Parameter Identification of Canalyzing Boolean Functions with Ternary Vectors for Gene Networks

Annika Eichler[1] and Gerwald Lichtenberg[2]
[1]*Automatic Control Laboratory, ETH Zurich, Physikstrasse, Zurich, Switzerland*
[2]*Faculty Life Sciences, Hamburg University of Applied Sciences, Ulmenliet, Hamburg, Germany*

Keywords: Parameter Identification, Networks, Gene Dynamics, Systems Biology, Boolean Functions, Ternary Logic.

Abstract: In gene dynamics modeling, parameters of Boolean networks are identified from continuous data under various assumptions expressed by logical constraints. These constraints may restrict the dynamics of the network to the subclass of canalyzing functions, which are known to be appropriate for genetic networks. This paper introduces a high performance algorithm, which solves the parameter identification problem by so called Zhegalkin identification and exploits the restriction to canalyzing functions resulting in reduced calculation time. The canalyzing constraint is formulated in terms of orthogonal ternary vector lists - which are intrinsically used in a Branch-and-Cut algorithm obeying this constraint. The algorithm is applied to mRNA micro array data from mice under different contaminant conditions and good correspondence to a known apoptotic pathway can be shown.

1 INTRODUCTION

A current field of research in systems biology is gene dynamics modeling, since understanding the dynamics of the genetic model could help the therapeutic process (Lin and Khatri, 2013). Canalyzing Boolean functions have shown to be appropriate to model genetic networks, due to their common characteristics, as periodicity, global complexity and self organization (Kauffman, 1993). In genetic networks canalization is the ability of a genotype to produce the same phenotype regardless of environmental variability (Jarrah et al., 2007). Thus, due to their stabilizing effect on the discrete dynamical behavior, they turned out to describe the highly ordered dynamics of gene networks better than other Boolean models (Kauffman et al., 2003).

A successful approach to identify parameters of Boolean functions from contiuous-valued signals like microarray data uses Zhegalkin polynomials to represent these functions, see Lichtenberg et al. (2005); Faisal et al. (2010); Veliz-Cuba et al. (2010); Breindl et al. (2013). The Zhegalkin identification problem is a Mixed Integer Quadratic Program (MIQP) which can in principle be solved with standard tools like CPLEX or Xpress, where Branch-and-Cut algorithms are used. One major problem of Boolean identification is the exponential growth of the cardinality of the solution set with the number of interacting genes. Thus, those methods are applicable up to a model order of $n = 10$, where already very large runtimes of hours or days occur, Faisal (2008).

Furthermore, a clustering problem has to be solved to determine groups of genes of unknown cardinality—denoted *connectivity degree*—which affect each other. Combining the clustering and the Zhegalkin identification problem leads to a problem of discrete optimization with even higher complexity. First approximations for the solution of this combined problem have been found by a preprocessing step based on the Pearson Correlation Coefficient in Faisal (2008). Next, exploiting efficient representations of Zhegalkin polynomials as orthogonal ternary vector lists (OTVLs), (Bochmann and Steinbach, 1991), and adapting tensor decomposition techniques from Kolda and Bader (2009) allows integration of both steps reported in Lichtenberg and Eichler (2011). Moreover, the solution set of the identification algorithm can be reduced by fixing the maximum number of rows of the OTVL representing the solution. This leads to highly efficient computation with controllable degree of accuracy, because optimality of the solution is guaranteed by a Branch-and-Cut algorithm used for the reduced solution set.

In this paper, the latter method is restricted to the subclass of canalyzing functions due to their interest-

Eichler, A. and Lichtenberg, G.
Parameter Identification of Canalyzing Boolean Functions with Ternary Vectors for Gene Networks.
DOI: 10.5220/0005978701100118
In *Proceedings of the 6th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2016)*, pages 110-118
ISBN: 978-989-758-199-1
Copyright © 2016 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

ing properties. This introduces additional constraints for the optimization problem, as already reported in Faisal et al. (2006) and Breindl et al. (2013), but the reduced solution set is not efficiently exploited therein. This work shows how to incorporate those constraints in the identification algorithm by expressing canalizing functions as OTVLs. The proposed algorithm for the identification of canalyzing functions is by orders of magnitude more efficient since the search space is considerably reduced as obvious from Table 1. The adapted identification is applied to gene expression data from mRNA extracted from mouse liver cells.

This work is organized as follows. Section 2 introduces fundamentals of Boolean functions, Zhegalkin polynomials and OTVLs. In Section 3 the Branch-and-Cut Boolean identification algorithm from Lichtenberg and Eichler (2011) is described. Section 4 presents how to express canalyzing functions as OTVLs and adapt the identification therefore. The results on an application to real data are shown in Section 5. Finally conclusion are drawn in Section 6.

2 FUNDAMENTALS

The set $\mathbb{B}=\{0,1\}$ denotes the set of logicals, $\mathbb{U}=[0,1]$ the unit interval. Negation of Booleans is denoted by $\neg z=\bar{z}$, for real ones $\bar{x}=1-x$ holds. With $\otimes$ the Kronecker product is denoted.

2.1 Boolean Functions and Zhegalkin Polynomials

A Boolean function $b:\mathbb{B}^n\to\mathbb{B}$ can be represented by its truth vector $\mathbf{b}=(b_1,...,b_{2^n})'\in\mathbb{B}^{2^n}$, i.e. the last column of the truth table as shown in Table 2.

