depends on the order of the vectors $\mathbf{v}_i$'s. For example, let $n = d = 2, \mathbf{v}_1 = (1,0)^{\mathrm{T}}, \mathbf{v}_2 = (-1,1)^{\mathrm{T}}$. For any vector $\mathbf{b}$ (1) has at most one solution. Using the vector $\mathbf{v}_1$ as many times as possible, the vector $\mathbf{v}_2$ can increase the infeasibility in one component and decrease in the other one. If the vectors are used in the opposite order this may not happen and the search ends up in an infinite cycle. The naive algorithm can be used only if the termination of the procedure is a priori ensured.

3.1 Algorithm of Contejean and Devie

In this subsection an improved version of the algorithm of Contejean and Devie, proposed in [3], is discussed. This algorithm is to find the minimal solutions of a homogeneous linear Diophantine equation. Equation (1) has finite number of solutions, if and only if every solution is minimal. Thus it is reasonable to search for $\mathcal{M}'$ only, instead of $\mathcal{M}$.

The algorithm of Contejean and Devie is a clever modification of the trivial infinite procedure of finding the minimal solutions by breadth-first search such that this new procedure is not only faster and more space-efficient, but it also terminates. The basic breadth-first procedure to solve the homogeneous problem is the following:

1. **[init]** $\mathcal{A} := \{\mathbf{0}^n\}$, $\mathcal{M}' := \emptyset$.
2. **[breadth-first search]** $\mathcal{A} := \{\mathbf{a} + \mathbf{e}^{(k)} \mid \mathbf{a} \in \mathcal{A}, 1 \le k \le n\}$
3. **[new minimal results]** $\mathcal{M}' := \mathcal{M}' \cup \{\mathbf{a} \in \mathcal{A} \mid \mathbf{a}$ is a solution of (1)$\}$
4. **[unnecessary branches]** $\mathcal{A} := \mathcal{A} \setminus \{\mathbf{a} \mid \exists \mathbf{m} \in \mathcal{M}' : \mathbf{m} \le \mathbf{a}\}$
5. **[test]** If $\mathcal{A} = \emptyset$, stop. Otherwise go to step 2.

This procedure may never terminate. The modification proposed in [3] is the following: in Step 2 the vector $\mathbf{a}$ is increased by $\mathbf{e}^{(k)}$ only if the scalar product $\langle \sum_{i=1}^n a_i \mathbf{v}_i,\, \mathbf{v}_k \rangle$ is negative. Thus, in the homogeneous Contejean–Devie algorithm Step 2 is the following:

2. **[modified BFS step]** $\mathcal{A} := \{\mathbf{a} + \mathbf{e}^{(k)} \mid \mathbf{a} \in \mathcal{A},\ 1 \le k \le n, \langle \sum_{i=1}^n a_i \mathbf{v}_i,\, \mathbf{v}_k \rangle < 0\}$

It is easy to adopt the algorithm to solve inhomogeneous systems. The vector $-\mathbf{b}$ should be added to the vectors on the left side, i.e. let $\mathbf{v}_{n+1} = -\mathbf{b}$, and the null vector in the first step should be replaced by the $(n+1)$-dimensional vector $(\underbrace{0,\ldots,0}_{n},1)$. Finally, the last component of the elements of $\mathcal{M}'$ must be omitted.

The main result in [3] is that the modified algorithm is not only correct, but it terminates. Another advantage of it is that it can be implemented in a very space-efficient way. The set $\mathcal{A}$, in which partial results are collected, can be represented as a stack of depth n (or $n+1$ in the case of inhomogeneous systems).

A further improvement will be discussed, where the number of steps can be reduced even more, provided that an upper bound on the length of the minimal solutions is known.

