Logic Integer Programming
Models for Signaling Networks

Utz-Uwe Haus* Kathrin Niermann* Klaus Truemper#
Robert Weismantel*

Abstract

We propose a static and a dynamic approach to model biological signaling networks, and show how each can be used to answer relevant biological questions. For this we use the two different mathematical tools of Propositional Logic and Integer Programming. The power of discrete mathematics for handling qualitative as well as quantitative data has so far not been exploited in Molecular Biology, which is mostly driven by experimental research, relying on first-order or statistical models. The arising logic statements and integer programs are analyzed and can be solved with standard software. For a restricted class of problems the logic models reduce to a polynomial-time solvable satisfiability algorithm. Additionally, a more dynamic model enables enumeration of possible time resolutions in poly-logarithmic time. Computational experiments are included.

Key Words: biological signaling networks, modeling, integer programming, satisfiability, monotone boolean functions

1 Introduction

Cellular decisions are determined by highly complex molecular interactions. In some biological systems like *Saccharomyces* or *E. coli*, detailed measurements of the interacting molecules, including reaction kinetics, have successfully been performed, allowing the construction of quantitative models (Feist et al., 2007). In many other systems such extensive measurements are not available, because of practical experimental restrictions or ethical constraints. However, in these cases, there is often still a sizable amount of qualitative information available, but a lack of suitable predictive modeling tools.

We focus here on interactions in form of signal transduction processes. For such a process we assume that a set of molecules that are important for the biological unit is known. The biological unit reacts to external signals or environmental challenges like stimulation or infection. Typically,
the molecules may be subdivided into input components (e.g. receptors), intermediate components
and output components (e.g. transcription factors): When an external signal arrives, this signal is
processed through the entire unit by first influencing a subset of the input components. Activation
state or presence/absence information is propagated through intermediate nodes towards some of
the output molecules. Based on the assumption that we know the “local” mechanism of activation,
it is our goal to predict the global behavior of the system, identifying the underlying network
structure. For our purposes ‘activation’ can mean any interaction that can be explained biologi-
cally: A protein may be considered activated after phosphorylation, Ca$^{++}$ flux may be detected, or
a component may change its location within the biological unit. Similarly, ‘biological unit’ need
not be restricted to a single cell or compartment, but any collection of components that are to be
considered.

Subsequently we propose a logic and an integer programming model to analyze the static be-
havior of signaling networks. With both approaches one is able to verify the biological modeling,
find potential failure modes and determine suitable intervention strategies. But some phenomena
in biological units like time delays and (negative) feedback loops can not be modeled in this static
fashion. Thus we focus on the dynamics of signaling networks in the second part. We extend the
previous models so that activations can be modeled as occurring at different time points or with
different signaling speed. In this section we present a generic framework for computations with
the dynamic model which can use solvers specific to each of the modeling techniques, logic and
integer programming, as oracles. This ability requires a basic understanding of the transformation
from satisfiability systems to integer programming models and vice versa. Therefore we discuss
both sides in the static as well as in the dynamic context hand in hand. The last section provides
computational tests showing the appropriateness of our techniques.
2 Modeling Logical Interactions

The simplest model of signaling processes is to collect local data in the form of logical formulas, that can be written down in propositional logic \cite{Saez-Rodriguez2007}: Introduce logical variables for each component under consideration, and write down implication formulas for experimentally proven knowledge statements like \textit{“MEK activates ERK”} as

\[ \text{MEK} \rightarrow \text{ERK} \]

and \textit{“In the absence of (activated) pten and ship1 we find that pi3k generates (active) pip3”} as

\[ \neg \text{pten} \land \neg \text{ship1} \land \text{pi3k} \rightarrow \text{pip3} \]

Let the formulas be denoted as \( S_i \) with \( i \in \{1, \ldots, s\} \). We can then identify the formula

\[ S = \bigwedge_{i=1}^s S_i \]

with the model of the biological unit considered: All logical statements \( S_i \) should be valid at the same time to model the global behavior of the unit. We will, as usual, use \( A \rightarrow B \) as abbreviation of \((\neg A) \lor B\), and \( A \leftrightarrow B \) instead of \((A \rightarrow B) \land (B \rightarrow A)\). For the reader unfamiliar with the formalism of propositional logic we refer to \cite{Buning1999}.

We will henceforth assume that all implications in the set \( S_i \) are given in \textit{OR}-form \( \bigvee_{j \in L^A_i} A_j \rightarrow \bigvee_{j \in L^B_i} B_j \), with literals \( A_j, B_j \). Here, \( L^A_i \) is the set of literals appearing on the left in formula \( i \). We will also require that the set \( S \) has been extended with reverse implications by first aggregating formulas with the same right-hand-side, and then adding the reverse implication. Thus, for each set of literals \( R = \{B_j | j \in L^B_i \text{ for some } i \in \{1, \ldots, s\}\} \) appearing on the right-hand-side of the implications indexed by \( I = \{i | S_i = (\bigvee_{j \in L^A_i} A_j \rightarrow \bigvee_{r \in R} r)\} \), we also find the implication \( \bigvee_{i \in I} \bigvee_{j \in L^A_i} A_j \leftarrow \bigvee_{r \in R} r \) in \( S \). This enforces that every activation must have a ‘cause’ within the given model.

The question whether there exists a pattern of activations satisfying all formulas of \( S \) is an instance of the satisfiability (SAT) problem \cite{Truemper2004}. In general, this problem is to ask for a \textit{truth assignment} such that the logical formula in CNF form is \text{TRUE}:

\textbf{Definition 1} (CNF, truth assignment, satisfiable).

1. A \textit{clause} \( \alpha \) is a disjunction of literals, i.e. \( \alpha = A_1 \lor \ldots \lor A_n \) with literals \( A_i \).
2. A formula \( \alpha \) is in \textit{Conjunctive Normal Form} (CNF) if and only if \( \alpha \) is a conjunction of clauses.
3. A formula \( \alpha \) is in \textit{k-Conjunctive Normal Form} (k-CNF) if and only if \( \alpha \) is a conjunction of \( k \)-clauses, i.e. every clause consists of at most \( k \) literals.
4. A \textit{truth assignment} \( \Gamma \) of a propositional formula \( \alpha \) is defined by

\[ \Gamma : \{ \alpha | \alpha \text{ is a propositional formula} \} \rightarrow \{0, 1\}. \]

It can be calculated according to three rules:

(i) \( \Gamma(\neg \alpha) := 1, \text{ iff } \Gamma(\alpha) = 0 \)
(ii) \( \Gamma(\alpha \lor \beta) := 1, \text{ iff } \Gamma(\alpha) = 1 \text{ or } \Gamma(\beta) = 1 \)
(iii) \( \Gamma(\alpha \land \beta) := 1, \text{ iff } \Gamma(\alpha) = 1 \text{ and } \Gamma(\beta) = 1 \)
5. A propositional formula $\alpha$ is **satisfiable** if and only if there exists a truth assignment $\Gamma$, so that $\Gamma(\alpha) = 1$.

**Definition 2.** A model $m$ of a CNF formula $C$ is a satisfying truth assignment of $C$. The set of all models of $C$ are denoted by models($C$). We denote that $m$ assigns 1 (0) to variable $x$ by $m(x) = 1$ ($m(x) = 0$). If $m(x) = 1$ implies $m^*(x) = 1$ for two models $m$ and $m^*$, we say that $m \leq m^*$. If neither $m \leq m^*$ nor $m \geq m^*$ is true, the two models are incomparable. We call a model maximal (minimal) if there is no model $m^*$ such that $m < m^*$ ($m > m^*$). We denote the set of all maximal (minimal) models of $C$ by maximal($C$) (minimal($C$)).

**Problem 1** (SAT). Given a CNF (3-CNF) formula $C$, the SAT (3-SAT) problem is to decide if $C$ is satisfiable and, if so, to return a possible model.

In the setting of SAT problems we can also answer the question whether, given a partial set of activations, there exists a solution for the entire formula $S$, by fixing some logical variables in $S$ to the prescribed values and solving the SAT problem for the remaining formula $S'$. We are thus prepared to introduce the signaling network satisfiability problem (IFFSAT) as:

**Problem 2** (IFFSAT). Let

$$S = \{ S_i = \bigvee_{j \in L_i} A_j \leftrightarrow \bigvee_{j \in L_i^b} B_j \}$$

be a set of $s = |S|$ equivalence formulas over the literal set $L$, and $L_0, L_1 \subseteq L$ two sets of variables to be fixed. An instance of the IFFSAT problem is of the form

$$\bigwedge_{i=1}^{s} S_i \land \bigwedge_{x \in L_0} (\neg x) \land \bigwedge_{x \in L_1} x. \quad \text{(IFFSAT)}$$

Much research has been done to find effective solution algorithms for subclasses of SAT (Truemper, 2004). There is, however, no algorithm that can efficiently (Garey and Johnson, 1979) check satisfiability for arbitrary propositional formulas, as we have to consider for this application:

**Lemma 1.** The satisfiability problem for problems of the form IFFSAT is equivalent to 3-SAT, hence $NP$-complete.

**Proof.** We only need to show that 3-SAT instances can be written in IFFSAT form. Using $\approx$ to designate logical equivalence this can be seen as follows:

$$\bigwedge_{j \in I} x_1^j \lor \bigwedge_{j \in I} x_2^j \lor \bigwedge_{j \in I} x_3^j$$

$$\approx \bigwedge_{j \in I} \left[ (x_1^j \lor x_2^j \leftrightarrow u_1^j) \land (u_1^j \lor x_3^j) \right]$$

$$\approx \bigwedge_{j \in I} (x_1^j \lor x_2^j \leftrightarrow u_1^j) \land \bigwedge_{j \in I} (u_1^j \lor x_3^j \leftrightarrow u_2^j) \land \bigwedge_{j \in I} u_2^j$$

This is an instance of IFFSAT form in which at most 2 $|I|$ literals and 2 $|I|$ propositional formulas were added. □
As already known to Dantzig (Dantzig, 1963), SAT problems can be formulated as integer programs, i.e. feasibility or optimization problems over linear systems of inequalities, where the solutions are required to be integral. For an overview of this field see (Bertsimas and Weismantel, 2005). For the IFFSAT problem the associated integer program (IP) is constructed by introducing \( |L| \) binary variables \( x_l \) and their complements \( \bar{x}_l \), and translating each IFF formula into the system

\[
\begin{align*}
\sum_{j \in L^A_i} x_{A_j} - x_{B_k} & \geq 0 & \text{for all } S_i \in S, k \in L^B_i \\
-x_{A_j} + \sum_{k \in L^B_i} x_{B_k} & \geq 0 & \text{for all } S_i \in S, j \in L^A_i \\
x_l + \bar{x}_l & = 1 & \text{for all } l \in L \\
x_p = 1, x_q = 0 & \text{for all } p \in L_1, q \in L_0
\end{align*}
\]

where we will assume that for non-negated literals \( A \in L \) the variable \( x_A \), and for negated literals \( \neg A \in L \), \( \bar{x}_A \) has been used in the formulation of the inequalities.