Example 1. Consider the Boolean function

$$b(y_1,y_2)=\neg(y_1\wedge y_2), \tag{1}$$

which is given by the truth table

y_2	y_1	$b(y_1,y_2)$
0	0	1
0	1	1
1	0	1
1	1	0

(2)

with its truth vector. $\mathbf{b}=\begin{pmatrix}1 & 1 & 1 & 0\end{pmatrix}'$.

Definition 1. A *Zhegalkin polynomial* $p(\mathbf{y})=\mathbf{l}(\mathbf{y})'\mathbf{b}$ is a multilinear polynomial with $\mathbf{b}\in\mathbb{B}^{2^n}$ being a truth vector and $\mathbf{l}(\mathbf{y})$ the so called *literal vector*, given by Lichtenberg and Eichler (2011) as

$$\mathbf{l}(\mathbf{y})=\begin{pmatrix}\bar{y}_n\\ y_n\end{pmatrix}\otimes\cdots\otimes\begin{pmatrix}\bar{y}_1\\ y_1\end{pmatrix}\in\mathbb{U}^{2^n}. \tag{3}$$

Table 1: Number of all Boolean functions and the canalyzing ones.

n	Boolean functions	CFs
1	4	4
2	16	14
3	256	120
4	65536	3514
5	$4.2950\cdot10^9$	1292276
6	$1.8447\cdot10^{19}$	$1.0307\cdot10^{11}$

Table 2: Truth table.

y_n	$\cdots$	y_2	y_1	$b(y_1,...,y_n)$
0	$\cdots$	0	0	b_1
0	$\cdots$	0	1	b_2
0	$\cdots$	1	0	b_3
0	$\cdots$	1	1	b_4
$\vdots$		$\vdots$	$\vdots$	$\vdots$
1	$\cdots$	1	1	b_{2^n}

Proposition 1 (Zhegalkin (1928))**.** *A Zhegalkin polynomial evaluated at Boolean values* $\mathbf{y}\in\mathbb{B}^n$ *gives the same (Boolean) result as the Boolean function represented by the truth vector* $\mathbf{b}$.

Thus the Zhegalkin polynomials can be seen as the bridge between the Boolean and the real set $\mathbb{U}$. Since if $\mathbf{y}\in\mathbb{U}$ then $p(\mathbf{y})\in\mathbb{U}$ as well, if however $\mathbf{y}\in\mathbb{B}$ then $p(\mathbf{y})\in\mathbb{B}$.

Example 1. ***(continued)*** To illustrate this for the Boolean function (1) the corresponding Zhegalkin polynomial is calculated as

$$\begin{aligned}\mathbf{l}'(\mathbf{y})\mathbf{b}&=\begin{pmatrix}(1-y_1)(1-y_2)\\ y_1(1-y_2)\\ (1-y_1)y_2\\ y_1y_2\end{pmatrix}'\begin{pmatrix}1\\1\\1\\0\end{pmatrix}\\ &=1-y_1y_2.\end{aligned} \tag{4}$$

It can be easily seen that if $y_1,y_2\in\mathbb{B}$, then the Zhegalkin polynomial leads to the same solution as the Boolean function (1), as declared in Proposition 1.

2.2 Ternary Vector Lists

Ternary Vector Lists (TVLs) are a common concept in Boolean algebra, because of its outstanding advantages for large scale problems, Bochmann and Steinbach (1991). A TVL of a Boolean function represents all elements of the Boolean space $\mathbb{B}^{2^n}$ where the function is 1 by ternary vectors (TVs). A TV $\mathbf{t}$ has the structure

$$\mathbf{t}\in\mathbb{T}^n=\{0,1,-\}^n. \tag{5}$$

A zero element '0' in the TV describes that the corresponding variable appears negated, a one element '1'

that it appears not negated. The latter '$-$' is the *don't care* symbol, that can stand for either '1' or '0'.

A TVL with k lines is of the form

$$\mathbf{T} = \begin{bmatrix} \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_k \end{bmatrix}.$$

Taking all lines of the truth table with ones always leads to a valid TVL of a Boolean function. TVLs with smaller number of lines might be possible by using '$-$'.

Example 1. ***(continued)*** With the truth table in (2) valid TVLs for the Boolean function (1) of the running example are

$$\mathbf{T_1} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad \mathbf{T}_2 = \begin{bmatrix} 0 & 1 \\ - & 0 \end{bmatrix}, \tag{6}$$

$$\mathbf{T}_3 = \begin{bmatrix} 0 & - \\ - & 0 \end{bmatrix}, \qquad \mathbf{T}_4 = \begin{bmatrix} 0 & - \\ 1 & 0 \end{bmatrix}. \tag{7}$$

This can easily be checked by replacing '$-$' with both '0' and '1'.

This example shows that TVLs are not unique, i.e there exist different TVLs for the same Boolean function. Another important property is orthogonality (Bochmann and Steinbach, 1991).

Definition 2. A TVL $\mathbf{T}$ is orthogonal, if each binary vector appears only once in $\mathbf{T}$. This is the case, if for any pair of lines of $\mathbf{T}$ in at least one column a (0,1)-combination appears. Two TVLs $\mathbf{T}_A$ and $\mathbf{T}_B$ are orthogonal if $\mathbf{T}_A$ and $\mathbf{T}_B$ have no binary vectors in common. This is the case if for any pair of lines of $\mathbf{T}_A$ and $\mathbf{T}_B$ in at least one column a (0,1)-combination appears.