Theorem. To obtain the minimal solutions of length not greater than m only, the homogeneous Contejean-Devie algorithm can be modified as follows: in step 2 the vector $\mathbf{a}$ can be increased by $\mathbf{e}^{(k)}$ only if $m > \|\mathbf{a}\|_1$ and

$$\langle \sum_{i=1}^{n} a_i \mathbf{v}_i,\ \mathbf{v}_k \rangle \leq -\frac{\|\sum_{i=1}^{n} a_i \mathbf{v}_i\|^2}{m - \|\mathbf{a}\|_1} \tag{2}$$

Proof. The finiteness algorithm of the algorithm follows evidently from the same property of the original one. Only the completeness of the algorithm is to be proved.

Let $\mathbf{a}^* = (a_1^*, a_2^*, \ldots, a_n^*)$ a minimal solution, $\mathbf{a} \lneqq \mathbf{a}^*$, and $a_i^* = a_i + r_i$ $(i = 1, ..., n)$. Then

$$\begin{aligned} 0 = \|\sum_i a_i^* \mathbf{v}_i\|^2 &= \|\sum_i a_i \mathbf{v}_i\|^2 + \|\sum_i r_i \mathbf{v}_i\|^2 + 2\langle \sum_i a_i \mathbf{v}_i,\ \sum_i r_i \mathbf{v}_i \rangle \\ &= 2\|\sum_i a_i \mathbf{v}_i\|^2 + 2\langle \sum_i a_i \mathbf{v}_i,\ \sum_i r_i \mathbf{v}_i \rangle. \end{aligned}$$

Thus

$$\begin{aligned} -\|\sum_i a_i \mathbf{v}_i\|^2 &= \langle \sum_i a_i \mathbf{v}_i,\ \sum_i r_i \mathbf{v}_i \rangle \\ &= \sum_k r_k \langle \sum_i a_i \mathbf{v}_i,\ \mathbf{v}_k \rangle. \end{aligned}$$

The sum on the right-hand side has $\sum r_k < m$ terms, and each term has the form $\langle \sum_{i=1}^{n} a_i \mathbf{v}_i,\ \mathbf{v}_k \rangle$. This can only be equal with the expression on the left-hand side, if one of the terms is not greater than $(1/\sum r_k)$ times the left-hand side. This term satisfies condition (2). Thus we can proceed with the algorithm at any partial solution $\mathbf{a}$, if there is an $\mathbf{a}^*$ minimal solution satisfying $\mathbf{a} \lneqq \mathbf{a}^*$). ■

Remark. If $m \to \infty$, then the statement of the theorem is identical to the quoted condition formulated in [3]. If one has a good bound of m then a great number of steps can be saved by using the stronger condition. Unfortunately, the proof of the original theorem in [3] does not imply any upper bound for the length of the minimal solutions. In the case of the decomposition of overall reactions even chemical evidences can be used.

3.2 Upper bounds on the lengths of minimal solutions

In his paper [4] Domenjoud describes an algorithm, that finds the minimal solutions of (1) in a way that leads to an upper bound on the length of the minimal solutions. The algorithm uses an algebraic approach and requires the computation of the r-th minors of the matrix of the system (where r is the rank of the matrix), which is too much computational effort for larger systems.

Pottier's paper [10] is a complete summary of the best upper bounds known, including those derived from Domenjoud's results. Its main theorem is the following:

Theorem. Denote the matrix of the homogeneous system by $\mathbf{A}$. Let $r = \mathrm{rank}(\mathbf{A})$, D_r and D'_r be the largest absolute value of the minors, of order r of $\mathbf{A}$, and of order $r+1$ of $\binom{1\cdots 1}{\mathbf{A}}$, respectively. The columns of $\mathbf{A}$ are denoted by $\mathbf{v}_1, ..., \mathbf{v}_n$. Then for every minimal solution $\mathbf{m} \in \mathcal{M}'$ of the homogeneous equation (1) the following inequalities hold:

$$\begin{aligned}
\|\mathbf{m}\|_1 &\leq (1 + \max_i \|\mathbf{v}_i\|)^r, && (3)\\
\|\mathbf{m}\|_1 &\leq (n-r)D'_r, && (4)\\
\|\mathbf{m}\|_\infty &\leq (n-r)\left(\frac{\|\mathbf{A}\|_1}{r}\right)^r, && (5)\\
\|\mathbf{m}\|_\infty &\leq (n-r)D_r. && (6)
\end{aligned}$$