**Remark 1.** The integer programming formulation (1) of (IFFSAT) has the form of a generalized set cover problem

\[
Ax \geq 1 - n(A), \quad x \in \{0, 1\},
\]

where \( n(A) \) is the number of negative entries in the corresponding row of \( A \).

We illustrate the presented methods with the help of a small example.

**Example 1.** The inequality description to the network shown in Figure 1 reads

\[
\begin{align*}
(1 - x_A) + (1 - x_B) & \geq (1 - x_D) \quad x_D + x_E \geq x_F \\
(1 - x_A) & \leq (1 - x_D) \quad x_D \leq x_F \\
(1 - x_B) & \leq (1 - x_D) \quad x_E \leq x_F \\
-x_{C} & = x_{E} \quad (1 - x_{G}) = x_{D} \\
-x_{F} & = x_{G} \quad x_{G} = x_{H}
\end{align*}
\]

\( x_l \in \{0, 1\} \ \forall l \)
Several scenarios can be tested with these inequalities. First of all certain input and output patterns can be checked for validity. For this purpose fix $x_A$, $x_B$, $x_C$ and $x_H$ to the desired value and solve the IP with arbitrary objective value. If it is feasible, the input/output pattern is a valid assignment. If one is interested in the output of the network for a prescribed input pattern, one fixes the inputs to interesting values again and solves the IP with the objective to maximize $x_H$. The returned objective value is the value of $H$. In the example the input pattern $x_A = 0, x_B = 1, x_C = 0$ gives and output $x_H = 1$.

Another issue for modeling signaling networks is to check the completeness of the model. This can be done by checking whether the set described by the inequalities (3) contains feasible points. Our example is feasible, as the point $(x_A, x_B, x_C, x_D, x_E, x_F, x_G, x_H) = (1, 1, 0, 1, 0, 0, 0, 0)$ is valid.

Potential failure modes and corresponding suitable intervention strategies can be found by testing knock-in/knock-out scenarios and checking if this forces other variables to obtain a specific value. Knock-in/knock-out scenarios are done by fixing various variables to a desired value. To test whether other variables are thereby fixed, we solve different IPs. Two IPs are needed for checking if two variables have a certain value. In our example we set $x_E = 0$ and $x_H = 1$. To check whether e.g. $x_D$ and $x_A$ need to be fixed, we solve the IP with the objective function $x_D + x_A$. The solution is 1 and the variables are $x_D = 0$ and $x_A = 1$. Thus, we know that $x_D$ must have the value 0. Another optimization problem with the objective to minimize $x_A$ gives the solution 0, and thus $x_A$ is not necessarily 1 but can have both values.
3 Some Complexity Results for IFFSAT

The IFFSAT problem becomes easier if certain structural properties are fulfilled, as has been shown in (Haus et al., 2007). From now on we will restrict the signaling networks to equivalence formulas with only one literal on the right-hand-side, i.e. $|L_B^i| = 1 \forall i$, unless it is explicitly stated differently.

Initially we have to transform IFFSAT to a special form that allows us to perform the subsequent analysis.

**Definition 3** (cascade form). A signaling network (IFFSAT) is called in cascade form if for all clauses $S_i$ it holds that $|L_A^i| \leq 2$ and $|L_B^i| = 1$.

**Remark 2.** Any signaling network can be transformed to cascade form by introducing additional literals and equivalence clauses. Indeed, this can be achieved by recursively applying the following replacement:

$$S_i = \left\{ \left( \bigvee_{j \in J_i} A_j \right) \leftrightarrow B \right\}$$

gets replaced by

$$S'_i = \{A_1 \lor A_2 \leftrightarrow C\}$$

and

$$S''_i = \left\{ \left( \bigvee_{j \in J_i \setminus \{1,2\}} A_j \right) \lor C \leftrightarrow B \right\}$$

where $C$ is a new literal.

Secondly we will review some notation from logic.

**Definition 4.** Let $S$ be a 3-CNF formula. The undirected graph $G(S)$ is defined by the variables of $S$ as its nodes and for every 2-clause of $S$ there is an edge between the corresponding nodes.

**Definition 5** (cutnode, cutnode condition).

1. Let $a$, $b$ and $c$ be nodes of a graph. We call $c$ an $a/b$ cutnode if removing $c$ from the graph disconnects the nodes $a$ and $b$.

2. An IFFSAT instance $S$ in cascade form fulfills the cutnode condition if for every equivalence formula $S_i$, $B_i$ is an $A_1i/A_2i$ cutnode in $G(S)$.

After IFFSAT is transformed to cascade form, the cutnode condition can easily be checked by computing the connected components of $G(S)$. In (Haus et al., 2007) it is proved that

**Theorem 2.** If an instance of IFFSAT in cascade form satisfies the cutnode condition, it can be solved in linear time.

The cutnode condition is a restriction to realistic networks but the following example shows that, e.g., feedback loops do not in general contradict the condition.

**Example 2.** The network defined by

| formula       | 2-clause(s) |
|---------------|-------------|
| $A \lor B \leftrightarrow C$ | $\neg A \lor C, \neg B \lor C$ |
| $A \rightarrow D$ | $\neg A \lor D$ |
| $D \rightarrow E$ | $\neg D \lor E$ |
| $\neg E \rightarrow A$ | $E \lor A$ |
Figure 2: Graph of the network from Example 2

shown in Figure 2 has a cycle with an odd number of negations, but satisfies the cutnode condition (there is one equivalence formula, and its output C is a cutnode).

For the inequality description (1) some nice polyhedral properties can be obtained. One of them is unimodularity (Bertsimas and Weismantel, 2005), which leads to integral relaxations of the polyhedron, for the smallest nonempty IFFSAT problems.

Definition 6 (unimodularity). A matrix $A \in \mathbb{Z}^{m \times n}$ of full row rank is unimodular if the determinant of each basis of $A$ is $\pm 1$.

Lemma 3. The submatrix of a single equivalence clause $S_i$ in the IP model (1) is unimodular if the set $L_A^i$ has cardinality 2 and $L_B^i$ has cardinality 1.

Proof. Consider the case where the formula considered is exactly $A \lor B \leftrightarrow C$ with non-negated atoms $A, B, C$. The matrix $U = \begin{pmatrix} 1 & 1 & -1 \\ -1 & 0 & 1 \\ 0 & -1 & -1 \end{pmatrix}$ is clearly unimodular. Hence, $M = \begin{pmatrix} U & 0 \\ 1 & 1 \end{pmatrix}$, where 1 denotes a $3 \times 3$ unit matrix, is unimodular.

All other cases arise from $M$ by unimodular row operations, i.e. subtracting the complementarity constraint in the top 3 rows. □

Even for the simple formulas $(A \lor B) \leftrightarrow (C \lor D)$ as well as $(A \lor B \lor C) \leftrightarrow D$, the inequality description (1) is non-unimodular. However, the linear relaxation of (1) for one equivalence formula with arbitrary large $L_A^i$ and $L_B^i$ is still integral:

Lemma 4. The inequality description of a single equivalence clause $S_i$ in the IP model (1) is integral.

Proof. See (Hooker, 2007, p. 338). □

These integrality results cannot be generalized to IFFSAT problems with an arbitrary number of equivalence formulas, as the next example shows.

Example 3. We consider a set of two equivalences that model a small negative feedback cycle, i.e.

$$S = \{ (x_1 \lor \bar{x}_3 \leftrightarrow x_4) \land (x_2 \lor x_4 \leftrightarrow x_3) \}.$$

The integer programming formulation from (1) has the following LP relaxation:
\[
x_1 + \bar{x}_3 - x_4 \geq 0 \quad x_2 - x_3 + x_4 \geq 0 \\
-x_1 + x_4 \geq 0 \quad -x_2 + x_3 \geq 0 \\
-\bar{x}_3 + x_4 \geq 0 \quad +x_3 - x_4 \geq 0 \\
x_i + \bar{x}_i = 1, \quad 0 \leq x_i \leq 1.
\]

Computing the vertices we find both integral and fractional points:

\[
\begin{array}{cccccc}
x_1 & x_2 & x_3 & \bar{x}_3 & x_4 \\
0 & 0 & 1/2 & 1/2 & 1/2 \\
0 & 1/2 & 1/2 & 1/2 & 1/2 \\
1/2 & 0 & 1/2 & 1/2 & 1/2 \\
1/2 & 1/2 & 1/2 & 1/2 & 1/2 \\
0 & 1 & 1 & 0 & 0 \\
1 & 0 & 1 & 0 & 1 \\
1 & 1 & 1 & 0 & 1 \\
\end{array}
\]

It should be noted that not all vertices are fractional, which means that this SAT instance is not infeasible.
4 Dynamics of Signaling Networks

In practical applications signaling networks often turn out to be infeasible. This is not reasonable in a biological sense, but it can occur due to delayed reactions modeled as instantaneous, or due to modeling errors. Especially (negative) feedback loops with time delays make the static models infeasible, but at the same time have a huge impact on the functionality of a signaling network since certain activation cascades can be enabled initially and switched off at a later time point to avoid overreaction. In order to model the dynamics of a signaling network we introduce an extension of IFFSAT, the requirement IFFSAT problem.