A binary vector (BV) is a vector with only '0's and '1's. It can represent only one line of the truth table, while a ternary vector (TV) due to '$-$' can represent multiple BVs.

Example 1. ***(continued)*** For the TVLs of the example it is obvious that all TVL representations are orthogonal except of $\mathbf{T}_3$ with no (0,1)-combination in any column. Here the binary vector $\begin{bmatrix} 0 & 0 \end{bmatrix}$ appears in both lines.

In the following an orthogonal TVL is denoted as OTVL. In Bochmann and Steinbach (1991) operations for OTVLs are described. Important for this work are the complement and the difference operators, which are visualized in Table 3 for 3 variables. The complement $\mathrm{CPL}(\mathbf{T}) = \overline{\mathbf{T}}$ of a given OTVL $\mathbf{T}$ is defined as the OTVL of all binary vectors that are not in $\mathbf{T}$. The difference $\mathrm{DIF}(\mathbf{T}_A, \mathbf{T}_B)$ of the OTVLs $\mathbf{T}_A$ and $\mathbf{T}_B$ results in an OTVL of all BVs, that are in $\mathbf{T}_A$ but not in $\mathbf{T}_B$. If the result is the empty OTVL, $\mathbf{T}_A$ is totally included in $\mathbf{T}_B$.

Table 3: Graphical representation of operands for TVLs, Bochmann and Steinbach (1991).

$\mathbf{T_A} = \begin{bmatrix} 0 & 0 & - \\ 1 & - & 1 \end{bmatrix}$ 	$\mathrm{CPL}(\mathbf{T_A}) = \begin{bmatrix} 0 & 1 & - \\ 1 & - & 0 \end{bmatrix}$ 
$\mathbf{T_B} = \begin{bmatrix} 1 & - & - \end{bmatrix}$ 	$\mathrm{DIF}(\mathbf{T_A},\mathbf{T_B}) = \begin{bmatrix} 0 & 0 & - \end{bmatrix}$ 

Lemma 1. *An OTVL* $\mathbf{T}$ *is orthogonal to its complement* $\overline{\mathbf{T}}$.

Proof. With Definition 2 two TVLs are orthogonal, if they do not have any BVs in common. The complement of an OTVL $\mathbf{T}$ contains all BVs, that are not in $\mathbf{T}$ and is thus orthogonal to $\mathbf{T}$. □

Proposition 2. *For an OTVL* $\mathbf{T}$ *with k lines the number of ones in the correspondoing truth vector* $\mathbf{b}$ *is* $N_1 = \mathbf{b}'\mathbf{1} = \sum_{i=1}^{k} 2^{N_{i-}}$ *where* N_{i-} *is the number of '*$-$*'s in the i-th line of* $\mathbf{T}$.

Proof. The number ones in $\mathbf{b}$ is equivalent to the number of BVs in $\mathbf{T}$. A TV with no '$-$'s represents a single BV and since a '$-$' stands for either 1 or 0, a TV with N_- times the '$-$' symbol, includes 2^{N-} BVs. Due to orthogonality no BV appears more than once in $\mathbf{T}$, so that the number of BVs in each line can simply be added. □

2.3 OTVLs and Zhegalkin Polynomials

Since OTVLs and Zhegalkin polynomials are two different representations of Boolean functions, it is possible to find the corresponding mapping between both representations.

Proposition 3. *Given is an OTVL* $\mathbf{T}$ *of n variables, that is representing a Boolean function* f*, then the corresponding Zhegalkin polynomial, determined by* $p_{\mathbf{T}}$*, is calculated as*

$$p_{\mathbf{T}}(\mathbf{y}) = \sum_{j=1}^{k} \prod_{i=1}^{n} T(t_{ji}, y_i) \tag{8}$$

$$\text{with } T(t_{ji}, y_i) = \begin{cases} \overline{y_i}, & \text{if } t_{ji} = 0, \\ y_i, & \text{if } t_{ji} = 1, \\ 1, & \text{if } t_{ji} = -. \end{cases}$$

Proof. Assume $\mathbf{T}$ is an OTVL, i.e. without '$-$'s, then $\prod_{i=1}^{n} T(t_{ji}, y_i)$ corresponds to the l-th row of the literal vector. Since $\mathbf{t}_j$ is only a line of $\mathbf{T}$ when $\mathbf{b}_l = 1$ due to the construction of an OTVL, (8) is equal to $\mathbf{l}(\mathbf{y})'\mathbf{b}$, what finishes the proof for OTVLs without '$-$'s. If $\mathbf{T}$ is an OTVL with a '$-$' in the k-th column, than this is equal to a TVL with the same row and a '1' in the k-th column and additionally the same row and a '0' in the k-th column. For the row with the '1', if it is the

n-th row, it is $\prod_{i=1}^{n} T(t_{ni}, y_i) = y_k \prod_{i=1, i\neq k}^{n} T(t_{ni}, y_i)$, and for that with the '0', if it is the m-th row, it is $\prod_{i=1}^{n} T(t_{mi}, y_i) = \bar{y_k} \prod_{i=1, i\neq k}^{n} T(t_{mi}, y_i)$. Thus the sum is $(y_k + \bar{y_k}) \prod_{i=1, i\neq k}^{n} T(t_{mi}, y_i) = \prod_{i=1, i\neq k}^{n} T(t_{mi}, y_i)$, since $\prod_{i=1, i\neq k}^{n} T(t_{mi}, y_i) = \prod_{i=1, i\neq k}^{n} T(t_{ni}, y_i)$. What finishes the proof for all OTVLs. □

Example 1. ***(continued)*** Let's consider

$$\mathbf{T}_2 = \begin{bmatrix} 0 & 1 \\ - & 0 \end{bmatrix}$$

of the running example. Evaluating (8) for $\mathbf{T}_2$ leads to

$$p(\mathbf{y}) = \bar{y_1} y_2 + 1\bar{y_2} = (1 - y_1) y_2 + (1 - y_2) = 1 - y_1 y_2$$

as derived with the literal form (4) before.