■

Obviously (6) is sharper than (5) (in fact, (4) and (5) both are consequences of (6)), but the bounds on the right-hand side of (4) and (6) are not computable in practice. For larger systems only the first and the third bounds are applicable. All these bounds are sharp in some sense. (See [10] for details.) In practical chemical examples bounds coming from chemical conditions are much more restrictive than these bounds as only decompositions having less than, say, a few hundred steps are of interest.

It is interesting to note, that while the last three bounds come from an algorithm that finds the solutions, the proof of the first bound does not involve algorithmic techniques and is not closely related to any algorithm that solves (1). However, these bounds can accelerate the Contejean–Devie and the LP-based enumerative algorithms, which do not imply directly any upper bound on the solutions.

3.3 LP relaxation

The algorithm presented in this subsection is based on the LP relaxation of the problem. The algorithm works as follows: the general (parameterized) solution of the equation (1) is determined over real numbers. Then taking into account the nonnegativity constraints the integer solutions of the new inequality system are determined.

It is easy to see that the following lemma is true.

Lemma. Let $r = \operatorname{rank}(\mathbf{A})$. The vector space of the solutions of a homogeneous system of equations has a basis $\{\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_r\}$ such that the matrix $(\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_r)$ has a $r \times r$ diagonal submatrix containing positive integers in the diagonal.

If one particular solution of the inhomogeneous equation is $\mathbf{p} = (p_1, p_2, \ldots, p_n), (p_i \in \mathbb{Q})$, then the general solution of the inhomogeneous equation can be written in the form

$$\mathbf{a} = \mathbf{p} + \mathbf{b}_1 x_1 + \cdots + \mathbf{b}_r x_r \quad (x_i \in \mathbb{R}).$$

Those vectors $\mathbf{x} = (x_1, x_2, \ldots, x_k)$ are to be determined, for which each component of $\mathbf{a}$ is a nonnegative integer. Assume that $\beta_i \in \mathbb{Z}^+$ is the positive component of $\mathbf{b}_i$ in the diagonal mentioned in the lemma. Then the i-th component of $\mathbf{a}$ is $p_i + \beta_i x_i$, thus x_i must be a rational number of the form

$$(t - p_i)/\beta_i, \text{ with some } t \in \mathbb{Z}.$$

If lower and upper bounds on the components of $\mathbf{x}$ are known, then only a finite number of possible solutions remain to check. The bounds can be determined by linear programming: the variables x_i need to be minimized and maximized subject to the system of inequalities $\mathbf{a} \geq \mathbf{0}$ as linear constraints. The total number of the vectors $\mathbf{x}$ satisfying the bounds can be still to high to enumerate all of them explicitly. Therefore first the two bounds are determined for one of the variables only, say for x_1. Then for every possible value of x_1 the two bounds are determined for another variable, etc.

This procedure can be implemented in many different ways. The easiest (but not necessarily the most efficient) way is bounding each variable from below and above and using enumeration instead of repeated bounding. Another straightforward method is using linear programming twice in every iteration. Enumeration can be improved using the method of Land and Doig (see [7, section 3.6]). The main point of this method is that if $x_1, ..., x_{i-1}$ are fixed and the lower an upper bounds of x_{i+1} are considered as the function of x_i then the sign of the first difference of the bounds can change only once each. Furthermore if by changing x_i, i.e. by increasing or decreasing x_i, the set of feasible solutions becomes empty then it remains empty if x_i is changed further on in the same direction.

It follows from the integrality of the coefficients that the set of integer feasible solutions is unbounded only if the set of feasible solutions of the LP relaxation is unbounded, too. If the set of solutions is unbounded, then we can still use the LP-based enumerative algorithm to obtain every solution not longer than some given constant. If this constant is determined by one of the bounds presented in section 3.2, then these solutions include every minimal solution.