Problem 3 (RIFFSAT). Let a set $R$ of $r = |R|$ equivalence formulas of the form

$$R = \left\{ R_i = \bigvee_{j \in L_i^A} (A_j \land y_j) \leftrightarrow (B_i \land \bigvee_{j \in L_i^A} y_j) \right\}$$

over the literal set $L \cup Y$ be given, where $Y$ is the set of requirement variables. Let $L_0, L_1 \subseteq L \cup Y$, sets of fixings, be given, then the RIFFSAT problem is to find a satisfying solution of

$$\bigwedge_{i=1}^r R_i \land \bigwedge_{x \in L_0} (\neg x) \land \bigwedge_{x \in L_1} x.$$  

(4)  

(RIFFSAT)

In the signaling network context $y_j = 1$ denotes influence of the corresponding component $A_j$ on the right hand side while $y_j = 0$ denotes no effect. In case $y_j = 0$ for all $j \in L_i^A$, we request $B_i$ to be free.

Remark 3. We ask the requirement variable $y_j$ of $A_j$ to be the different for every IFF formula in which $A_j$ occurs on the left hand side, and to be equal to $\bigvee_{k \in L_i^A} y_k$ if $A_j$ is the right hand side of clause $R_i$.

We denote by $\mathcal{F}$ the set of all $0/1$-points for which (RIFFSAT) is True, i.e.

$$\mathcal{F} = \left\{ x \in \{0,1\}^{2|L|} \mid \Gamma(\bigwedge_{i=1}^r R_i \land \bigwedge_{x \in L_0} (\neg x) \land \bigwedge_{x \in L_1} x) = 1 \right\}.$$  

Usually one is interested in solving (RIFFSAT) with special properties on the set of requirement variables $y_j$, like a maximal or minimal models over $y$. Such a problem can be solved by a variation of SAT, namely MAXVAR SAT (see Truemper [2004]). Here a satisfiable CNF system $C$ and a set $T$ with True/False fixings of a variable subset is given, such that $C$ is not satisfiable if all variables are fixed according to $T$. The task is to determine a maximal subset $T^*$ of $T$ so that $C$ is satisfiable. In our setting $T$ can be $\{y_1 = \ldots = y_{|L|} = \text{True}\}$. MAXCLS SAT is also a related problem. We remark that both problems are special cases of MAXSAT, which can not be approximated polynomially better than $8/7$ (Håstad [2001]) and hence it is $\mathcal{NP}$-complete.

Besides the presented method from logic one can also find a maximal solution with the help of integer programming techniques.  

Lemma 5 (inequality description for RIFFSAT). Given one equivalence formula $R_i \in R$ as in (4), introduce $n_i = |L_i^A|$ additional binary variables $Y_i = \{x_{y_1}, \ldots, x_{y_{n_i}}\}$. Then the corresponding set of feasible points $\mathcal{F}_{R_i}$ can be described by $n_i + n_i \cdot 2^{n_i-1}$ inequalities plus binary constraints of the form:

$$\begin{align*}
    x_{B_i} &\geq x_{A_j} - (1 - x_{y_j}) & j \in L_i^A \\
    x_{B_i} &\leq \sum_{k \in S} x_{A_k} + (1 - x_{y_j}) + \sum_{k \in S} x_{y_k} & j \in S, \ 0 \neq S \subseteq L_i^A \\
    x_i &\in \{0,1\} & l \in L_i^A \cup \{B_i\} \cup Y_i \\
    1 &= x_i + \bar{x}_i &
\end{align*}$$

(5)
We will assume that $\bar{x}_j$ is used in the inequality description if the corresponding atom is negated. To derive the inequality description for RIFFSAT, introduce such inequalities for all $R_i$, $i = 1, \ldots, r$, and fix the variables according to $L_0$ and $L_1$.

Remark 4. Note that this formulation still preserves the form of a generalized set covering problem. In addition, for each fixed $x_j$, the formulation reduces to an instance of (1).

Maximizing the sum over all requirement variables $x_{y_j}$ yields one maximal feasible solution. However, there are many inequalities needed to describe the feasible points. It turns out that all of them are needed:

Lemma 6. The inequalities for a single relaxed equivalence clause (5) are facets for the convex hull of their integral points $F$.

Proof. We will show that for every inequality there exist $\dim(\text{conv}(F)) = 2n+1$ affine independent, integral points fulfilling the corresponding inequality with equality, where $n = |L_A|$. $x_{B_i} \geq x_{A_j} - (1 - y_j)$: For reasons of symmetry we restrict the analysis to $j = 1$. The $2n + 1$ linearly independent points are displayed in Table 1.

\begin{table}[h]
\centering
\begin{tabular}{cccccccccc}
\hline
$x_{A_1}$ & $x_{A_2}$ & $x_{A_3}$ & $\cdots$ & $x_{A_n}$ & $x_{B_i}$ & $y_1$ & $y_2$ & $y_3$ & $\cdots$ & $y_n$ \\
\hline
$n$ points exploiting all free $y_j$: \\
1 & 0 & 0 & $\cdots$ & 0 & 1 & 1 & 1 & 1 & $\cdots$ & 1 \\
1 & 0 & 0 & $\cdots$ & 0 & 1 & 1 & 0 & 1 & $\cdots$ & 1 \\
$\vdots$ & $\vdots$ & $\vdots$ & $\ddots$ & $\vdots$ & $\vdots$ & $\vdots$ & $\vdots$ & $\vdots$ & $\ddots$ & $\vdots$ \\
1 & 0 & 0 & $\cdots$ & 0 & 1 & 1 & 1 & 1 & $\cdots$ & 0 \\
\hline
$n-1$ points exploiting all free $x_{A_j}$: \\
1 & 1 & 0 & $\cdots$ & 0 & 1 & 1 & 1 & 1 & $\cdots$ & 1 \\
1 & 0 & 1 & $\cdots$ & 0 & 1 & 1 & 1 & 1 & $\cdots$ & 1 \\
$\vdots$ & $\vdots$ & $\vdots$ & $\ddots$ & $\vdots$ & $\vdots$ & $\vdots$ & $\vdots$ & $\vdots$ & $\ddots$ & $\vdots$ \\
1 & 0 & 0 & $\cdots$ & 1 & 1 & 1 & 1 & 1 & $\cdots$ & 1 \\
\hline
two possible points with $x_{B_i} = 0$: \\
1 & 0 & 0 & $\cdots$ & 0 & 0 & 0 & 1 & 1 & $\cdots$ & 1 \\
1 & 1 & 0 & $\cdots$ & 0 & 0 & 0 & 1 & 1 & $\cdots$ & 1 \\
\hline
\end{tabular}
\caption{2n + 1 linearly independent points for $x_{B_i} \geq x_{A_1} - y_1$.}
\end{table}

This concludes the proof.

Lemma 6 thus implies that the inequalities that are needed to describe $\text{conv}(F)$ definitely include all constraints in (5). Hence, the number of inequalities is exponential in the number of inputs of the clause. This motivates to explore alternative, extended formulations for modeling $F$ based on a cascade representation of their underlying IFFSAT instance.

Lemma 7. Each RIFFSAT instance can be transformed to cascade form.
$n - k + 1$ points exploiting all free $x_{A_j}$:

| $x_{A_1}$ | $x_{A_2}$ | ⋮ | $x_{A_k}$ | $x_{A_{k+1}}$ | ⋮ | $x_{A_n}$ | $x_{B_1}$ | $y_1$ | $y_2$ | ⋮ | $y_k$ | $y_{k+1}$ | $y_{k+2}$ | ⋮ | $y_n$ |
|----------|----------|---|----------|----------|---|----------|----------|---|-----|---|---|----------|----------|---|-----|---|
| 0        | 0        | ⋮ | 0        | 0        | ⋮ | 0    | 0        | 1    | 1    | ⋮ | 1    | 0        | 0    | ⋮ | 0    |
| 0        | 0        | ⋮ | 0        | 1        | 0        | ⋮ | 0    | 0    | 1    | 1    | ⋮ | 1    | 0        | 0    | ⋮ | 0    |
| 0        | 0        | ⋮ | 0        | 0        | 0        | ⋮ | 0    | 0    | 1    | 1    | ⋮ | 1    | 0        | 0    | ⋮ | 0    |
| ⋮        | ⋮        | ⋮ | ⋮        | ⋮        | ⋮        | ⋮ | ⋮    | ⋮    | ⋮    | ⋮    | ⋮ | ⋮        | ⋮    | ⋮ | ⋮    |
| 0        | 0        | ⋮ | 0        | 0        | 0        | ⋮ | 1    | 0    | 1    | 1    | ⋮ | 1    | 0        | 0    | ⋮ | 0    |