3 ZHEGALKIN IDENTIFICATION BY BRANCH-AND-CUT ALGORITHM

Finding the best Boolean model for continuous normalized data is known as *Zhegalkin identification* problem, see Faisal et al. (2005), that has been shown to be well suited for Boolean identification of gene networks (Faisal, 2008; Veliz-Cuba et al., 2010; Breindl et al., 2013). In Lichtenberg and Eichler (2011) the Zhegalkin identification problem is solved with the help of OTVLs by a Branch-and-Cut algorithm.

In contrast to the first references, the efficient algorithm in Lichtenberg and Eichler (2011) allows to include this clustering problem in the identification. A cluster is denoted as the set of genes, which affects the dynamics of a gene of interest, since a gene is never affected by all others genes, but only a subset, the cluster. The size of the cluster, called connectivity degree, and the cluster itself are unknown and have to be determined in the clustering problem.

Before the main contribution, how OTVLs of canalyzing functions are structured and how to restrict the identification to canalyzing functions, the Zhegalkin identification algorithm from Lichtenberg and Eichler (2011) is shortly introduced here.

3.1 Minimization Problem

A Zhegalkin function of n signals can be modeled by n truth vectors or the respective OTVLs. The state space model for signal l is then given as

$$y_l(t+1) = \mathbf{l}(\mathbf{y}(t))' \mathbf{b}_l = p_{\mathbf{T}_l}(\mathbf{y}(t)), \qquad \forall l = 1, \ldots, n \tag{9}$$

with $p_{\mathbf{T}_l}(\mathbf{y})$ as defined in (8). The prediction error between $y_l(t+1)$ predicted with the OTVL $\mathbf{T}_l$ as model as in (9) and the measurement value $\tilde{y}_l(t+1)$ of signal l at any time $t = 0, \cdots, T-1$ is defined as $d_l(t+1) = y_l(t+1) - \tilde{y}(t+1)$. The task of the Zhegalkin identification problem is to find the optimal OTVL $\mathbf{T}_l^\star$ and the corresponding Zhegalkin polynomial that solves the minimization problem

$$\min_{\mathbf{T}_l} J_l \quad \text{with} \quad J_l = \sqrt{\sum_{t=0}^{T-1} d_l(t)^2} \tag{10}$$

with J_l, the 2-norm of the prediction error, being the error function. It is clear that this minimization problem has to be solved for all signals $l = 1, \ldots, n$. Therefore this index is omitted in the following.

One major problem of Boolean and thus Zhegalkin identification is the high cardinality of the search space. There exist $2^{(2^n)}$ different Boolean functions of n variables. This fast growth in the number of variables n is exemplarily shown in Table 1. To deal with this problem, the algorithm presented here finds the best approximation $\mathbf{T}^+$ with fixed maximal number of rows, instead of searching for the optimal solution. This row restriction significantly reduces the search space by preserving the basic properties as it is approved in Section 5 by the numerical example.

3.2 Branch-and-Cut Algorithm

The Zhegalkin identification with rank restriction from Lichtenberg and Eichler (2011) is a Branch-and-Cut algorithm, where the nodes represent possible OTVLs. The algorithm is initialized with the empty OTVL. The children in the next level are all 3^n OTVLs with one line. The following levels are built respectively by adding one TV, that is orthogonal to the parent node, to the OTVL of the parent node while descending in the search tree. This is equivalent to elongate the OTVL of the parent node by one line. The algorithm can be summarized in the following steps

(1) Initialization

(2) Repeat: Define branching node, branch node, cut nodes

(3) End: According to stop criteria

The implemented Branch-and-Cut algorithm uses a best first strategy, therefore the branching node is always the leaf (node without children) with smallest error function and with less than the maximal permitted row number. When branching the branching node, for each TV, that is orthogonal to the OTVL of the branching node, a leaf where this TV is added to it is

generated. For each new node the prediction error is calculated, and when it is clear, that this new branch can not decrease the current global best solution J^+, the node is cut, i.e. deleted from the search tree. The cutting condition hereby is

$$\text{cut node } j \text{ if } \exists\, t \in \{1,...,T\} :\ d_j(t) > \sqrt{\hat{J}^+}. \quad (11)$$

Here $d_j(t)$ is the prediction error of node j at time t and $\hat{J}^+$ is the cost of the current best solution. The cutting condition (11) can be explained by the fact that $y(t) \in \mathbb{U}$ and thus non-negative. Therefore the Zhegalkin polynomial of every TV is non-negative as well. Thus if the modeled value for one time exceeds the measured one by more than the current error, the error can not get smaller if a further TV is added. For more explanations see Lichtenberg and Eichler (2011).

Several stopping criteria exist, like a desired lower threshold of the cost or a maximum number of iterations, can be set manually. If the algorithm stops because no node is branchable anymore, i.e. every leaf has reached the maximal permitted row number, then the optimal $\mathbf{T}^+$ in the restricted search space is found with minimal cost J^+.