This type of algorithm is called *LP-based enumerative* method.

4 Determination of elementary reactions

In this section three different applications of the theory discussed so far are presented. All of them are well-known problems in chemistry.

4.1 Examples

Permanganate/oxalic acid reaction [6]. This reaction has been investigated since 1864, but its decompositions are still unknown. The mechanism includes 19 species, which consist of four elements: manganese, hydrogen, coal and oxygen. The species are listed in Table 1, the overall reaction is the following:

$$2MnO_4^- + 6H^+ + 5H_2C_2O_4 \;\rightarrow\; 2Mn^{2+} + 8H_2O + 10CO_2$$

	C	H	Mn	O	c
$H_2C_2O_4$	2	2	0	4	0
$HC_2O_4^-$	2	1	0	4	−1
H^+	0	1	0	0	1
$C_2O_4^{2-}$	2	0	0	4	−2
Mn^{2+}	0	0	1	0	2
MnC_2O_4	2	0	1	4	0
MnO_4^-	0	0	1	4	−1
MnO_2	0	0	1	2	0
Mn^{3+}	0	0	1	0	3
CO_2	1	0	0	2	0
H_2O	0	2	0	1	0
$[MnO_2, H_2C_2O_4]$	2	2	1	6	0
CO_2^-	1	0	0	2	−1
$[Mn(C_2O_4)]^+$	2	0	1	4	1
$[Mn(C_2O_4)_2]^-$	4	0	1	8	−1
$[MnC_2O_4, MnO_4^-, H^+]$	2	1	2	8	0
$[MnC_2O_4^{2+}, MnO_3^-]^+$	2	0	2	7	1
$[MnC_2O_4^{2+}, MnO_3^-, H^+]^{2+}$	2	1	2	7	2
$[H^+, MnO_2, H_2C_2O_4]^+$	2	3	1	6	1

Table 1: The species taking part in the permanganate/oxalic acid reaction

To obtain every possible elementary step it is required to represent the vector of each species, and the sum of any two (not necessarily different) species as the nonnegative integral linear combination of the others in every possible way. Altogether there are 209 systems of equations to solve. Although we found the Contejean–Devie algorithm competitive in some of these equations, but in many cases, when the length of a few solutions exceeded four, i.e. there are at least five molecules on the right-hand side of the reaction, the LP-based enumerative algorithm was significantly faster. In the end, the search for possible elementary reactions resulted in 1022 steps, which were obtained in 52 seconds.[1]

Air pollution. For a current research at the Dept. of Fuel and Energy at University of Leeds we had to generate the list of those elementary reactions consisted of the species given in Table 2 that contain sulfur.

H_2	CH_4	C_2H_2	C_2H_4	C_2H_6	C_3H_4	C_3H_6	C_4H_2
O_2	H_2O	H_2O_2	CO	CO_2	CH_2O	CH_2CO	C
H	CH	CH_2	$CH_2(S)$	CH_3	C_2H	C_2H_3	C_2H_5
C_3H_2	H_2CCCH	H_2CCCCH	O	OH	HO_2	HCO	CH_3O
CH_2OH	HCCO	CH_2HCO	N_2	CN	HCN	N	NH
NO	HNO	NH_2	H_2NO	NCO	N_2O	NO_2	N_2H_2
HOCN	H_2CN	NNH	NH_3	N_2H_3	C_2N_2	HNCO	NO_3
S	SH	H_2S	SO	SO_2	SO_3	HSO_2	HOSO
$HOSO_2$	SN	S_2	CS	COS	HSNO	HSO	HOS
HSOH	H_2SO	HOSHO	HS_2	H_2S_2	CS_2	HSOO	H_2SO_4

Table 2: The species of the Air Pollution example. Species containing sulfur are below the dashed line.