$k - 1$ points exploiting all free $y_j$:

| $x_{A_1}$ | $x_{A_2}$ | ⋮ | $x_{A_k}$ | $x_{A_{k+1}}$ | ⋮ | $x_{A_n}$ | $x_{B_1}$ | $y_1$ | $y_2$ | ⋮ | $y_k$ | $y_{k+1}$ | $y_{k+2}$ | ⋮ | $y_n$ |
|----------|----------|---|----------|----------|---|----------|----------|---|-----|---|---|----------|----------|---|-----|---|
| 0        | 0        | ⋮ | 0        | 0        | ⋮ | 0    | 0        | 1    | 0    | ⋮ | 1    | 0        | 0    | ⋮ | 0    |
| ⋮        | ⋮        | ⋮ | ⋮        | ⋮        | ⋮ | ⋮    | ⋮    | ⋮    | ⋮    | ⋮ | ⋮        | ⋮    | ⋮ | ⋮    |
| 0        | 0        | ⋮ | 0        | 0        | 0        | ⋮ | 0    | 0    | 1    | 1    | ⋮ | 0    | 0        | 0    | ⋮ | 0    |
| 0        | 0        | ⋮ | 0        | 0        | 0        | ⋮ | 1    | 0    | 1    | 1    | ⋮ | 1    | 0        | 0    | ⋮ | 0    |
| ⋮        | ⋮        | ⋮ | ⋮        | ⋮        | ⋮ | ⋮    | ⋮    | ⋮    | ⋮    | ⋮ | ⋮        | ⋮    | ⋮ | ⋮    |
| 0        | 0        | ⋮ | 1        | 0        | 0        | ⋮ | 0    | 1    | 1    | 1    | ⋮ | 1    | 0        | 0    | ⋮ | 0    |

$k$ points with one $x_{A_j} = 1$, $j \in S$ each:

| $x_{A_1}$ | $x_{A_2}$ | ⋮ | $x_{A_k}$ | $x_{A_{k+1}}$ | ⋮ | $x_{A_n}$ | $x_{B_1}$ | $y_1$ | $y_2$ | ⋮ | $y_k$ | $y_{k+1}$ | $y_{k+2}$ | ⋮ | $y_n$ |
|----------|----------|---|----------|----------|---|----------|----------|---|-----|---|---|----------|----------|---|-----|---|
| 1        | 0        | ⋮ | 0        | 0        | ⋮ | 0    | 1        | 1    | 1    | ⋮ | 1    | 0        | 0    | ⋮ | 0    |
| 0        | 1        | ⋮ | 0        | 0        | ⋮ | 0    | 1        | 1    | 1    | ⋮ | 1    | 0        | 0    | ⋮ | 0    |
| ⋮        | ⋮        | ⋮ | ⋮        | ⋮        | ⋮ | ⋮    | ⋮    | ⋮    | ⋮    | ⋮ | ⋮        | ⋮    | ⋮ | ⋮    |
| 0        | 0        | ⋮ | 1        | 0        | 0        | ⋮ | 0    | 1    | 1    | 1    | ⋮ | 1    | 0        | 0    | ⋮ | 0    |

$n - k$ points with one $y_j = 1$, $j \in L_A \setminus S$ each:

| $x_{A_1}$ | $x_{A_2}$ | ⋮ | $x_{A_k}$ | $x_{A_{k+1}}$ | ⋮ | $x_{A_n}$ | $x_{B_1}$ | $y_1$ | $y_2$ | ⋮ | $y_k$ | $y_{k+1}$ | $y_{k+2}$ | ⋮ | $y_n$ |
|----------|----------|---|----------|----------|---|----------|----------|---|-----|---|---|----------|----------|---|-----|---|
| 0        | 0        | ⋮ | 0        | 1        | 1        | ⋮ | 1    | 1    | 1    | ⋮ | 1    | 1        | 0    | ⋮ | 0    |
| 0        | 0        | ⋮ | 0        | 1        | 1        | ⋮ | 1    | 1    | 1    | ⋮ | 1    | 1        | 1    | ⋮ | 0    |
| ⋮        | ⋮        | ⋮ | ⋮        | ⋮        | ⋮ | ⋮    | ⋮    | ⋮    | ⋮    | ⋮ | ⋮        | ⋮    | ⋮ | ⋮    |
| 0        | 0        | ⋮ | 0        | 1        | 1        | ⋮ | 1    | 1    | 1    | ⋮ | 1    | 1        | 1    | ⋮ | 0    |

one possible point with $y_1 = 0$:

| $x_{A_1}$ | $x_{A_2}$ | ⋮ | $x_{A_k}$ | $x_{A_{k+1}}$ | ⋮ | $x_{A_n}$ | $x_{B_1}$ | $y_1$ | $y_2$ | ⋮ | $y_k$ | $y_{k+1}$ | $y_{k+2}$ | ⋮ | $y_n$ |
|----------|----------|---|----------|----------|---|----------|----------|---|-----|---|---|----------|----------|---|-----|---|
| 0        | 0        | ⋮ | 0        | 0        | ⋮ | 0    | 1        | 0    | 1    | ⋮ | 1    | 0        | 0    | ⋮ | 0    |

Table 2: $2n+1$ linearly independent points for $x_{B_i} \leq \sum_{j \in S} x_{A_j} + (1-y_1) + \sum_{j \in S} y_j$, with $S = \{1, \ldots, k\}$. 

12
The recursive construction is analogous to the IFFSAT case, handling each conjunction as one literal and introducing variables of y-type for the artificial variables.

Although transforming the RIFFSAT instance to this specific form leads to more variables and equivalence formulas, the growth of the instance is only quadratic. Recall that \( r = |R| \) and let \( n \) be the largest number of inputs of all clauses. Then, due to the transformation, the number of literals and clauses in the RIFFSAT instance increases by at most \( 2r(n - 2) \) and \( r(n - 2) \), respectively. In contrast to that there are \( 6(r + r(n - 2)) \) inequalities needed to encode the signaling network in cascade form compared to \( (n + n \cdot 2^{n-1})r \) inequalities in the IP representation of Lemma 6. Both counts leave out binary constraints and fixings. Hence, it is a reasonable tool to reduce the complexity of the inequality description.

So far both approaches, MAXVAR SAT as well as IP, provide one maximal feasible solution. But one is interested in all maximal solutions with respect to the requirement variables to find suitable intervention strategies or to plan specific experiments verifying an actual network structure. In order to find the set of all maximal models (see Definition \( \Box \)) or all maximal feasible solutions with respect to the y-variables it is necessary to use a ‘clever enumeration method’. For this purpose we want to make use of the joint generation algorithm (Fredman and Khachiyan 1996). This method checks whether a pair of monotone boolean functions are dual. The problem is equivalent to finding all maximal models of a monotone CNF formula.

**Definition 7** (monotone CNF). Let \( C \) be a CNF expression and let \( Z \) be a subset of the literals \( L \). Then \( C \) is called **down-monotone** (**up-monotone**) in \( Z \) if from \( m^*(Z, L \setminus Z) \) satisfying \( C \) it follows that \( m(Z, L \setminus Z) \) satisfies \( C \) for all \( m(Z) \leq m^*(Z) \) \((m(Z) \geq m^*(Z))\) and \( m(L \setminus Z) = m^*(L \setminus Z) \).

An integral set is down-monotone in the vector \( z \) if \((x, z^*) \in P \) leads to \((x, z) \in P \) for all \( z \leq z^* \).

**Remark 5.** A CNF formula is down-monotone in \( x \) if all literals of \( x \) have only negative occurrence.

Note that RIFFSAT is not necessarily down-monotone in \( y \) as the next example shows:

**Example 4.** We consider the easy RIFFSAT instance

\[
Z = (A \land y_A) \lor (B \land y_B) \leftrightarrow C \land (y_A \lor y_B).
\]

It is not down-monotone in \( Y = \{y_A, y_B\} \) since the truth assignment

\[
A = 1 \quad B = 0 \quad y_A = 1 \quad y_B = 1 \quad C = 1
\]

satisfies \( Z \), while

\[
A = 1 \quad B = 0 \quad y_A = 0 \quad y_B = 1 \quad C = 1
\]

does not.

As RIFFSAT instances are non-monotone in general, we apply a transformation proposed by (Kavvadias et al., 2000) to monotonize the requirement variables of non-monotone CNF terms. The method preserves the set of maximal models while the general models can differ. Therefore the positive \( y \)-variables are eliminated according to a recursive resolution procedure:

1. Expand the RIFFSAT instance \( K \) to CNF form.
2. Choose a variable \( y_k \) that occurs as positive and negative literal in \( K \).
3. Divide the clauses into three parts \( S_y \cup S_{\overline{y}} \cup A_{\overline{x}} \), the set of clauses with occurrence of \( y_k \), of \( \overline{y}_k \), and no occurrence at all. In this context ‘\( \cup \)’ denotes a conjunction. We write \( K_i = (C_i \lor y_k) \) for \( K_i \in S_y, i = 1, \ldots, |S_y| \), and \( K_j = (D_j \lor \overline{y}_k) \) for \( K_j \in S_{\overline{y}}, j = 1, \ldots, |S_{\overline{y}}| \). Thus, \( C_i \) and \( D_j \) are disjunctions of all other variables apart from \( y_k \).

4. Compute all \textit{resolvents} \( R_{y_k} \) of each pair of clauses in \( S_y \) and \( S_{\overline{y}} \) with respect to \( y_k \), i.e., \( R_{y_k} = \{(C_i \lor D_j) \mid K_i \in S_y, K_j \in S_{\overline{y}}\} \).

5. The expression \( K_{y_k} = R_{y_k} \cup S_{\overline{y}} \cup A_{\overline{x}} \) is monotone in \( y_k \), since \( y_k \) has only negative occurrences.