3.2.1 Including the Clustering Problem

In general the Branch-and-Cut algorithms runs for each possible cluster, set of genes the considered gene may depend on, separately. However, if the initial lower bound $\hat{J}^+$ for each new cluster is set to the lowest optimal bound J^+ of all previously identified clusters, advantage of this information can be taken: if a cluster with a very good solution has been found, the cutting condition (11) of the following clusters is tightened from the beginning on, i.e. a lot of nodes are cut, leading to reduced calculation effort.

4 CANALYZING FUNCTIONS

Canalyzing functions are a subclass of Boolean functions with the property, that their result is fixed, if one specific input takes a specific value, no matter what values the other inputs take.

Definition 3 (Lichtenberg et al. (2005))**.** A Boolean function f is canalyzing if there exists an $i \in \{1,...,n\}$ and a fixed s, $v \in \{0,1\}$ such that for all $y \in \mathbb{B}^n$ we have $f(y_1,...,y_i,...,y_n) = v$ if $y_i = s$.

The variable y_i is termed as *canalyzing variable*, s as *canalyzing value* and v as *canalyzed value*. If no i can be found, so that the condition above is fulfilled, the function is classified as non-canalyzing. For a canalyzing Boolean function the following holds

Lemma 2. *Given an Boolean function f for n variables that is canalyzing in y_i with canalyzing value s and canalyzed value v, then its complement $\bar{f}$ is canalyzing in y_i with s and $\bar{v}$.*

Proof. The complement of the Boolean function f is defined as $\bar{f} = 1 - f$. Thus if $f(y_1,...,y_i = s,...,y_n) = v$ the complement $\bar{f}$ evaluated for $y_i = s$ is

$$\bar{f}(y_1,...,y_i = s,...,y_n) = 1 - v = \bar{v}.$$

□

4.1 OTVLs of Canalyzing Functions

Whereas expressing canalyzing functions as Zhegalkin polynomials has been considered in Faisal (2008); Faisal et al. (2010), this work is focused on expressing canalyzing in form of OTVLs to be able to restrict the Branch-and-Cut algorithm of Section 3 to only canalyzing functions.

If a Boolean function is canalyzing, for the respective OTVL one of the two following Lemmas holds, depending on the canalyzed value.

Lemma 3. *Given an OTVL $\mathbf{T}$ for n variables and with k lines, then $\mathbf{T}$ is canalyzing in variable y_c with canalyzing value s and canalyzed value $v = 0$ if and only if $t_{jc} = \bar{s}$ for all $j = 1,\ldots,k$.*

Proof. The corresponding Zhegalkin polynomial is calculated by (8). Since $t_{jc} = \bar{s}$ for all $j = 1,\ldots,k$, (8) can be written as

$$p_{\mathbf{T}}(\mathbf{y}) = T(\bar{s},y_c) \sum_{j=1}^{k} \prod_{i=1,i\neq c}^{n} T(t_{ji},y_i). \quad (12)$$

If $y_c = s$, i.e. the canalyzing value is taken, then $T(\bar{s},y_c) = T(\bar{s},s) = 0$, thus $p(\mathbf{y})$ with $y_c = s$ is equal to $v = 0$. □

Lemma 4. *Given an OTVL $\mathbf{T}$ for n variables and with k lines, then $\mathbf{T}$ is canalyzing in variable y_c with canalyzing value s and canalyzed value $v = 1$, if and only if $\mathbf{T}$ includes a TV $\mathbf{t}^c$ defined as $\mathbf{t}^c = [t_1^c,\ldots,t_n^c]$ with $t_c^c = s$ and $t_i^c = -$ for all $i \in \{1,\ldots,n\}\backslash c$.*

Remark 1. To be included in $\mathbf{T}$, the TV $\mathbf{t}^c$ must not be a line of $\mathbf{T}$, but all BVs in $\mathbf{t}^c$ must appear in $\mathbf{T}$, i.e. DIF($\mathbf{t}^c$,$\mathbf{T}$) = {}. The empty TVL corresponds to a Boolean vector with only zeros.

Proof. If $\mathbf{T}$ is canalyzing with $v = 1$ its complement $\bar{\mathbf{T}}$ is canalyzing with $v = 0$, see Lemma 2. According to Lemma 1 the complement $\bar{\mathbf{T}}$ is orthogonal to all TVs in $\mathbf{T}$. Thus there has to be a (0,1)-combination for any pair of rows out of $\mathbf{T}$ and $\bar{\mathbf{T}}$. As proposed $\mathbf{T}$ has to include $\mathbf{t}^c$, where are only '$-$'s in row j except of in the c-th column. To be orthogonal to $\mathbf{t}^c$ in every line

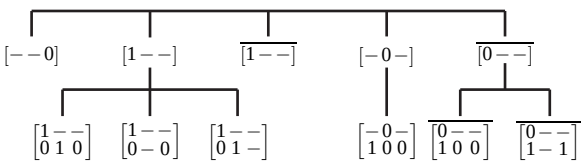

Figure 1: Search tree for Boolean identification restricted to canalyzing functions with $n = 3$ and row number restricted to two.

of the complemented $\overline{\mathbf{T}}$ in the cth-column there has to be the element $\overline{s}$. Thus $\overline{\mathbf{T}}$ is canalyzing with $v = 0$ according to Lemma 3. □

Example 1. ***(continued)*** The Boolean function (1) from the running example is canalyzing with canaylzing variable y_1 as well as y_2, both with canalyzing value '0' and canalyzed value of '1': if y_1 or y_2, respectively, takes the value '0', than the result of the Boolean function is '1', independently of the other variable. This is also obvious from the OTVL representations in (6), which fall in the class of OTVLs described in Lemma 4.