In this case we had 1668 equations with 78 or 79 variables. The number of solutions was above 400,000, which was too much for further investigations. So the model was changed so that only steps with at most three products are taken into account.

The modified Contejean–Devie algorithm can be used directly to obtain every such solution, and since in this case the length of the solutions is very small, this algorithm is expected to be very fast. On the other hand the use of the LP-based enumerative algorithm is also reasonable. In this particular example the improved Contejean–Devie algorithm turned out to be the fastest one, this is due to the very limited size of the solutions. The original Contejean–Devie algorithm performs significantly worse in the example.

Oxalate–persulfate–silver oscillator [2]. This mechanism presented in [2] consists of exactly the 16 species shown in Table 3. The 152 systems derived have 89 solutions. Our

[1]All timing data relate to a PC with AMD Athlon XP 1666 MHz, 121.8 MIOps CPU, 768 MB RAM, running Windows XP and *Mathematica* 5.0

implementation of the Contejean–Devie algorithm needed about two minutes to find each solution, while the LP-based enumerative algorithm was ready in 2 seconds. In this very simple problem the naive algorithm works almost as fast as the LP-based enumerative algorithm.

Ag^+	Ag^{2+}	H^+	SO_4^-
SO_4^{2-}	$S_2O_8^{2-}$	$C_2O_4^{2-}$	$Ag(C_2O_4)^-$
OH	H_2O	CO_2^-	O_2
HO_2	H_2O_2	$O_2CO_2^-$	CO_2

Table 3: Species in the oxalate–persulfate–silver oscillator

As all the three examples show that in general the algorithm of Contejean and Devie is far less efficient than the LP-based enumerative algorithm. The former is only recommended when both the size of the problem and the length of the solutions are very small. Our modification of this algorithm proved to be a significant improvement.

We used the `LinearProgramming` function of *Mathematica* 5 to solve the linear programming instances created during the LP-based enumerative algorithm. Due to several bugs in this function its most efficient methods may malfunction in some (rare) cases. If it is parameterized to use the correctly implemented simplex method, the efficiency of our algorithm decreases dramatically. The table below shows the timing data of the fastest version of our implementation.

Problem	No. systems	equations/ variables[2]	Algorithm used	CPU time (s)
Permanganate/ oxalic acid	209	5 / 17	naive BFS	163.04
			Contejean–Devie	1589.48
			LP-based enumerative	51.66
Air Pollution	1668	5 / 78	naive BFS	> 8 hrs.
			improved Contejean–Devie	13429.5
			original Contejean–Devie	> 8 hrs.
			LP-based enumerative	> 8 hrs.
Oxalate/ persulfate/ silver	152	6 / 14	naive BFS	3.10
			Contejean–Devie	116.67
			LP-based enumerative	2.00

Table 4: Timing measurements on the examples (generating elementary reactions).

5 Decomposition of overall reactions

5.1 Reduction of the searching space

A complex overall reaction might have a large number of decompositions. Therefore it is important to reduce the search space to enhance the efficiency of the algorithms.

5.1.1 Reactions that take part in every decomposition

There are some elementary steps which must take part in every decomposition. These steps can be subtracted from the overall reaction, and the length of the solutions can be decreased. These elementary reactions can be selected via linear programming.

Let $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$, and $x_1, \ldots, x_n$ be the vectors of the elementary reactions, and their multiplicities, respectively. The vector of the overall reaction is denoted by $\mathbf{b}$. Then the linear program to solve during the investigation of the k-th reaction is the following:

$$\min x_k; \quad \sum_i x_i \mathbf{v_i} = \mathbf{b}, \ \forall\, i\colon \ x_i \geq 0$$

The minimum obtained in the k-th linear program is obviously a lower bound on the multiplicity of the k-th reaction. The main advantage of this preprocessing procedure is its speed.

5.1.2 Elimination of never used reactions

Every decomposition method becomes faster if those elementary reactions are eliminated, which cannot take part in any decomposition. Three methods are presented for this purpose in this subsection.