\textbf{Theorem 8} (Kavvadias et al. 2000). With the above transformation it holds that \( \text{models}(K) \subseteq \text{models}(K_{y_k}) \) and \( \text{maximal}(K) = \text{maximal}(K_{y_k}) \).

Thus, we can apply this concept recursively and obtain a CNF expression which is monotone in the \( y \)-variables and hence, we can use the joint generation algorithm (Fredman and Khachiyan, 1996) to compute the invariant maximal models.

However, it is easy to see that the transformation can lead to an exponentially larger expression. But in the case of RIFFSAT instances the number of clauses even decreases.

Monotonizing the requirement variables in one \( R_i \in R \), formula \((4)\) reduces the clauses to

\[
\bigwedge_{j \in L^I_1} \left( (\neg A_j \lor \neg y_j \lor B_j) \land \bigvee_{k \in L^A} A_k \lor (\neg y_j \lor \neg B_i) \right)
\approx \left( \bigvee_{j \in L^I_1} (A_j \land y_j) \rightarrow B_j \right) \land \left( \bigvee_{j \in L^I_1} (C \land y_j) \rightarrow \bigvee_{k \in L^A} A_k \right).
\]

(6)

In terms of inequalities this is modeled by

\[
\begin{align*}
x_{R_i} &\geq x_{A_j} - (1 - x_{y_j}) \\
x_{B_i} &\leq \sum_{k \in L^A} x_{A_k} + (1 - x_{y_j}) \\
x_{l} &\in \{0, 1\} \\
1 &= x_l + \overline{x}_l \\
i &\in L^I_1 \cup \{B_i\} \cup Y_i.
\end{align*}
\]

(7)

Introducing such constraints for every \( R_i \in R \) gives the monotonized version of \( \text{RIFFSAT} \). We denote by \( F_{\text{mon}} \) the set of integral points fulfilling \((6)\) and thus also \((7)\) for each \( i = 1, \ldots, r \).

\textbf{Lemma 9.} The system \((6)\), and thus \((7)\), are monotone in \( y \) and have the same maximal \( y \)-solutions as \( \text{RIFFSAT} \) and \((5)\).

\textbf{Proof.} From the inequalities it is easy to see that it is down-monotone in \( y \), since pushing one \( y \) component to 0, say \( x_{y_j} = 0 \), only relaxes both inequalities containing \( x_{y_j} \). In particular, this means that the same \( A \) and \( B \) components are feasible for \( x_{y_j} = 0 \) as for \( x_{y_j} = 1 \). In the logic formula one can see the down-monotonicity in \( y \) by expanding the implications to CNF. All \( y \)-variables occur as negated atoms.

To prove the accordance of the maximal solutions we can focus on a single RIFFSAT formula \( \bigvee_{j=1}^{n} (A_j \land y_i) \leftrightarrow B \land \bigvee_{i=1}^{n} y_i \), \( I = \{1, \ldots, n\} \), because the \( y \)-variables are disjoint for every RIFFSAT formula and thus the monotonized version of RIFFSAT can be built by monotonizing each formula \( R_i \).
We will show that the following two integer programs (V) and (W) have the same optimal solutions with respect to the y components.

\[
\begin{align*}
\text{max} & \quad \sum_{i=1}^{n} x_{yi} \\
\text{s.t.} & \quad x_B \geq x_{Ai} - (1 - x_{yi}) \quad j \in I \\
& \quad x_B \leq \sum_{k \in S} x_{Ak} + (1 - x_{yi}) + \sum_{k \notin S} x_{yk} \quad j \in S, \; 0 \neq S \subseteq I \\
& \quad x_j \in \{0, 1\} \\
& \quad 1 = x_l + \bar{x}_l \quad l \in \{A_I, y_I, B\}
\end{align*}
\]

and

\[
\begin{align*}
\text{max} & \quad \sum_{i=1}^{n} x_{yi} \\
\text{s.t.} & \quad x_B \geq x_{Ai} - (1 - x_{yi}) \quad j \in I \\
& \quad x_B \leq \sum_{k=1}^{n} x_{Ak} + (1 - x_{yi}) \quad j \in I \\
& \quad x_I \in \{0, 1\} \\
& \quad 1 = x_l + \bar{x}_l \quad l \in \{A_I, y_I, B\}
\end{align*}
\]

For an ease of notation we will subsequently use the index l for the variable x_l. Let A_l and y_l denote (A_1, \ldots, A_n) and (y_1, \ldots, y_n). Let \mathcal{F}_V and \mathcal{F}_W be the set of integral points of problem (V) and (W), respectively, and let x^{*L} indicate the optimal solution of problem (L), L=V,W. Then, y^{*L} denotes the corresponding y components. Note that \mathcal{F}_V \subset \mathcal{F}_W.

**Case 1:** If there are no A_l, B components fixed to a certain value, there exists an x^{*L} \in \mathcal{F}_L with y^{*L} = 1. For example, the vector (A_I, y_I, B) = (0, 1, 0) is feasible for both IPs. Thus the optimal solutions are equal.

**Case 2:** If some A_l, B components are fixed to 0 or 1, there are two critical cases in which there exists no x^{*L} \in \mathcal{F}_L such that y^{*L} = 1 for L=V,W.

The first one is A_l = 0 and B = 1. Here, y^{*L} = 0 is necessary for both L since the second type of inequalities of (W), which are also valid for (V), forces B immediately to 0 if one y_j = 1. Thus, the optimal values match.

The second critical configuration is that A_l = 1 for i \in I_l \subseteq I with I_l \neq \emptyset and A_j = 0, j \notin I_l, but B = 0. The optimal y solution is y^{*L} = 0 for i \in I_l and y^{*L} = 1, j \notin I_l, L=V,W, by the first type of inequalities, which is valid for both programs. It forbids to lift another y_l to 1 as this switches B also to 1.

If x_{yi} variables are fixed to 0 or 1, it influences both objective values equally. Thus, the objective value of the integer programs (V) and (W) coincide in every possible case which concludes the proof.

\[\square\]

**Lemma 10.** The inequality system (7) describes \text{conv}(\mathcal{F}_{mon}) when substituting \{0, 1\} by \{0, 1\}.

A proof for the Lemma can be found in the Appendix. Note that the system (7) are even a facet description for \text{conv}(\mathcal{F}_{mon}) as the inequalities (5) are facets for \text{conv}(\mathcal{F}) and all feasible points are preserved.

The LP relaxation of a monotonized dynamic IP (7) containing more than one formula, i.e. r \geq 2, is not integral in general as the following example shows.

**Example 5.** Consider the monotonized version of the dynamic IP (7) of the signaling network from
Figure 3: A signaling network with two negative feedback loops from Example 6. The dashed line denotes inhibition.

Example 3. Its linear relaxation is the following polyhedron

\[
\begin{align*}
  x_1 - (1 - y_1) &\leq x_4 & x_2 - (1 - y_2) &\leq x_3 \\
  (1 - x_3) - (1 - y_3) &\leq x_4 & x_4 - (1 - y_4) &\leq x_3 \\
  x_1 + (1 - x_3) + (1 - y_1) &\geq x_4 & x_2 + x_4 + (1 - y_2) &\geq x_3 \\
  x_1 + (1 - x_3) + (1 - y_3) &\geq x_4 & x_2 + x_4 + (1 - y_4) &\geq x_3 \\
  0 &\leq x_i, y_i &\leq 1 &\text{for } i = 1, \ldots, 4
\end{align*}
\]

This polyhedron is not integral, since e.g. \((x,y) = (1/2, 1, 1/2, 1/2, 1, 1/2, 1, 1)\) is a vertex. But it also has integral vertices like \((0, 0, 1, 0, 1, 0, 1, 0)\) so that it is not infeasible.

To illustrate the use of the dynamic model we give a small example of a signaling network.

Example 6. Consider the network defined by

\[
\begin{align*}
  x_C &\geq x_A - (1 - y_A) & x_A &\geq x_E - (1 - y_E) \\
  x_C &\leq x_A + (1 - y_A) & x_A &\leq x_E + (1 - y_E) \\
  x_C &\geq x_B - (1 - y_B) & x_F &\geq x_D - (1 - y_D) \\
  x_C &\leq x_B + (1 - y_B) & x_F &\leq x_D + (1 - y_D) \\
  x_D &\geq x_C - (1 - y_C) & x_G &\geq (1 - x_F) - (1 - y_F) \\
  x_D &\leq x_C + (1 - y_C) & x_G &\leq (1 - x_F) + (1 - y_F) \\
  x_E &\geq (1 - x_D) - (1 - y_D) & x_B &\geq x_G - (1 - y_G) \\
  x_E &\leq (1 - x_D) + (1 - y_D) & x_B &\leq x_G + (1 - y_G) \\
  x_i, y_i &\in \{0, 1\} &\forall i
\end{align*}
\]

which is illustrated in Figure 3. As there are two negative feedbacks involved, using the static approach leads to infeasibility. Thus, we are interested in the dynamics like possibly late reactions, i.e. in maximal feasible subnetworks. In this case there are different possibilities to disturb the cycles. One can either set \(y_C = 0\) and the rest to 1 or cut two disjoint arcs each in one cycle, e.g. \(y_E = 0\) and \(y_G = 0\). The remaining variables can then be 1. With this guideline and precise experiments the practitioner can then identify how the network structure actually looks like. Of course, the obtained \(y\) values also give hints at suitable intervention strategies.
Remark 6. Note that one can also model timing information using the RIFFSAT formulation: Let $T \in \mathbb{Z}_+$, a time horizon, be given. Then, for every $t \in \{1, \ldots, T\}$ we have different signaling networks, because of distinct reaction times. Due to these delays it is possible for a molecule to be absent at one point $t_1$, but to be present at $t_2$. This can be modeled by $T$ copies of the same networks but each with a different $y^t$ vectors $t = 1, \ldots, T$, which imply absence or presence of each interdependence in the biological unit at time point $t$. Each $y^t$ vector is according to Equation (5). In this setting the changes of the signaling network over time is encoded by the difference of two consecutive $y^t$ vectors, $y^t - y^{t-1}$, $t \geq 2$.

Our proposed approach is also able to handle extra information about the structure of the network over time. Such information can be statements like if an interaction is present at time point $t^*$, it is present for all $t \geq t^*$, or an interaction is only valid for exactly one $t \in \{1, \ldots, T\}$. It can be expressed in terms of logical formulas/inequalities over the $y^t$. 
| cols | rows | inputs | outputs | #feas | #infeas | total time (s) | avg time (s) |
|------|------|--------|---------|-------|---------|---------------|-------------|
| 214  | 376  | 3      | 14      | 36    | 131036  | ≈ 120         | 0.001       |

Table 3: In/Out fixing of TCR from Saez-Rodriguez et al. (2007).

5 Computations

We have tested the ideas presented in this paper on the TCR-signaling network from Saez-Rodriguez et al. (2007), as well as several randomly generated signaling networks. To demonstrate that the integer programming formulation is in fact useful to test feasibility of scenarios (on the full model or in knock-in/knock-out tests), we performed several tests: Table 3 shows feasibility tests for all possible combinations of input/output values in the TCR model. Computations were performed using CPLEX 9.1 and Allegro Common Lisp 8.1 on a SUN-Fire-V890 with 1.2 GHz. Even though the IPs do not generally reduce to an LP description, the whole instances could be solved within two minutes.

Due to the lack of real instances we generated three types of random signaling networks which are built with different probability distributions so that each has a distinct structure. For technical reasons we deviate slightly from the standard form of a signaling network: we construct IFFormulas of the form \( \bigvee_{j \in J} A_j \rightarrow B \) (OR-clauses) and \( \bigwedge_{j \in J} A_j \rightarrow B \) (AND-clauses). All types are constructed according to the following procedure:

1. Set the number of components \( n \) and operation nodes \( m \) and fix the proportion of AND- and OR-nodes \( A/O \). In our random types \( A/O \) will always be equal to 0.5.

2. Determine, with respect to \( A/O \), which operation nodes are AND- and which are OR-nodes. Therefore the realization of a random variable \( X \) which probability distribution reflects the ratio \( A/O \) is generated. Clearly this means, \( P(X = \text{AND}) = A/O = 1 - P(X = \text{OR}) \).

3. List all components from 1 to \( n \).

4. Let \( a_i \) be the number of inputs of operation node \( i = 1, \ldots, m \). Determine \( a_i \) as a realization of \( A_i \) with \( A_i \sim R \) with some probability distribution \( R \). Note that \( R \) changes for the different generated types.

5. Choose the input and output components for each operation node separately by generating the realization \( k \) of a random variable \( K \), which is uniformly distributed on the set \( \{1, \ldots, n\} \). The resulting number \( k \) refers to the index of the components. If a component is selected more than once, the duplicates are deleted leading to a smaller input degree. Note that for the selection of the output component of operation node \( i \), the input components of \( i \) must not be taken into account.

6. Decide if \( S_k \) or \( \overline{S}_k \) is assigned as an input, with a random number taking two equally probable values.

For the first type of random networks, we generate the number of inputs according to a \( \chi^2_f \) - distribution. The parameter of this distribution \( f \), the degrees of freedom, equals the mean of
|          | 10-5a | 10-5b | 20-10a | 20-10b | 100-50a | 100-50b | 100-100a | 100-100b |
|----------|-------|-------|--------|--------|---------|---------|----------|----------|
| # subs   | 10    | 10    | 20     | 20     | 100     | 100     | 100      | 100      |
| # ops    | 5     | 5     | 10     | 10     | 50      | 50      | 100      | 100      |
| min in   | 1     | 1     | 1      | 1      | 1       | 1       | 1        | 1        |
| max in   | 5     | 5     | 6      | 7      | 15      | 9       | 11       | 12       |
| avg in   | 3.2   | 3     | 2.6    | 3.96   | 3.44    | 3.27    | 3.48     |          |
| # sources| 6     | 5     | 7      | 6      | 38      | 30      | 33       | 34       |
| # sinks  | 1     | 1     | 2      | 3      | 6       | 5       | 3        | 2        |
| # variables | 26   | 20    | 40     | 46     | 208     | 214     | 252      | 262      |
| # rows   | 38    | 30    | 58     | 63     | 354     | 335     | 566      | 599      |
| time (sec) | 0.010 | 0.000 | 0.000  | 0.000  | 0.010   | 0.010   | 0.010    | 0.020    |
| # B&B nod| 0     | 0     | 0      | 0      | 0       | 0       | 0        | 0        |
| feas/infeas | f    | f     | f      | f      | f       | f       | f        | f        |

Table 4: Overview of the properties of type I networks, which are specified by number of components – number of operations. We always generated two networks, a and b, of the same size.

A_i. Hence, we set A_i to be $\chi^2_3$-distributed with three degrees of freedom in order to derive a ‘lean’ signaling network similar to the TCR network with three inputs per operation node on average. Results are displayed in Table 4. The number of molecules is denoted by ‘# subs’, ‘# ops’ denotes the number of AND/OR operations used and ‘min/max/avg in’ means the minimal/maximal/average number of elements on the left hand side of all IFFSAT formulas $S_i$. The number of inputs and outputs to the network are stated as ‘sources’ and ‘sinks’. The size of the IP is depicted in the following two rows. The IPs could be solved very fast even in large cases and without any branch and bound nodes used. The number of paths between sources and sinks is relatively large, there are, e.g., 14, 15, 33 and 21424 paths.

Additionally the shortest path between sinks and sources has length 2, even in the large networks. This is not a realistic network structure. To avoid this feature we force the network to hold more layers. This is done by generating several type I networks, such that the sinks of the previous network are the sources of the next one. Type II networks contain three Type I networks glued together, which is reasonable as the shortest path in the TCR model is 5. Type II networks are similar to the real model with respect to input degree number of paths and path length. Computations are listed in Table 5 and are as fast as in the Type I case. The number of paths from input to output layer decreases to 2, 1, 11, 422 and 9243, while its average length increases. Note that the number of components and operations characterizing the different networks differ from the entries in the table since they specify the amount of components and operations in each Type I network.

After the first two types were constructed similarly, the third type is generated distinctly. We set $R := U[1, 15]$, i.e. $A_i$ is uniformly distributed on the interval $[1, 15]$. To obtain integral input degrees, the generated random number is rounded up to the next integer. This results in an average input number of eight. Hence, the networks have comparably many arcs and are more or less ‘thick’. The number of paths between sink and source are like in Type I networks quite high, e.g. 24, 51, 26 494. In Table 6 the computational results are displayed.

Next, we employed the joint-generation method (Fredman and Khachiyan, 1996) as implemented in (Haus, 2008) to compute all minimal infeasible and maximal feasible vectors $y$. The
|                  | 5-1a | 5-1b | 10-5a | 10-5b | 30-10a | 30-10b | 50-25a | 50-25b |
|------------------|------|------|-------|-------|--------|--------|--------|--------|
| # subs           | 15   | 15   | 30    | 30    | 90     | 90     | 150    | 150    |
| # ops            | 3    | 3    | 15    | 15    | 30     | 30     | 75     | 75     |
| min in           | 1    | 1    | 0     | 1     | 0      | 0      | 0      | 0      |
| max in           | 3    | 1    | 6     | 6     | 5      | 8      | 8      | 9      |
| avg in           | 1.67 | 1    | 1.6   | 1.87  | 2.13   | 2.13   | 2.49   | 2.45   |
| # sources        | 2    | 1    | 8     | 4     | 12     | 14     | 30     | 29     |
| # sinks          | 1    | 1    | 4     | 2     | 2      | 5      | 10     | 13     |
| # variables      | 30   | 30   | 64    | 74    | 184    | 184    | 318    | 322    |
| # rows           | 22   | 22   | 74    | 70    | 186    | 186    | 428    | 429    |
| time (sec)       | 0.000| 0.000| 0.000 | 0.000 | 0.010  | 0.010  | 0.010  | 0.010  |
| # B&B nod        | 0    | 0    | 0     | 0     | 0      | 0      | 0      | 0      |
| feas/infeas      | f    | f    | f     | f     | f      | f      | f      | f      |

Table 5: Overview of the properties of type II networks, which are specified by number of components - number of operations. We always generated two networks, a and b, of the same size.

|                  | 25-5a | 25-5b | 50-10a | 50-10b | 100-20a | 100-20b | 150-50a | 150-50b |
|------------------|-------|-------|--------|--------|---------|---------|---------|---------|
| # subs           | 25    | 25    | 50     | 50     | 100     | 100     | 150     | 150     |
| # ops            | 5     | 5     | 10     | 10     | 20      | 20      | 50      | 50      |
| min in           | 3     | 2     | 1      | 1      | 1       | 1       | 1       | 1       |
| max in           | 9     | 10    | 13     | 10     | 14      | 13      | 14      | 14      |
| avg in           | 6.2   | 7     | 6.3    | 4.8    | 7.2     | 7.6     | 7.24    | 7.1     |
| # sources        | 11    | 9     | 20     | 13     | 42      | 42      | 75      | 73      |
| # sinks          | 1     | 2     | 5      | 2      | 6       | 4       | 4       | 2       |
| # variables      | 50    | 50    | 100    | 100    | 200     | 204     | 306     | 308     |
| # rows           | 65    | 61    | 123    | 108    | 263     | 277     | 568     | 561     |
| time (sec)       | 0.000 | 0.000 | 0.000  | 0.010  | 0.000   | 0.010   | 0.010   | 0.010   |
| # B&B nod        | 0     | 0     | 0      | 0      | 0       | 0       | 0       | 0       |
| feas/infeas      | f     | f     | f      | f      | f       | f       | f       | f       |

Table 6: Overview of the properties of type III networks, which are specified by number of components – number of operations. We always generated two networks, a and b, of the same size.
| TCRB | CD4 | CD28 | max feas | min feas | # oracle calls | time |
|------|-----|------|----------|----------|---------------|------|
|      |     |      |   # max |   avg |   min |   # max |   avg |   min |          |          |
| 0    | 0   | 0    | 1 324  | 324    | 324 | 0   | 1    | 324  | 324  | 0  | 1 0 |
| 0    | 0   | 1    | 1 322  | 322    | 322 | 2   | 323  | 323  | 323  | 974 | 0  |
| 0    | 1   | 0    | 1 322  | 322    | 322 | 2   | 323  | 323  | 323  | 974 | 0  |
| 0    | 1   | 1    | 1 320  | 320    | 320 | 4   | 323  | 323  | 323  | 1619| 0  |
| 1    | 0   | 0    | 2 323  | 322.5  | 322 | 2   | 322  | 322  | 322  | 1298| 0  |
| 1    | 0   | 1    | 2 321  | 320.5  | 320 | 4   | 323  | 322.5| 322  | 1943| 0  |
| 1    | 1   | 0    | 2 321  | 320.5  | 320 | 4   | 323  | 322.5| 322  | 1943| 0  |
| 1    | 1   | 1    | 2 319  | 318.5  | 318 | 6   | 323  | 322.3| 322  | 2584| 0  |

**Table 7:** Computation of minimal infeasible and maximal feasible vectors $y$ on the TCR model from [Saez-Rodriguez et al., 2007]: Comparison of the cardinalities and computation time, and the support of the 324-dimensional vector of $y$-variables for all 8 patterns of input signals. All timings below measurement threshold of 10 ms.

The feasibility oracle employed in this algorithm is solving integer feasibility problems of the form (5).

**Acknowledgments**

The first, second and fourth author were supported by the FORSYS initiative of the German Ministry of Education and Research through the Magdeburg Center for Systems Biology (MaCS) and the Research Focus Program Dynamic Systems funded by the Kultusministerium of Saxony-Anhalt.

**References**

Bertsimas, D. and Weismantel, R. (2005). *Optimization over Integers*. Dynamic Ideas, Belmont, Massachusetts, USA.

Büning, H. K. and Lettmann, T. (1999). *Propositional Logic: Deduction and Algorithms*, volume 48 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press.

Dantzig, G. B. (1963). *Linear Programming and Extensions*. Princeton University Press.

Feist, A. M.; Henry, C. S.; Reed, J. L.; Krummenacker, M.; Joyce, A. R.; Karp, P. D.; Broadbelt, L. J.; Hatzimanikatis, V.; and Palsson, B. O. (2007). "A genome-scale metabolic reconstruction for Escherichia coli K-12 MG1655 that accounts for 1260 ORFs and thermodynamic information". *Molecular Systems Biology*, 3(121).

Fredman, M. L. and Khachiyan, L. (1996). "On the complexity of dualization of monotone disjunctive normal forms". *J. Algorithms*, 21(3), pp. 618–628.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco.
Håstad, J. (2001). "Some optimal inapproximability results". *J. ACM*, 48(4), pp. 798–859.

Haus, U.-U. (2008). "cl-jointgen, a Common Lisp Implementation of the Joint-Generation Method". available at [http://www.primaldual.de/cl-jointgen](http://www.primaldual.de/cl-jointgen).

Haus, U.-U.; Truemper, K.; and Weismantel, R. (2007). "Linear Satisfiability Algorithm for 3CNF Formulas of Certain Signaling Networks". submitted to Journal on Satisfiability, Boolean Modeling and Computations.

Hooker, J. N. (2007). *Integrated Methods for Optimization*. Springer, Pennsylvania.

Kavvadias, D. J.; Sideri, M.; and Stavropoulos, E. C. (2000). "Generating all maximal models of a Boolean expression". *Information Processing Letters*, 74(3-4), pp. 157–162.

Saez-Rodriguez, J.; Simeoni, L.; Lindquist, J.; Hemenway, R.; Bommhardt, U.; Arndt, B.; Haus, U.-U.; Weismantel, R.; Gilles, E. D.; Klamt, S.; and Schraven, B. (2007). "An Extensive Logical Model Predicts Key Events in the T-cell Receptor Signaling Network". *PLoS Computational Biology*. DOI [http://dx.doi.org/10.1371/journal.pcbi.0030163](http://dx.doi.org/10.1371/journal.pcbi.0030163).

Truemper, K. (2004). *Design of Logic-based Intelligent Systems*. John Wiley & Sons.
A Appendix

Proof of Lemma 10. Without loss of generality let $I = \{1, \ldots, n\}$ implying the RIFFSAT formula $\forall_{i=1}^{n}(A_{i} \wedge y_{i}) \leftrightarrow B \wedge \forall_{i=1}^{n} y_{i}, I = \{1, \ldots, n\}$. Thus, let

$$F_{\text{mon}} = \left\{ \begin{pmatrix} A \\ y \\ B \end{pmatrix} \in [0, 1]^{2n+1} : B \geq A_{i} - (1 - y_{i}) \; \forall \; i = 1, \ldots, n \right\}$$

and

$$P = \left\{ \begin{pmatrix} A \\ y \\ B \end{pmatrix} \in [0, 1]^{2n+1} : B \geq A_{i} - (1 - y_{i}) \; \forall \; i = 1, \ldots, n \right\}.$$ 

We need to show that $P = \text{conv}(F_{\text{mon}})$. It is clear that $\text{conv}(F_{\text{mon}}) \subseteq P$. For the converse, assume that $P$ has a fractional vertex $f = (A, y, B)$. Since for a vertex $2n + 1$ inequalities have to be tight, $f$ has at least one integral component (there are only $2n$ inequalities without box conditions). Let $n_{f}$ be the number of fractional components of $f$; $0 \leq n_{f} \leq 2n$. It is to show that $n_{f} > 0$ gives a contradiction.

Let $k_{1}$ be the number of tight inequalities of the first type and $k_{2}$ the one of type two inequalities, $k_{1} + k_{2} = n_{f}$. $I_{j}$ denotes the corresponding equality index set of type $j = 1, 2$. To show the Lemma we will distinguish between three cases:

(i) $k_{1} > 0 \wedge k_{2} > 0$.

(ii) $k_{1} = n_{f} > 0 \wedge k_{2} = 0$.

(iii) $k_{1} = 0 \wedge k_{2} = n_{f} > 0$.

Case (i): Let $I_{1}, I_{2} \neq \emptyset$.

Since $B = \sum_{j=1}^{n} A_{j} + (1 - y_{l})$ for all $l \in I_{2}, y_{l}$ must have the same value for all $l \in I_{2}$. Additionally, for $i \in I_{1}$ and $l \in I_{2}$ it holds that

$$A_{j} - (1 - y_{l}) = \sum_{j=1}^{n} A_{j} + (1 - y_{l}) \iff 0 = \sum_{j \neq i} A_{j} + (1 - y_{l}) + (1 - y_{l}).$$

Since all variables are non-negative, we find that $A_{j} = 0 \forall j \neq i, y_{i} = y_{l} = 1 \forall l \in I_{2}$ and $A_{i} = B$. Thus, $A_{i \setminus I_{1}} = 0, y_{I_{1} \cup I_{2}} = 1$ and the only possible fractional components of $f$ are $y_{j}$ for $j \in I \setminus (I_{1} \cup I_{2}) = \{l_{1}, \ldots, l_{k}\}$ and $B = A_{j}$. But if they are fractional, we can construct a convex combination of integral points in $F_{\text{mon}}$, because the variables are all strictly between 0 and 1 and
all possible 0/1 combinations of the fractional variables are feasible points. For this purpose let w.l.o.g. the fractional \( y_i \) for \( j \in I \) variables in components of \( v \in \mathbb{B} \) for \( y \). Analogous to case (i), it follows that \( y \). The vectors used in this representation are all elements of \( f \). There are two cases to be considered: \( B = 1 \) and \( B = 0 \).

**Case (ii):** Let \( I_1 = \{1, \ldots, k_1\} \) and \( I_2 = \emptyset \). The number of integral components of \( f \) must be greater or equal to \( n+1 \), because \( n_f = k_1 \leq n \). Thus, there exists an index \( j \in I_1 \) such that the equality \( B = A_j - (1 - y_j) \) contains at least two integral variables. Thus, all contained variables are integral and in particular \( B \) is integral. In the equalities where only \( B \) is known to be integral, say \( B = A_k - (1 - y_k), k \in I_1, A_k \) and \( y_k \) can be fractional. There are two cases to be considered: \( B = 1 \) and \( B = 0 \).

**B=1:** \( :1 = A_i - (1 - y_i) \) \( \forall i \in I_1 \) \( \iff 2 = A_i + y_i : A_i = y_i = 1 \). Thus, the only fractional components of \( f \) can be \( A_{p_i} \) and \( y_{p_i} \). As \( B = 1 \) and \( A_i = y_i = 1 \) for all \( i \in I_1 \), there is a ‘reason’ for \( B \) being 1. Therefore the fractional components can be convexly combined as in case (i) since \( A_{p_i} \) and \( y_{p_i} \) can be any 0/1 combination to complete a feasible point for \( P \). Hence, \( f \) is no vertex of \( P \).

**B=0:** \( :A_i = 1 - y_i \) \( \forall i \in I_1 \). Let \( \{p_1, \ldots, p_m\} = I_1^f \subseteq I_1 \) denote the index set of the fractional \( A \)-variables in \( I_1 \) and thus their corresponding \( y \)-variables, which also have to be fractional according to the equality. For \( j \in I \setminus I_1 \) the inequalities imply that \( A_j + y_j < 1 \). We proceed similar to the previous case. W.l.o.g. assume that \( A_i \leq y_i \) \( \forall i \in I_1^f \) and that \( A_{p_i} \leq y_{j_i} \leq A_{p_2} \leq \cdots \leq A_{j_k} \leq y_{j_i} \leq A_{p_m}, \) with \( j_i \in I \setminus I_1, i = 1, \ldots, l \), are the ordered fractional components. For ease of notation we will leave out the integral \( A_i \) and \( y_i, i \in I_1 \setminus I_1^f \) and consider only the remaining components \( f^* \). Accordingly, \( v^A, v^y \) and \( e_B \) are adjusted to the new dimension. Then,

\[
\begin{align*}
\tilde{f} &= A_{p_1}(v^A)_{I_1 \cup I_1^f} + v^y_{I_1^f} + (y_{j_1} - A_{p_1})(v^A)_{(I_1 \cup I_1^f) \setminus \{p_1\}} + v^y_{(I_1 \cup I_1^f) \setminus \{p_1\}} + (A_{p_2} - y_{p_1})(v^A)_{(I_1 \cup I_1^f) \setminus \{p_1, p_2\}} + v^y_{(I_1 \cup I_1^f) \setminus \{p_1, p_2\}} + \cdots + (A_{j_k} - A_{p_{m-1}})(v^A)_{(I_1 \cup I_1^f) \setminus \{p_1, \ldots, p_{m-1}\}} + v^y_{(I_1 \cup I_1^f) \setminus \{p_1, \ldots, p_{m-1}\}} + (y_{j_k} - A_{j_k})v^y_{I_1^f} + (y_{p_{m-1}} - y_{p_m})v^y_{I_1^f}.
\end{align*}
\]

The vectors used in this representation are all elements of \( P \). Furthermore the convex multiplier sum up to 1, as \( A_i + y_i = 1 \) for \( i \in I_1^f \). Thus, \( f \) is not a vertex.

**Case (iii):** Let \( I_2 = \{1, \ldots, k_2\} \) and \( I_1 = \emptyset \). Analogous to case (i), it follows that \( y_1 = \ldots = y_{k_2} \). We also know that there are \( n_f = k_2 \) fractional components of \( f \).

If these fractional components are \( y_i, i \in I_2 \), the other variables are all integral. But the equality \( B = \sum_{j=1}^n A_j + (1 - y_i) \) for all \( i \in I_2 \) and the integrality of all involved variables but \( y_i \) yield that \( y_i \in \{0, 1\} \) \( \forall i \in I_2 \) and therefore \( f \) is integral.

Thus, \( y_i \) must be integral for all \( i \in I_2 \). We need to distinguish between 6 cases:
a) \( A_i \) for \( i \in I^A \subseteq I \) is fractional \((|I^A| = k_2)\),

b) \( y_i \) for \( i \in I^y \subseteq I \setminus I_2 \) is fractional \((|I^y| = k_2)\),

c) \( B, A_i, i \in I^A \subseteq I \) are fractional \((|I^A| = k_2 - 1)\),

d) \( B, y_i, i \in I^y \subseteq I \setminus I_2 \) are fractional \((|I^y| = k_2 - 1)\),

e) \( A_i, y_j, i \in I^A \subseteq I, j \in I^y \subseteq I \setminus I_2 \) are fractional \((|I^A| + |I^y| = k_2)\),

f) \( B, A_i, y_j, i \in I^A \subseteq I, j \in I^y \subseteq I \setminus I_2 \) are fractional \((|I^A| + |I^y| = k_2 - 1)\),

The case where only \( B \) is fractional does not have to be considered as equality in at least one type two inequality immediately forces \( B \) to be integral, as well.

a) With the equality it also follows that \( \sum_{i \in I^A} A_i \in \{0, 1\} \).

- \( \sum_{i \in I^A} A_i = 0 : A_i = 0 \forall i \in I^A \).  

- \( \sum_{i \in I^A} A_i = 1 : B = \sum_{j=1}^n A_j + (1 - y_j), l \in I_2 : B = 1, A_i = 0, i \in I \setminus I^A, y_j = 1, l \in I_2 \). As \( B = 1 \), we only need one \( A_i, i \in I^A \) to be 1, independent of the \( y \) pattern, for a feasible point.

Let \( I^A = \{i_1, \ldots, i_m\} \), then \( f \) can be convexly combined by

\[
f = A_{i_1}(e_{i_1}^A + v_{I_2}^y + v_S^y + e_B) + \cdots + A_{i_m}(e_{i_m}^A + v_{I_2}^y + v_S^y + e_B)
\]

for the subset \( S \) of \( I \setminus I_2 \), where \( y_j = 1 \). Since we know that \( \sum_{i \in I^A} A_i = 1 \), we have a real convex combination.  

b) As \( I^P \cap I_2 = \emptyset \), \( \sum_{j=1}^n A_j + (1 - y_k) = B < \sum_{j=1}^n A_j + (1 - y_i), i \in I^P : y_i < y_k \) for all \( k \in I_2 \).

- \( y_k = 0 \) for \( k \in I_2 : y_i < 0 \) for \( i \in I^P \).  

- \( y_k = 1 \). Thus, either \( B = 0 = A_1 = \ldots = A_n \) or (w.l.o.g.) \( B = 1 = A_1, A_2 = \ldots = A_n = 0 \).

In both cases, we can choose any \( y_P \) pattern to get a feasible point in \( P \). Therefore, one can follow the same procedure to construct a convex combination of integral points as in case (i).

c) Again, equality implies \( B = \sum_{i \in I^A} A_i \in \{0, 1\} \).

- \( B - \sum_{i \in I^A} A_i = 1 : B = 1, A_i = 0, i \in I^A \).  

- \( B - \sum_{i \in I^A} A_i = 0 \iff 0 = \sum_{j \notin I^A} A_j + (1 - y_k) \forall k \in I_2 \):

- \( A_j = 0, j \notin I^A \) and \( y_k = 1, k \in I_2 \). Additionally, \( y_j \in \{0, 1\} \) for \( j \in I \setminus I_2 \) and for a subset \( S \subseteq I \setminus I_2 y_j = 1 \). Then we can convexly combine integral elements of \( P \) to obtain \( f \). For this, let \( I^A = \{i_1, \ldots, i_m\} \):

\[
f = A_{i_1}(e_{i_1}^A + v_{I_2}^y + v_S^y + e_B) + \cdots + A_{i_m}(e_{i_m}^A + v_{I_2}^y + v_S^y + e_B) + (1 - \sum_{i \in I^A} A_i)(v_{I_2}^y + v_S^y)
\]

Therefore, \( f \) is no vertex.  

d) Since \( B = \sum_{j=1}^n A_j + (1 - y_k) \), \( k \in I_2 \) and all variables involved except for \( B \) are integral, \( B \) must be integral as well. Hence, we are in case b).

e) As shown in case b) it follows that \( y_i < y_k \) for all \( i \in I^P \) and \( k \in I_2 \) and with this \( y_k = 1 \forall k \in I_2 \). This implies that \( B = \sum_{j=1}^n A_j \) and hence \( \sum_{i \in I^A} A_i \in \{0, 1\} \).
\begin{itemize}
\item $\sum_{i \in I^A} A_i = 0 : A_i = 0 \forall i \in I^A$. The only remaining fractional components are $y_j, j \in I'$. The possible cases for the integral components are analogous to case b) and they can be convexly combined as in case (i). ∄
\item $\sum_{i \in I^A} A_i = 1 : A_i = 0 i \not\in I^A, B = 1, y_j = 1, j \in I_2$. We again want to apply the scheme used in case (i), i.e. look for the minimum within the fractional components, set all fractional components to 1 in the convex combinator, multiply it by the lowest value, then take the minimal difference between the fractional entries and the previous minimum, set the component of the previous minimum to 0 and multiply it by the difference of the two. For this purpose it is necessary to make sure that there is always at least one $A_i, i \in I^A$, that is one in the convex combinator. This can be achieved by adjusting the system a little bit to a combination of the procedures of case (i) and case (iii), a): We set only one $A_i$ at a time to 1 until its value is reached and then the next becomes 1. For the $y$ components an arbitrary pattern can be chosen. For an easier description assume that $I^A = \{i_1, \ldots, i_m\}, I' = \{j_1, \ldots, j_k\}$ and $A_{i_1} \leq y_{j_1} \leq A_{i_2} \leq \ldots$

\[ f = A_{i_1}(e^A_{i_1} + v^y_{I_2} + v^y_{I'} + e_B) + (y_{j_1} - A_{i_1})(e^A_{i_2} + v^y_{I_2} + v^y_{I'} + e_B) \\
+ (A_{i_2} - y_{j_1})(e^A_{i_3} + v^y_{I_2} + v^y_{I' \setminus \{j_1\}} + e_B) + \ldots \]

Since $\sum_{i \in I'} A_i = 1$ it follows that the constructions yields a real convex combination. ∄
\end{itemize}

f) In this case one can construct a convex combination of $f$ by using a combination of the cases e) and c). ∄

In a all cases the fractional vertex $f$ could be convexly combined and thus can not be a vertex. This implies $P = \text{conv}(F_{mon})$