4.2 Zhegalkin Identification with Canalyzing Constraints

In Faisal et al. (2005); Faisal (2008); Faisal et al. (2010); Breindl et al. (2013) it is shown how to express canalyzing functions as Zhegalkin polynomials and integrate those constraints in the Zhegalkin identification. Here it is shown how to restrict the Branch-and-Cut algorithm in Section 3 to canalizing constraints. In addition to its good biological properties another worthwhile advantage of canalyzing functions is their reduced number compared to all Boolean functions, see Table 1. There the number of canalyzing Boolean functions for n variables is compared all existing Boolean functions. A significant decrease of the number of canalyzing functions compared to all Boolean ones is obvious. The adaption introduced here of the identification algorithm takes advantage of that and can considerably reduces the calculation time thereby.

To restrict the Branch-and-Cut algorithm from Lichtenberg and Eichler (2011) to canalyzing functions, only few adaptions are necessary. First instead of initializing the search tree with the empty OTVL as before, it is to initialize with the $2n$ TVs of n variables, which are canalyzing with $v = 1$.

Example 1. For 3 variables, due to Lemma 4 all TVs, which are canalyzing with $v = 1$ are given as

$$\begin{array}{lll} [1--], & [-1-], & [--1], \\ [0--], & [-0-], & [--0], \end{array}$$

where the canalyzing variable of the two TVs in the first columns is the first variable with the canalyzing value 1 and 0, e.g. for the second and third variable.

Due to Lemma 4 any orthogonal TVs can be added to these root-nodes, without loosing the canalyzing property. Furthermore each existing canalyzing function with $v = 1$ (with respect to the maximum line constraint) is in the search space, because by initialization all existing combinations of canalyzing variable and value are covered, and can thus be identified.

To cover also the canalyzing functions with $v = 0$ as additional roots those $2n$ TVs, which are canalyzing with $v = 1$, are taken again, but subtracted from the TV only consisting of '$-$'s, describing the whole Boolean space. Note that the subtraction operation for Zhegalkin polynomials is equivalent to the Difference operation for the corresponding OTVLs. Subtracting a TV of the whole Boolean space is equivalent to building the complement, thus due to Lemma 2 the resulting OTVL is canalyzing with $v = 0$. If one of these root-nodes with $v = 0$ should be branched, then instead of adding all orthogonal TVs, all orthogonal TVs are subtracted. Hereby the canalyzing property with $v = 0$ is preserved. Note that for checking if a TV is orthogonal, it is more efficient to check if it is orthogonal to all TV's that are substracted, then from the difference itself. To distinguish between the OTVLs canalyzing with $v = 1$ and $v = 0$, v is added as further variable to each node. In the branching step, if for the branching node we have $v = 1$, orthogonal TVs have to be added, otherwise subtracted. For the cutting step, the cutting condition also depends on v as follows

$$\text{cut node } j \quad \begin{cases} \text{with } v = 0 & \text{if } \exists t \in \{1,...,T\}: \\ & d_j(t) > \sqrt{J^+}, \\ \text{with } v = 1 & \text{if } \exists t \in \{1,...,T\}: \\ & d_j(t) > -\sqrt{J^+}. \end{cases}$$

5 APPLICATION OF THE CONSTRAINED IDENTIFICATION ALGORITHM

The presented identification algorithm is applied to gene expression data also used in Faisal et al. (2010). The considered gene expression data are measurements of mRNA extracted from mouse liver cells using microarray technology (GeneChip Human Exon 1.0 ST Array). The measurements were repeated four times (T = 3) after 2, 4, 12 and 24 hours. In total the expression levels of 21799 genes could be detected.

Two different mRNA samples were tested, one treated with the contaminant Benzo(a)pyrene (BaP) with a concentration of 5μM and one with a lower one of 50nM, called **T5μM** and **T50nM** in the following. This contaminant BaP is found in cigarette smoke and automobile exhaust and is connected to deadly diseases such as cancer. Geneticists assume that the contamination of cells with BaP with the high concentration of 5μM leads to the cellular process apoptosis, programmed cell death, but not the contamination with the low concentration. Therefore the present gene data is analyzed with regard to apoptosis.

The apoptotic pathway for mice can be found in the KEGG database, Kanehisa and Goto (2000), hosted by Kanehisa Laboratory. From all detected genes, 78 are, due to the database, known to be involved in the apoptotic pathways. These are extracted and considered in the following. The database gives for each gene a set of genes where it may depend on. This knowledge is taken into account for a first identification, where these sets are taken as possible clusters for the identification of the respective gene. Thereby possible solutions of clusters are a priori reduced. The identification with canalyzing constraints as presented in Section 4.2 and without constraints as given in Lichtenberg and Eichler (2011) is applied. The maximum number of rows of the resulting OTVLs is restricted to two. For the identification for each gene a model for connectivity degree two up to the set size given in the database is identified with canalyzing constraints. For the identification without constraints the maximal connectivity degree for each gene is restricted to 5, although for some genes the database give a possible larger cluster, since already for 5 the average calculation time for one possible cluster is with 71 s more than a minute. And if a gene may depend on 11 genes, according to the database, with a connectivity degree of 5 this results in $\binom{11}{5} = 462$ possible clusters, and thus in more than 546 minutes for only one gene. In comparison with canalyzing constraints, one cluster takes 0.022 s for a connectivity degree of 5. For a connectivity degree of 11, the maximum one found in the database, the identification with canalyzing constraints takes 28.66×10^3 s.