Elimination by chemical expert. As a first filter, we should rule out every elementary step which are chemically impossible, e.g. because they disobey some thermodynamical law. In different cases different rules might be applied.

LP-based solution. The LP-based method that was used to bound the multiplicities of the reactions can be used again, but we can use it in a more sophisticated way, as proposed in [6].

1. **[init]** $M := \emptyset$
2. **[test]** Find an arbitrary $(a_1, \ldots, a_n)$, $a_i \in \mathbb{Q}_0^+$ satisfying (1) such that $\{i \mid a_i > 0\} \not\subset M$. If there is no such solution, stop.
3. **[jump]** $M = M \cup \{i \mid a_i > 0\}$, and go to step 2.

Step 2 is carried out by the solution of the following linear programming problem:

$$\min \mathbf{c}^{\mathrm{T}}\mathbf{x}; \quad \sum_{i=1}^{n} x_i \mathbf{v_i} = \mathbf{b},\ \mathbf{c}^{\mathrm{T}}\mathbf{x} \geq 1,\ \forall\, i\colon\ x_i \geq 0, \tag{7}$$

where the i-th component of the n dimensional vector $\mathbf{c}$ is 1 if $i \notin M$ and 0 otherwise.

It is clear, that the procedure terminates after a finite number of steps, because in every iteration it either stops or decreases the number of indices not included in set M, by at least one. Those vectors, which remain unmarked after termination cannot take part in any decomposition. It is worth noting, that the LP relaxation problem of the second step also has chemical meaning. The decomposition with nonnegative rational coefficients yields a decomposition of a multiple of the overall reaction. During further chemical investigations we might find these decompositions useful.

In Step 2 any other objective function can be used for which the minimum exists. But the conditions must be the same to find a decomposition including at least one unmarked elementary reaction. Different objective functions can give different decompositions. If vector $\mathbf{c}$ has only positive components, the algorithm determines only minimal decompositions.

Volpert indices. The previous algorithm uses only mathematical reasoning, but does not take chemical aspects into consideration. A decomposition is chemically infeasible if any of its species has zero concentration during the whole reaction. (Concentrations are determined via solving differential equations associated with the decompositions.) According to a theorem of Volpert a species has a constant zero concentration if and only if it gets infinite index in the following procedure [14]: first, the initial species get the index zero. Then the index of a reaction is the maximum of the indices of its reactants, while the index of a not initial species is one bigger than the minimum of the indices of those reactions which produce the species in question. (The minimum of the empty set is declared infinity.) More formally, let the set of species M and the set of reactions R be given and denote the set of initial species by M_0, finally let $R(r)$ and $P(r)$ be the reactants and the products of the reaction r. Then the indexing algorithm is the following.

1. **[init]** $R_0 := \{r \in R \mid R(r) \subset M_0\}, \quad i := 1.$
2. **[species indices]** $M_i := \bigcup_{j=0}^{i-1} \bigcup_{r \in R_j} P(r) \setminus \bigcup_{j=0}^{i-1} M_j$
3. **[reaction indices]** $R_i := \{r \in R \mid R(r) \subset \cup_{j=0}^{i} M_j\} \setminus \bigcup_{j=0}^{i-1} R_j.$
4. **[cycle]** If $R_i = \emptyset$, go to step 5. Otherwise $i := i + 1$ and go to step 2.
5. **[end]** $M_\infty := M \setminus \bigcup_{j=0}^{i} M_j, \quad R_\infty := R \setminus \bigcup_{j=0}^{i} R_j$

The Volpert-index of a species or a reaction is j if it is contained in the set M_j or R_j.

Although this procedure is used to check the chemical feasibility of the decompositions (see e.g. [6, 14]) with a slight modification it can also be used in preprocessing. One can consider all species and all elementary reactions as a single decomposition. After indexing it, the reactions and species with infinite index cannot take part in any decomposition.

Volpert-indexing is extremely efficient as both of its space and time complexity is linear.

5.2 Example

Permanganate/oxalic acid reaction [6]. In the previous section 1022 elementary reactions have been generated that may take place during the overall reaction. Many of these steps are chemically infeasible. Using chemical evidence 349 reactions have been omitted. The remaining 673 steps were investigated using the methods described in section 5.1.

First we found, that two reactions must take part in every decomposition. The step $H_2C_2O_4 + MnO_2 \rightarrow [MnO_2, H_2C_2O_4]$ takes place at least four times, and similarly the step $2\,CO_2^- \rightarrow C_2O_4^{2-}$ takes place at least once. With this observation we can rearrange the system of equations so the length of the solutions are decreased by 5. This step took about 2 minutes CPU time.

The LP-based method were applied with three different objective functions to determine the elementary reactions which can be eliminated. The first one, exactly formulated as in (7) found 314 steps that does not take part in any decompositions. The number of iterations were 313. As it was expected less number of iterations (exactly 280) was required with the objective function $(\mathbf{1} - \mathbf{c})^{\mathrm{T}}\mathbf{x}$. The third variation was the constant zero as objective functions. This step needed about 5–7 minutes depending on the objective function.

The remaining reactions were indexed using Volpert's algorithm with different set of initial species. Only 297 reactions and only 16 species got finite index using either the set of 5 initial species suggested in [6] or the set consisted of every non-complex species. In fact, we got the same reaction set after indexing the original set of 673 reactions. This step took less than a second in each case.

In the determination of the decompositions only the remaining 297 elementary steps were used. To accelerate the search linear programming was applied to obtain bound on the length of the smallest decomposition. The objective function was the sum of the coefficients of the elementary steps and as a by-product the following decomposition was obtained:

$4\times$	$H_2C_2O_4 + MnO_2$	$\rightarrow$	$[MnO_2, H_2C_2O_4]$
$2\times$	$H_2C_2O_4 + [MnO_2, H_2C_2O_4]$	$\rightarrow$	$2CO_2 + 2H_2O + MnC_2O_4$
$1\times$	$H^+ + HC_2O_4^-$	$\rightarrow$	$H_2C_2O_4$
$1\times$	$C_2O_4^{2-} + H^+$	$\rightarrow$	$HC_2O_4^-$
$2\times$	$MnC_2O_4 + MnO_4^-$	$\rightarrow$	$CO_2^- + CO_2 + 2MnO_2$
$1\times$	$2CO_2^-$	$\rightarrow$	$C_2O_4^{2-}$
$2\times$	$H^+ + [MnO_2, H_2C_2O_4]$	$\rightarrow$	$[H^+, MnO_2, H_2C_2O_4]^+$
$2\times$	$H^+ + [H^+, MnO_2, H_2C_2O_4]^+$	$\rightarrow$	$2CO_2 + 2H_2O + Mn^{2+}$

It is interesting to compare the performance of the two main algorithms in this problem. Using the LP-based enumerative algorithm it turned out in half a second that no solutions exist among the vectors shorter than 14. The Contejean–Devie algorithm ran for 25 minutes to reach the same conclusion even if the two obligatory elementary steps were taken into consideration.

Using our LP based algorithm we determined every solution not longer than 17 units. With problems of this size the Contejean–Devie algorithm is useless even in its improved version. At this point we reached the limits of the LP-based enumerative algorithm as the computational time was almost 4 hours. Finally 3170 decompositions were found. Out of them there were 1918 which proved to be indexable. (The preprocessing algorithms also produced many decompositions.) Our results are summarized in Table 5.

Step	CPU time (s)	LPs solved	Results
Filtering obligatory reactions	124.80	673	2 reactions found
LP-based reduction with $\mathbf{c}^{\mathrm{T}}$ in (7)	329.19	313	185 decompositions
LP-based reduction with $\mathbf{1}-\mathbf{c}^{\mathrm{T}}$ in (7)	443.11	280	230 decompositions
LP-based reduction with $\mathbf{0}$ in (7)	330.00	313	189 decompositions
LP-based reduction aggregated results	1102.30	906	420 decompositions
Volpert-indexing	0.5	—	110 indexable solutions
Generating decompositions of ≤ 14 steps	0.40	1	0 decompositions
Generating decompositions of ≤ 15 steps	656.56	648	11 decompositions
Generating decompositions of ≤ 16 steps	1096.44	8160	235 decompositions
Generating decompositions of ≤ 17 steps	13619.50	186515	3170 decompositions
Volpert-indexing	8.28	—	1918 indexable solutions

Table 5: Results of the decompositioning process of the permanganate/oxalic acid reaction.

6 Conclusion

In the paper four algorithms of solving linear Diophantine equations over nonnegative integers were discussed and compared to each other with respect to their usability in the decomposition of complex chemical reactions. New results in the decomposition of a complex reaction of high importance are presented.

The naive breadth-first-search type algorithm and Domenjoud's algorithm is recommended only if the size of the system is small.

The method of Contejean and Devie is applicable only if the solutions of the system are small. For these systems the modification proposed in this paper proved to be significant. Still this algorithm is only applicable to the generation of elementary reactions, but not to the decomposition of overall reactions. With regard to the solution of Diophantine equations in general, it is only useful if the length of the solutions is very small.

The LP-based enumerative algorithm, proposed in this paper, proved to be competitive in most of the cases. For large systems with large solutions, like those derived from the decomposition, it was the only applicable method.

References

[1] Ahlgren, S., Ono, K.: ,,Addition and Counting: the Arithmetic of Partitions", *Notices of the AMS*, **48** (9) (2001), 978–984.

[2] Clarke, B. L., ,,Stoichiometric network analysis of the oxalate–persulfate–silver oscillator", *Journal of Chemical Physics*, **97** (4) (1992), pp. 2459–2472.

[3] Contejean, E., Devie, H., ,,An efficient incremental algorithm for solving systems of linear Diophantine equations", *Information and Computation*, **113** (1994), 143–172.

[4] Domenjoud, E., ,,Solving Systems of Linear Diophantine Equations: an Algebraic Approach", *Mathematical Foundations of Computer Science*, 1991, pp. 141–150.

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

[6] Kovács K., Vizvári B., Riedel M, Tóth J., ,,Decomposition of the permanganate/oxalic acid overall reaction to elementary steps based on integer programming theory", *Phys. Chem. Chem. Phys.*, 2004 (6), pp. 1236–1242.

[7] Kovács, L. B., *Combinatorial Methods of Discrete Programming*, Akadémiai Kiadó, Budapest, 1980.

[8] Lichtblau, D., ,,Revisiting strong Gröbner bases over Euclidean domains", Wolfram Research, Inc., Champaign, 2001.

[9] Papadimitriou, C. H., *Computational Complexity*, Addison–Wesley, 1994.

[10] Pottier, L., ,,Minimal solutions of linear diophantine systems: bounds and algorithms", in ed. R.V. Book, Proc. 4th Conference on Rewriting Techniques and Applications, Como, Italy, *LNCS*, **488**, pp. 162-173.

[11] Smart, N., *The algorithmic resolution of Diophantine equations*, Cambridge UP, Cambridge, 1988.

[12] Szalkai I., ,,A new general algorithmic method in reaction syntheses using linear algebra", *Journal of Mathematical Chemistry*, **28** (2000)

[13] Tomlin, A. S., Turányi, T., Pilling, M. J., Mathematical tools for the construction, investigation and reduction of combustion mechanisms, In: *Comprehensive Chemical Kinetics*, Eds. Compton, R. G. and Hancock, G., Vol. 35, Elsevier, 1997.

[14] Volpert, A. I., Chudayev, S. I., *Analysis in classes of discontinuous functions and the equations of mathematical physics*, Nauka, Moszkva, 1975.

[15] de Weger, B. M. M., *Algorithms for Diophantine equations*, CWI, Amsterdam, 1989.