A cutout of the identified network is shown in Figure 2 for both concentrations. In general the apoptotic pathways consists of the extrinsic pathway and the intrinsic one. Here the extrinsic one is shown in detail. The expectation, that the concentration of **T5μM** leads to apoptosis, while that of **T50nM** does not, is affirmed here. According to the database the extrinsic pathway is triggered by engagements at the death ligands, which activate *caspase-8*. That induces a signaling cascade, resulting in an activation of *caspase-3*,

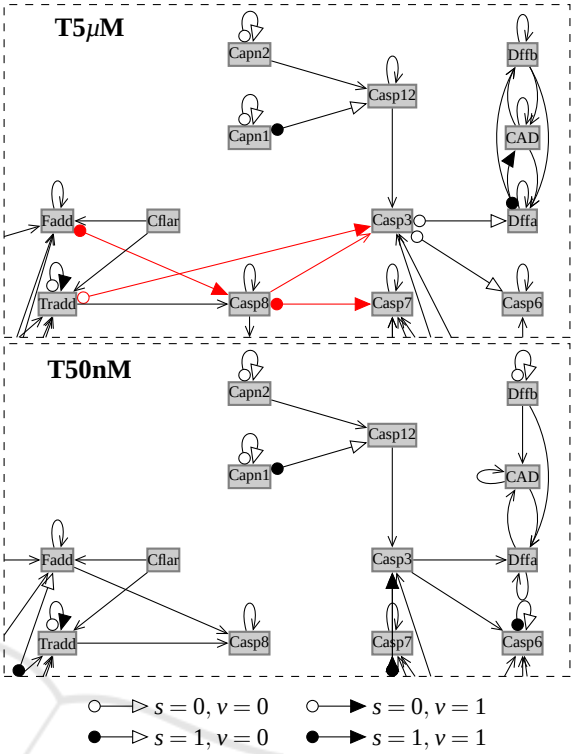

Figure 2: Identified extrinsic pathway for **T5μM** and **T50nM** with given clustering constraints, (canalyzing functions in red canalyzing functions, with no constraints in black, that with minimum error is shown).

what leads to cell death. This can be seen for **T5μM**, where *caspase-8* is activated leading to and activation of *caspase-3*. In Figure 2 the connections of importance here are marked in red. For **T50nM** there is no connection between *caspase-8* and *caspase-3* detected. The arcs with circled tail and triangular head, denote the canalyzing genes, thus the major influencing one. If the tail is colored, its canalyzing value is one, if the head is colored, the canalyzing value is one, and zero otherwise. Thus, for the interconnection from *caspase-8* to *caspase-7* in the network of **T5μM** this, e.g. means that an activation of *caspase-8* always activates *caspase-7*, irrespectively of other genes, whereas a deactivation of *Tradd* always activates *caspase-3*.

An a posteriori analysis of the models identified by the identification, where no canalyzing constraints were imposed, shows that a significant ratio of identified models are canalyzing functions. These ratios of canalyzing functions compared to all identified functions for a certain connectivity degree are shown in Figure 3. For comparison the overall ratios of canalyzing function in all Boolean functions, as calculated from Table 1, are given. It is obvious that expect for the connectivity degree of two the identified models

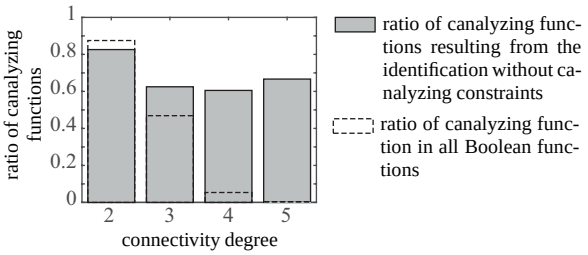

Figure 3: Ratio of canalyzing functions.

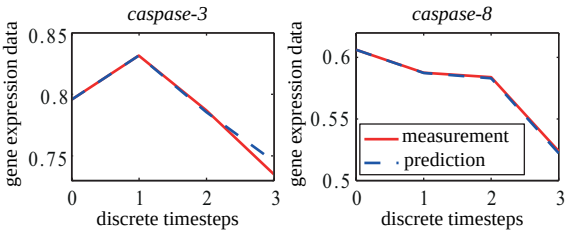

Figure 4: Measured vs. predicted gene expression level dynamics for **T5μM**.

show a significant higher ratio of canalyzing functions than there exists in total. This confirms the choice to restrict the identification to canalyzing functions, not only due to the restricted search space and thus reduced calculation time, but also because genetic models obviously tend to be canalyzing, as also reported by biologists.

The identification is repeated, without considering the dependency sets given by the database, but testing all possible clusters with connectivity degree from two to four with maximum number of rows of the OTVL restricted to two. Note that thus for one gene, $\binom{78}{4}+\binom{78}{3}+\binom{78}{2} = 1505504$ different clusters have to be checked. Here only identification with canalyzing constraints is performed, since without constraints, this is not tractable anymore. For **T5μM** in average an error of 6.87×10^{-5} is achieved, where the root mean square error is taken as error measure. Biologists talk about good approximations if an error $< 10^{-3}$ is achieved. This is not reached for only two out of the 78 genes. Remark that for the identification the maximum number of lines of the identified OTVLs was restricted to two, which is necessary to reduce the solution space and make the problem tractable. This seems to be very small. Nevertheless the very good fit of the identified models confirms that this might be enough. For **T50mM** the average error is with 1.43×10^{-4} slightly larger. This also let suspect, that the high concentration rather lead to apoptosis than the low one. Here only the genes involved in apoptosis are considered, but if other processes are executed, other genes may be involved.

To analyze the continuous gene expression level dynamics, the measurements and the prediction using the identified model are compared. The prediction of gene l, initialized with the measured values $\tilde{\mathbf{y}}(0)$, is determined by

$$y_l(t+1) = p_{\mathbf{T}_l}(\mathbf{y}(t)) \quad \text{with} \quad \mathbf{y}(0) = \tilde{\mathbf{y}}(0)\,.$$

The dynamic of two genes for **T5μM** is shown in Figure 4. Here with *caspase-3* and *caspase-8* , two genes right in the center of the extrinsic pathway are depicted. The prediction fits very well, what is not astonishing since errors of 1×10^{-7} and 5.6×10^{-8} are achieved. For the sample **T50nM** the error of the identified model is with 1.2×10^{-4} and 3.2×10^{-4} almost 10^3-times worse. This also supports the conclusion, that other processes then apoptosis with other genes involved occur for that sample.

6 CONCLUSIONS

The paper presents how to express canalyzing functions in terms of OTVLs. Based on that, it is shown how to restrict the solution space of the Boolean identification algorithm in Lichtenberg and Eichler (2011) to canalyzing functions by simple adaptions mainly in the initialization step. Thereby the restriction to a maximum number of lines, that as a core of the algorithm leads efficiently to a suboptimal solution, does not need to be given up. The advantage of the restriction to canalyzing function is twofold, first from the biological point of view, since canalyzing functions are known to describe gene networks better than other functions, and second from the computational point of view. By the adaption of the Zhegalkin identification algorithm presented in this paper, the search space is enormously reduced by the canalyzing constraints, what leads to managable computation times even for larger data. The presented algorithm has been applied to experimental gene data. By the canalyzing constraints the problem of identification and clustering of 78 genes got tractable and has shown very good fits. Further assumptions of the biologists regarding the network structure of specific processes could be approved by the algorithm presented here.

ACKNOWLEDGEMENTS

The authors acknowledge Saskia Trump (UFZ Leipzig) for her support and experimental data.

REFERENCES

Bochmann, D. and Steinbach, B. (1991). *Logikentwurf mit XBOOLE*. Verlag Technik.

Breindl, C., Chaves, M., and Allgöwer, F. (2013). A linear reformulation of boolean optimization problems and structure identification of gene regulation networks. In *Proc. 52nd IEEE Conf. Decision Control*, pages 733–738.

Faisal, S. (2008). *Discrete-Time Modelling of Gene Networks by Zhegalkin Polynomials*. Ingenieurwissenschaften. Dr. Hut Verlag.

Faisal, S., Lichtenberg, G., Trump, S., and Attinger, S. (2010). Structural properties of continuous representations of boolean functions for gene network modelling. *Automatica*, 46(12):2047–2052.

Faisal, S., Lichtenberg, G., and Werner, H. (2005). A polynomial approach to structural gene dynamics modelling. In *Proc. 16th IFAC World Congr.*, pages 2119–2119.

Faisal, S., Lichtenberg, G., and Werner, H. (2006). Canalizing Zhegalkin polynomials as models for gene expression time series data. In *Proc. 1st Int. Cong. Eng. Intell. Syst.*

Jarrah, A. S., Raposa, B., and Laubenbacher, R. (2007). Nested canalyzing, unate cascade, and polynomial functions. *PhysicaD: Nonlinear Phenomena*, 233(2):167–174.

Kanehisa, M. and Goto, S. (2000). Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, 28(1):27–30.

Kauffman, S. (1993). *The Origins of Order, Self Organization and Selection in Evolution*. Oxford University Press.

Kauffman, S. A., Petersen, C., Samuelsson, B., and Troein, C. (2003). Random boolean network models and the yeast transcriptional network. *PNAS*, 100(25):14796–14799.

Kolda, T. and Bader, B. (2009). Tensor decompositions and applications. *SIAM Review*, 51(3):455–500.

Lichtenberg, G. and Eichler, A. (2011). Multilinear algebraic boolean modelling with tensor decompositions techniques. In *Proc. 18th IFAC World Congr.*, pages 5603–5608.

Lichtenberg, G., Faisal, S., and Werner, H. (2005). Ein Ansatz zur dynamischen Modellierung der Genexpression mit Shegalkin-Polynomen. *Automatisierungstechnik*, 53:589–596.

Lin, P. and Khatri, S. (2013). *Logic Synthesis for Genetic Diseases: Modeling Disease Behavior Using Boolean Networks*. Springer.

Veliz-Cuba, A., Jarrah, A. S., and Laubenbacher., R. (2010). Polynomial algebra of discrete models in systems biology. *Bioinformatics*, 26(13):1637–1643.

Zhegalkin, I. (1928). Arithmetics of symbolic logic. *Mat. Sb.*, 35(3-4):311–377.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS