JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 17, Number 1, 2010
© Mary Ann Liebert, Inc.
Pp. 39–53
DOI: 10.1089/cmb.2009.0121

# Computing Combinatorial Intervention Strategies and Failure Modes in Signaling Networks

REGINA SAMAGA, AXEL VON KAMP, and STEFFEN KLAMT

## ABSTRACT

**The identification of combinatorial intervention strategies and the elucidation of failure modes that may cause aberrant behavior of cellular signaling networks are highly relevant topics in cell biology, medicine, and pharmaceutical industry. We have recently introduced the concept of minimal intervention sets (MISs)—minimal combinations of knock-ins and knock-outs provoking a desired/observed response in certain target nodes—to tackle those problems within a Boolean/logical framework. We first generalize the notion of MISs and then present several techniques for search space reduction facilitating the enumeration of MISs in networks of realistic size. One strategy exploits topological information about network-wide interdependencies between the nodes to discard unfavorable single interventions. A similar technique checks during the algorithm whether all target nodes of an intervention problem can be influenced in appropriate direction (up/down) by the interventions contained in MIS candidates. Another strategy takes lessons from electrical engineering: certain interventions are equivalent with respect to their effect on the target nodes and can therefore be grouped in fault equivalence classes (FECs). FECs resulting from so-called structural equivalence can be easily computed in a preprocessing step, with the advantage that only one representative per class needs to be considered when constructing the MISs in the main algorithm. With intervention problems from realistic networks as benchmarks, we show that these algorithmic improvements may reduce the computation time up to 99%, increasing the applicability of MISs in practice.**

**Key words:** Boolean networks, diagnosis, drug target identification, failure equivalence classes, signal transduction networks.

## 1. INTRODUCTION

Cellular signaling and regulatory networks are of utmost importance for a coherent functioning of living systems. Aberrant behavior of these networks may lead to severe diseases such as cancer. An important topic highly relevant for cell biology, medicine, and pharmacy is the search for intervention strategies in signaling networks that may counteract pathological behavior (without inducing side effects) or overwrite the natural signal processing. Given a biological outcome to be repressed or provoked, possible interventions to fulfill this goal have to be identified. In the majority of cases, multiple interventions are

Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg, Germany.

necessary to cope with the redundancy and multifunctionality characteristic of biological networks. Redundancy requires that several routes must be targeted whereas multifunctionality implies that intervening at central players often causes unintended side effects so that one has to search for alternative and possibly more distributed points of interventions (Fitzgerald et al., 2006). The complicated structure of signaling networks and the wealth of data and biochemical information generated call for a systemic approach and thus the employment of mathematical models.

Depending on the system under study and the problem one wants to address, different modeling techniques have been used to describe signal transduction networks. Dynamic modeling approaches based on differential equations are widely used for studying the behavior of selected signaling pathways and allow for a detailed analysis of the kinetic behavior. In contrast, qualitative approaches are limited in reflecting kinetic and quantitative aspects but can in exchange be applied to large-scale (and thus more complete) networks and may sometimes better reflect the qualitative biological knowledge typically available (e.g., pathway maps obtained from databases). An example for qualitative approaches are Boolean (logical) models. Originally applied to random networks (Kauffman, 1969) or gene regulatory networks (Thomas, 1973; Thomas and D'Ari, 1990; Mendoza et al., 1999; Albert and Othmer, 2003), a number of recent publications show that Boolean models are also well-suited for analyzing a number of important functional aspects of large signaling networks (Saez-Rodriguez et al., 2007; Franke et al., 2008; Sahin et al., 2009; Christensen et al., 2009; Samaga et al., 2009). Herein, we consider the Boolean modeling framework as introduced by Klamt et al. (2006) tailored for studying the qualitative input-output response of signaling networks. Within this framework, we define minimal intervention sets (MISs) as support-minimal sets of Boolean interventions (knock-ins and knock-outs) that fulfill a given intervention goal (i.e., executing the interventions contained in an MIS will force a set of defined target nodes into a desired state).

Figure 1 shows a toy model as an example for a Boolean model of a signaling network. The nodes in the network may represent biological species such as receptors, kinases, or phosphatases, each having an associated logical state determining whether the species is active/present (1) or not (0). Signaling events are encoded as Boolean operations on the network nodes. A possible problem is to find MISs that lead to an inactivation of O1 ($O1 = 0$) and at the same time to an activation of O2 ($O2 = 1$). This can, for example, be achieved by setting E to 1, and simultaneously, C to 0.

The notion of MISs in signaling or regulatory networks is in several aspects related to the concept of minimal cut sets (Klamt, 2006) introduced for metabolic reaction networks carrying mass instead of signal flows. Minimal cut sets are defined as minimal sets of reactions that, when removed, repress a given functionality in the network (e.g., synthesis of a certain metabolite). As has been pointed out for minimal cut sets, several applications of MISs lie beyond the apparent task of drug target identification. MISs can be used as a diagnosis tool, for example, to identify failure modes in a signaling network that might cause an observed aberrant or pathological behavior. Related to network reconstruction, MISs can also be used to

**A**

I1 (Input)
I2 (Input)
A = ! D
B = A • I1
C = B + E
D = C
E = ! I1 • I2
F = E + G
G = F
O1 = C
O2 = G

**B**

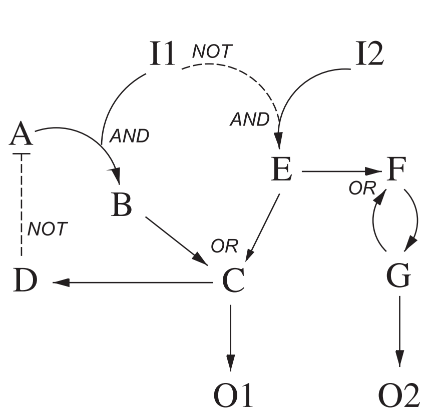


**FIG. 1.** Example of a logical signal transduction network (toy model): Boolean functions (**A**) and the resulting logical interaction hypergraph representation (**B**).

find necessary changes in network structure to reproduce experimental results (Saez-Rodriguez et al., 2007). Finally, MISs may uncover the importance of network elements for certain biological functions and help to detect fragile points in the network.

The enumeration of MISs is computationally expensive: in a network with $n$ nodes, $3^n$ possible combinations of interventions exist (each MIS can be represented by an n-element vector where each element [node] is either $-1$ [knock-out], 0 [no intervention], or 1 [knock-in]). Thus, even in the simple example above, $3^{11}$ candidates exist and 43 MISs fulfilling the intervention goal can be eventually identified. Therefore, in order to facilitate the deployment of MISs in realistic large-scale problems, algorithmic approaches are required to cope with the combinatorial complexity of this problem. The aim of this work is to introduce several techniques that reduce the search space without losing any solution. We start with definitions giving a rigorous and, compared to the original definition, generalized framework of MISs.

## 2. DEFINITIONS

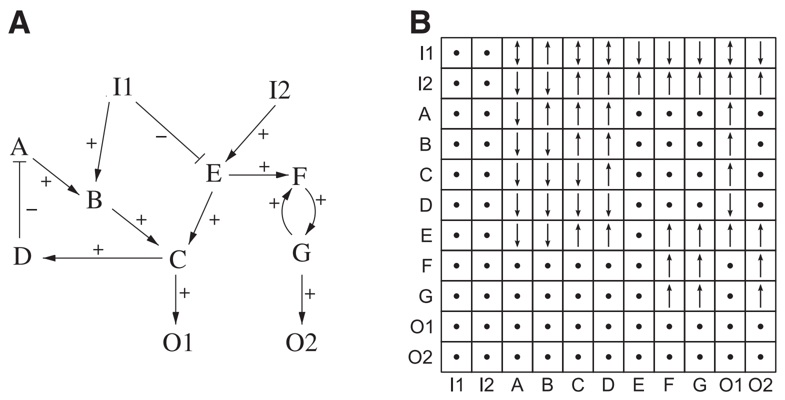
We consider Boolean networks $\mathbf{B} = (V, F)$, where $V = \{v_1, \ldots, v_n\}$ denotes a set of nodes (proteins, genes, or other biological entities) and $F = \{f_1, \ldots, f_n\}$ a set of Boolean functions (representing biochemical interactions between the nodes). Each node $v_i$ has an associated logical variable $x_i$ that can be determined by its corresponding Boolean function $f_i : \mathbf{x} \to \{0, 1\}, \mathbf{x} = (x_1, \ldots x_n)^T$. We assume that the Boolean functions contain only AND ($\cdot$), OR ($+$) and NOT (!) operators and are represented in (usually prime) disjunctive normal form (DNF; also known as sum-of-products representation). A DNF is a disjunction of terms where each term is either a single, possibly negated Boolean variable (a literal) or a conjunction of several literals (i.e., a DNF is composed of OR-connected (AND-)terms).[1] Note that any Boolean function can be represented in this way. Usually, logical variables in Boolean networks are binary (i.e., take only the values "1" or "0"). However, in the following, it is convenient to allow a third value "*" that represents an undefined or unknown state of a Boolean variable. The Boolean operators can be generalized straightforwardly to this three-valued logic as follows (Abramovici et al., 1990): the AND operator applied to arbitrary many variables $x_i \in \{0, 1, *\}$ returns 1 if and only if all $x_i$ are 1; it returns 0 iff at least one $x_i$ is 0 and it returns * in all other cases. The OR operator returns 0 iff all of its arguments $x_i$ are 0; it returns 1 iff at least one $x_i$ is 1, and it returns * in all other cases. Finally, the negated value of * is *, and as in the binary case, NOT(0) is 1 and NOT(1) is 0.

A Boolean network where the Boolean functions are given in DNF can be represented as directed hypergraph. A hypergraph $\mathbf{H} = (V, A)$ consists of a set $V$ of nodes and a set $A$ of directed hyperarcs. In the most general case, each directed hyperarc connects an arbitrary number of start (or tail) nodes with an arbitrary number of end (or head) nodes, i.e., each hyperarc $a_i \in A$ is an ordered pair $(S_i, E_i)$ with $S_i, E_i \subset V$. Hyperarcs considered in the following have only one head node (also called backward hyperarc), i.e., $|E_i| = 1$ for all $i$. By using the DNF formalism for each logical function, the logical network can be converted to the hypergraph representation in a straightforward manner. The hypergraph contains the same set of nodes $V$ as the given Boolean network $\mathbf{B} = (V, F)$. Each term in the DNF representation of the function $f_i$ is displayed as hyperarc with head $v_i$. Those nodes whose associated logical variables are the arguments of the respective term form the tail of the hyperarc.[2] As the terms are OR-connected in the DNF, different hyperarcs pointing into the same node represent the OR operation. Finally, to consider NOT operators, we need to extend each hyperarc $a_i = (S_i, E_i)$ to $a_i = (S_i, E_i, l_i)$, where $l_i$ is a mapping $l_i : S_i \to \{+, -\}$ assigning each start node a sign label indicating whether the associated logical variable is negated ($-$) in the respective term or not ($+$). Signed directed hypergraphs as described above that are induced by a logical network represented in DNF are also called logical interaction hypergraphs (LIHs) (Klamt et al., 2006, 2007). An example (toy model) of a logical network represented as LIH is shown in Figure 1. Negated inputs (i.e., minus signs at certain hyperarc branches) are here indicated by dashed lines.

Any LIH (and thus any Boolean network) can be uniquely projected onto an interaction graph, i.e., onto a signed directed graph: the nodes are the same as in the hypergraph and for each node $v_j$ that occurs

[1]In our application, we can assume that the terms are the prime implicants of the Boolean function.

[2]If a term in the DNF of $f_i$ consists only of a single literal $x_k$ the hyperarc becomes a simple directed arc connecting $v_k$ with $v_i$.



| | I1 | I2 | A | B | C | D | E | F | G | O1 | O2 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| I1 | • | • | ↕ | ↑ | ↕ | ↕ | ↓ | ↓ | ↓ | ↕ | ↓ |
| I2 | • | • | ↓ | ↓ | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ | ↑ |
| A | • | • | ↓ | ↑ | ↑ | ↑ | • | • | • | ↑ | • |
| B | • | • | ↓ | ↓ | ↑ | ↑ | • | • | • | ↑ | • |
| C | • | • | ↓ | ↓ | ↓ | ↑ | • | • | • | ↑ | • |
| D | • | • | ↓ | ↓ | ↓ | ↓ | • | • | • | ↓ | • |
| E | • | • | ↓ | ↓ | ↑ | ↑ | • | ↑ | ↑ | ↑ | ↑ |
| F | • | • | • | • | • | • | • | ↑ | ↑ | • | ↑ |
| G | • | • | • | • | • | • | • | ↑ | ↑ | • | ↑ |
| O1 | • | • | • | • | • | • | • | • | • | • | • |
| O2 | • | • | • | • | • | • | • | • | • | • | • |

**FIG. 2.** Interaction graph and dependency matrix of the toy model (Fig. 1). (**A**) Interaction graph as derived from the hypergraph in Figure 1B. (**B**) Corresponding dependency matrix. Each entry in the matrix indicates the influence of the species in the row on the species in the column. ↑, activator; ↓, inhibitor; ↕, ambivalent factor; •, neutral factor (not affecting).

with positive (negative) sign label in the tail of at least one hyperarc with head node $v_i$ a positive (negative) edge is drawn from $v_j$ to $v_i$. This can be interpreted as splitting the hyperarcs (representing the ANDs) and subsequent removal of redundant arcs that may occur, for instance, in a logical function like $A = B \cdot C + B \cdot D$. The corresponding interaction graph of the toy model (Fig. 1B) is shown in Figure 2A. Although the resulting interaction graph cannot be uniquely mapped back to the logical network, it captures important topological properties such as influence paths and interdependencies (Klamt et al., 2006) that we will exploit for search space reduction when computing minimal intervention sets.

The formalism of logical interaction hypergraphs can be used to conveniently describe cellular signaling networks and analyze their structural and functional behavior. The nodes of the hypergraph represent the biological species (e.g., kinases, ligands, receptors) whereas the hyperarcs refer to the activation mechanisms. Different hyperarcs pointing into the same node reflect alternative ways to activate this node. The logical variable associated to each node reflects whether the species is active/present (1) or not (0).

Given a logical network, one can study its qualitative dynamic behavior (i.e., analyze how the state of the system changes over time). Besides the classical synchronous model where one assumes that the states of all species are updated at the same time, i.e., $x_i(t+1) = f_i(\mathbf{x}(t))$ for all $i$, asynchronous models have been studied allowing the consideration of different time delays (Thomas, 1973; Thomas and D'Ari, 1990). A logical steady state (LSS) of the system is a state vector $\mathbf{x}^{\mathbf{S}}$ in which the state of each node obeys the value of its associated Boolean function: $x_i^S = f_i(\mathbf{x}^{\mathbf{S}})$ for all $i$. A Boolean network with binary logic may contain no, one, or several LSSs, all of which are equivalent for synchronous and asynchronous scheduling. For practical reasons, in our setting we consider the LSS with respect to the three-valued logic introduced above which will allow us to deal with unknown/undefined states. More specifically, we consider here the LSS in three-valued logic (denoted by LSS$_3$) that follows from a given set of fixed (clamped) values $x_i^0 \in \{0, 1\}$, $i \in I_0 \subseteq \{1, \ldots, n\}$, which either represent external stimuli (inputs) or introduced interventions (knock-outs/knock-ins). Clamping certain nodes $v_i, i \in I_0 \subseteq \{1, \ldots, n\}$, to a fixed value thus means that the Boolean functions $f_i$ of these nodes are overwritten by the simple binary values $x_i^0 \in \{0, 1\}$. The LSS$_3$ is then a state vector $\mathbf{x}^{\mathbf{S}} \in \{0, 1, *\}^n$ for which holds that $x_i^S = f_i(\mathbf{x}^{\mathbf{S}})$ for all $i \in \{1, \ldots, n\} \setminus I_0$ whereas $x_i^S = x_i^0$ for all $i \in I_0$. While in the case of two-valued Boolean networks several LSSs might result from a given set of fixed values (non-uniqueness) or a LSS may not exist at all (e.g., non-existence due to negative feedback loops), a unique LSS$_3$ follows for any given set of clamped values in the three-valued logic. For example, if no value is fixed in the network, all nodes will have state * in the associated LSS$_3$. In the general case, a subset of the nodes will have a "defined" value (0 or 1), and the others will be in the undefined (*) state.[3] Nodes with * in LSS$_3$ can be interpreted as states which cannot be uniquely resolved to a 0/1 value either because the initial values of some other nodes need to be known to infer a unique 0/1

[3]The subset of 0/1-valued nodes in LSS$_3$ corresponds to the term partial logical steady state used in Klamt et al. (2006).

value or because these nodes will never reach a fixed value because they are part of a negative feedback loop inducing oscillations in the Boolean (and possibly also in the continuous) dynamics.

The $LSS_3$ can be determined in an iterative and intuitive way by signal propagation using the logical functions for three-valued logic. The algorithm actually simulates the signal propagation taking place in the real situation: one initializes the state vector $\mathbf{x}$ with * except for the clamped nodes whose value is fixed according to the given values, i.e., $x_i \in \{0, 1\}$ for all $i \in I_0 \subseteq \{1, \dots, n\}$. We initialize an index set $I_K$ which contains all those node indices for which we have found a 0 or 1 value that cannot change in the future. Clearly, at the beginning we have $I_K = I_0$. In each iteration we then look for nodes $v_i$, $i \notin I_K$, for which we can deduce a value 0 or 1 given the current state vector $\mathbf{x}$. If we can change the state of a node $v_m$ from * to 0 or 1, then we update its state $x_m$ accordingly, include the index m in $I_K$ and start a new iteration. The algorithm stops if no further node could be added to $I_K$ and $\mathbf{x}$ then represents the $LSS_3$ $\mathbf{x^S}$. The algorithm is polynomial in input size and will terminate at latest after $n - |I_0|$ iterations because a node will not change its state once it has been assigned a 0/1 value. This follows from the conservative nature of three-valued logic. Note that this algorithm was outlined in Klamt et al. (2006), though not explicitly embedded in three-valued logic.

An example: consider we want to compute the $LSS_3$ that results from the given values $x_{I1}^0 = 1$ and $x_D^0 = 0$ in the toy model (Fig. 1). By clamping these values and initializing all other nodes with * the states of A and E will be updated in the first iteration of the algorithm ($x_A^S = 1, x_E^S = 0$). In the next step, we can only conclude $x_B^S = 1$; the state of F remains * (it cannot be resolved to 0 or 1 due to the positive feedback $F \rightarrow G \rightarrow F$). In the following two iterations we can deduce $x_C^S = 1$ and $x_{O1}^S = 1$. Then the algorithm stops as no further node with state * can be updated. At the end of the calculation, we result in the unique $LSS_3$ in which we have 0/1 ("known") values for A, B, C, E, and O1 (indicating the on/off states of these nodes that we expect as response in experiments where I1 is present and D set inactive), whereas the values of I2, F, G, and O2 remain in the "unknown" state *.

The idea of intervention sets is to search for combinations of knock-ins and knock-outs in the network that provoke a desired network response. In other words, we search for a pattern of species values to be fixed (the intervention set), so that certain species reach a predefined value in the resulting $LSS_3$. In the logical framework, interventions (knock-ins/knock-outs) are treated as nodes whose value is clamped to 1 or 0, respectively. In the biological context, examples for knock-ins are mutations that lead to constitutively activated species or a continuous stimulation with external signals (e.g., by a ligand) whereas a knock-out may correspond to gene knock-outs, specific blocking of the activity of a certain species or—approximately—the knock-down of certain nodes using RNA interference techniques. The desired network response is specified by the intervention goal $G = (T, t)$ consisting of a set of target species $T = \{v_i\}_{i \in I_T}, I_T \subseteq \{1, \dots, n\}$, and a mapping $t : T \rightarrow \{0, 1\}$ that specifies the desired values of the target species. To distinguish between nodes that shall be inhibited and nodes that shall be activated, we introduce the sets $T^- = \{v_i \in T : t(v_i) = 0\}$ and $T^+ = \{v_i \in T : t(v_i) = 1\}$. In the examples given below, we will sometimes simply write $G = \{v_i = t(v_i)\}_{i \in I_T}$ to declare the intervention goal. Optionally, we may fix the state of a set of nodes $E = \{v_i\}_{i \in I_E}$, $I_E \subseteq \{1, \dots, n\}$, by $e : E \rightarrow \{0, 1\}$ as natural or external side constraints $C = (E, e)$. This enables us to state the conditions under which the intervention goal should be fulfilled, for example, in the presence/absence of a ligand. The intervention goal together with the given side constraints forms an intervention scenario $R = (G, C)$. As most general case, we will also consider intervention problems $P$ that may be composed of several intervention scenarios $R_j = (G_j, C_j)$ and for which we search for intervention sets that fulfill for each scenario $R_j$ the intervention goal $G_j = (T_j, t_j)$ considering the respective side constraints $C_j = (E_j, e_j)$. An intervention set for the intervention problem $P = \{R_1, \dots, R_k\}$ can now be defined as a set of states $x_i^* \in \{0, 1\}$, $i \in I_* \subseteq \{1, \dots, n\}$, obeying the following: for each scenario $R_j$, $j \in \{1, \dots, k\}$, it holds for all its target indices $i \in I_{T_j}$ that $x_i^{S,j} = t_j(v_i)$ where $x_i^{S,j}$ denotes the $LSS_3$ of species $v_i$ that results from the fixed values $x_i^{0,j}$ given by the interventions and side constraints ($x_i^{0,j} = x_i^*$ for all $i \in I_*$ and $x_i^{0,j} = e_j(v_i)$ for all $i \in I_E \setminus I_*$). As for the intervention goals, we will sometimes use the simplified notation $\{v_i = x_i^*\}_{i \in I_*}$ to specify an intervention set. For practical reasons we are only interested in (support-) minimal intervention sets (MISs), in which no subset of the involved interventions fulfills the intervention goal. Furthermore, we concentrate on the nodes of the network although, in principle, we could also search for interventions at the hyperarcs. However, in cell signaling, the species rather than the interactions can be externally controlled—in contrast to metabolic networks, where typically the reactions are subject to interventions (Klamt, 2006).

We want to exemplify the concept of MISs by the toy model given in Figure 1. Consider the intervention scenario $R_1 = (G_1, C_1) = ((T_1, t_1), (E_1, e_1))$ with the set of target species $T_1 = \{O1, O2\}$ with $t_1(O1) = 0$ and $t_1(O2) = 1$ and the set of fixed nodes $E_1 = \{I1\}$ with $e_1(I1) = 1$. An MIS for the intervention problem $P_1 = \{R_1\}$ is, for instance, $\{D = 1, F = 1\}$. We now extend the intervention problem $P_1$ to $P_2 = \{R_1, R_2\}$ considering an additional intervention scenario $R_2$ with intervention goal $G_2 = \{A = 1\}$ (i.e., $T_2 = \{A\}$, $t_2(A) = 1$) without side constraints (i.e., $E_2 = \emptyset$). The set $\{C = 0, F = 1\}$ is an MIS for the intervention problem $P_2 = \{R_1, R_2\}$, whereas $\{D = 1, F = 1\}$ is not sufficient anymore. The complete sets of MISs for the two problems are given in Table 1.

## 3. GENERAL ALGORITHM FOR COMPUTING MISs

Our basic algorithm for computing MISs, as outlined in Klamt et al. (2006, 2007) (see also pseudo-code in Section 5), checks systematically combinations of interventions (first of size 1, then of size 2, and so on) whether or not they lead to a fulfillment of the intervention goal when the network reaches the logical

TABLE 1. MISs FOR THE INTERVENTION PROBLEMS $P_1$ AND $P_2$

| | | | | *MISs for problem $P_1$* | | | | |
|---|---|---|---|---|---|---|---|---|
| *I1* | *I2* | *A* | *B* | *C* | *D* | *E* | *F* | *G* |
| | | | | × | | • | | |
| | | | | × | | | • | |
| | | | | × | | | | • |
| × | × | | | | | | • | |
| × | × | | | | | | | • |
| × | • | | | × | | | | |
| × | | | | | | × | • | |
| × | | | | | | × | | • |
| | | × | | | | | • | |
| | | × | | | | | | • |
| | | | | | • | | • | |
| | | | | | • | | | • |
| | | | × | | | | • | |
| | | | × | | | | | • |
| | | | | *MISs for problem $P_2$* | | | | |
| *I1* | *I2* | *A* | *B* | *C* | *D* | *E* | *F* | *G* |
| | | | | × | | • | | |
| | | | | × | | | • | |
| | | | | × | | | | • |
| × | × | | | | | | • | |
| × | × | | | | | | | • |
| × | • | | | × | | | | |
| × | | | | | | × | • | |
| × | | | | | | × | | • |
| | × | | × | | | | • | |
| | | | × | | × | | • | |
| • | | | × | | | | • | |
| | | | × | | | × | • | |
| | × | | × | | | | | • |
| | | | × | | × | | | • |
| • | | | × | | | | | • |
| | | | × | | | × | | • |

The MISs are given in the rows.
MISs including interventions for the target species are not shown.
MISs, minimal intervention sets; •, knock-in; ×, knock-out.

steady state ($LSS_3$). For minimality, supersets of found MISs need not be further considered. Clearly, this brute-force approach leads quickly to an explosion of possible candidates for higher cardinalities. Therefore, strategies are required to make this computation also feasible in larger networks. One simple yet very useful strategy is to limit the number of allowed interventions per MIS. In fact, one is typically interested in the small MISs (Saez-Rodriguez et al., 2007). Other strategies try to reduce the search space without losing any solution, and in the following section, we shall present several of those techniques.

## 4. METHODS FOR SEARCH SPACE REDUCTION

The primary goal of search space reduction is to reduce the number of candidates that have to be tested for the MIS property. Although the computation of the $LSS_3$ for a given MIS is polynomial, these tests turn out to be the most time-consuming parts of the whole algorithm. We use two different strategies for search space reduction: (i) in a pre-processing step one seeks to safely discard some of the $2n$ possible single interventions; either because they can be shown to be not part of any MIS (see Section 4.1) or because there are interventions being equivalent with respect to the intervention goal(s) so that only one representative needs to be considered from which in a post-processing step the MISs with the equivalent interventions can be easily obtained (see Section 4.2). (ii) A second technique, applied during the main algorithm, is to discard certain MIS candidates without explicit testing by employing some simple rules (see Section 4.3).

In the whole section, we consider only intervention problems with one single intervention scenario. The extension to intervention problems with multiple intervention scenarios is described in Section 5.

### *4.1. Exploiting dependencies*

The underlying interaction graph of the Boolean network (see Section 2) can be used to examine direct and indirect interdependencies between the nodes. A suitable technique to discard uninfluential single interventions is to consider only those interventions that can affect the target species in the desired direction. The required information is captured in the dependency matrix that can be derived from the interaction graph (Klamt et al., 2006). Although the computation of the dependency matrix is in general an NP-complete problem, the algorithms we are using (Klamt and von Kamp, 2009) compute the matrix reasonably fast (less than one second) in realistic networks we considered. Alternatively, algorithms delivering approximations with high precision are available, and in networks without negative feedbacks, an exact polynomial algorithm can be used (Klamt and von Kamp, 2009).

From the dependency matrix, we cannot only deduce whether a certain species influences at least one target node at all (if not, then neither knock-in nor knock-out needs to be considered for this node), but also whether it acts as activator and/or inhibitor. A node $A$ is an activator (inhibitor) of $B$ if only positive (negative) paths lead from $A$ to $B$. If $B$ is a target to be switched on, a knock-out (knock-in) of $A$ cannot contribute to achieve this goal and thus needs not to be considered. Taking those interdependencies into account may drastically decrease the number of interventions and thus of MIS candidates to be checked. To illustrate this, consider the toy model given in Figure 1. Assume the intervention goal is to activate O1 (i.e., $T^+ = \{O1\}$) under the side constraint that I2 is set to 1 (i.e., $E = \{I2\}$ and $e(I2) = 1$). The interaction graph and resulting dependency matrix of the toy model are shown in Figure 2. As the species F, G and O2 do not influence O1 (shown by bullets in the respective fields of the dependency matrix, Fig. 2B), neither knock-ins nor knock-outs need to be considered for these nodes. D is an inhibitor of O1, implying that a knock-in of this node may only decrease the value of O1 (contrary to the intervention goal) thus suggesting that only knock-outs of D are useful. Note that we do not need to distinguish between weak/strong activator as usually done in the dependency matrix (Klamt et al., 2006, 2007) as we are here only interested in the logical steady state, not in states that can be reached during the transition. Analogously, knock-outs of I2, A, B, C, E, and obviously O1 can be safely discarded from the list of interventions. Activation of I2 needs not to be considered as intervention as this is already fulfilled by the given side constraint $e(I2) = 1$. I1 is an ambivalent factor for O1 (positive and negative paths from I1 to O1 exist), and therefore knock-out and knock-in of I1 is allowed. In this example, the number of single interventions reduces from 22 to 8 (five nodes can be set to 1, one node to 0, and one node to 0 and 1) and the total number of possible MIS candidates thus decreases from $3^{11}$ to $3 \cdot 2^6$.

### 4.2. *Exploiting fault equivalence classes*

Another pre-processing step for reducing the number of single interventions exploits the equivalence of certain interventions with respect to the intervention goal. Two interventions W and Z are equivalent if W can be replaced by Z in all MISs where W is a part of yielding a new set of MISs (the same must hold in the other direction). Hence, it suffices to consider only one representative of each class of equivalent interventions. Searching for equivalent interventions with respect to a certain intervention goal is highly related to equivalence fault collapsing in logical networks studied for systems testing in electrical engineering (McCluskey and Clegg, 1971; Abramovici et al., 1990). In electrical engineering, one considers stuck-at-0 (s-a-0) faults and stuck-at-1 (s-a-1) faults which may occur, for instance, by physical damages. These faults directly correspond to the knock-outs (s-a-0) and knock-ins (s-a-1) in our Boolean setting (though we are interested in constructing suitable intervention sets whereas in electrical engineering one is interested in identifying faults) and we can easily adapt the methods from fault collapsing for our application. We restrict ourselves to considering structurally equivalent interventions that can be determined by a local analysis of the network. The more general concept of functional equivalence relations requires a global analysis and determining whether two arbitrary faults are functionally equivalent is an NP-complete problem (Abramovici et al., 1990).

The occurrence of equivalent interventions in logical networks is based on the fact that the controlling value of an AND gate is 0 and the controlling value of an OR gate is 1; i.e., setting one of the input values of an AND gate to 0 determines the gate output to 0 whereas setting one of the input values of an OR gate to 1 determines the gate output to 1 (Abramovici et al., 1990). The intervention $B = 1$ where $B$ is the output of an OR gate is therefore equivalent to setting any input of the OR gate to 1. In analogy, the intervention $B = 0$ where $B$ is the output of an AND gate is equivalent to setting any input of the AND gate to 0. To state the conditions under which two interventions in our network are equivalent, recall that the network is represented as logical interaction hypergraph where each hyperarc represents an AND gate and different hyperarcs pointing into the same node are OR-connected. Furthermore, the inputs of the AND gates can be negated using the NOT operator. A hyperarc that has only one tail node represents a simple activation or inhibition depending on whether the value of the input node is negated or not. The knock-in intervention $B = 1$ (s-a-1 fault) is equivalent to $A = 1$ (or to $A = 0$) with respect to the set of target species $T$ if the following conditions hold:

(1.1) $A$ is a direct predecessor of $B$,
(1.2) $A$ is connected to $B$ via a hyperarc that has $A$ as the only tail node,
(1.3) the value of $A$ is not negated (for equivalence with $A = 1$)/negated (for equivalence with $A = 0$),
(1.4) the out-degree of $A$ is 1 (i. e. one hyperarc is pointing out of A),
(1.5) $A$ is not a target species, i. e. $A \notin T$.

The knock-out intervention (s-a-0 fault) $B = 0$ is equivalent to $A = 0$ (or to $A = 1$) with respect to the set of target species $T$ if the following conditions hold:

(2.1) A is a direct predecessor of B,
(2.2) the in-degree of B is 1 (i.e., one hyperarc is pointing into B),
(2.3) the value of A is not negated (for equivalence with $A = 0$)/negated (for equivalence with $A = 1$),
(2.4) the out-degree of A is 1,
(2.5) $A \notin T$.

Conditions 1.1–1.3 and 2.1–2.3 follow directly from the fact that an OR gate is controlled by any input set to 1, whereas an AND gate is controlled by any input set to 0. Condition 1.4/2.4 is needed to ensure that the upstream node ($A$) may not influence the target nodes via other paths not leading over $B$. A special case for s-a-0 faults in which we can find an equivalence relation between two adjacent nodes, although condition 2.4 is not met, arises if all hyperarcs pointing out of $A$ point into the same node $B$ (e.g., for the Boolean function $B = A \cdot C + A \cdot D$). The need for conditions 1.5/2.5 is based on the fact that an intervention at a certain node does not generally lead to the fulfillment of an intervention goal for a preceding species or reaction. For example, $\{B = 1\}$ is an MIS for the intervention goal $\{B = 1, O1 = 1\}$ in the toy model (Fig. 1). Ignoring condition 1.5 we find that $C = 1$ is equal to $B = 1$. However, $\{C = 1\}$ is not an MIS for our intervention goal as it does not lead to $B = 1$.

The rules described above detect the equivalence relations between interventions on the head and tail nodes of a hyperarc. Apart from that, we can also find equivalence relations between interventions on input nodes of the same gate. Some of these relations are indirectly detected by applying the rules above since pairwise equivalent interventions can be combined to equivalence classes. For example, if conditions 1.1–1.5 hold for the two inputs $A_1$, $A_2$ of an OR gate with output $B$, the equivalent interventions $B=1/A_1=1$ and $B=1/A_2=1$ are merged to the equivalence class $\{B=1,\ A_1=1,\ A_2=1\}$ reflecting also the equivalence between interventions at $A_1$ and $A_2$. Whereas for OR gates, all equivalence relations are detected in this way, s-a-0 faults at the inputs of an AND gate (hyperarc) can be equivalent although the equivalence with interventions at the output of the gate does not hold. This arises if condition 2.2 is not met, and we therefore have to consider the possible equivalence of interventions at the input nodes for this case separately: let $A_1, \ldots, A_k$ be the inputs of an AND gate and let $s_i$ denote whether the value of $A_i$ is negated ($s_i=1$) or not ($s_i=0$). It then holds that the interventions $A_i=s_i$ and $A_j=s_j$, $i,j \in \{1, \ldots, k\}$, are equivalent if (i) the out-degree of $A_i$ and $A_j$ is 1 and (ii) neither $A_i$ nor $A_j$ is a target species.

In the toy model (Fig. 1), the following classes of equivalent interventions with respect to an intervention goal containing O1 as the (only) target species can be found: $\{A=1, D=0\}$, $\{A=0, B=0, D=1\}$, $\{C=1, B=1\}$, $\{E=0,\ I2=0\}$, $\{G=1,\ F=1\}$, $\{G=0,\ F=0\}$. As we only consider one representative of each class of equivalent interventions as a possible element of an MIS, the number of candidates for single interventions is reduced by seven.

We emphasize that fault equivalence classes are specific for a given set of target species. One may exploit fault equivalence classes in combination with the dependency analysis: before computing the MISs for one specific intervention scenario, it is advantageous to first exploit the species interdependencies as described above and then, for the remaining interventions, to determine the fault equivalence classes further reducing the set of possible interventions. As an example, consider again the intervention goal $\{O1=1\}$ with I2 set to 1 as side constraint. As described in Section 4.1, we can restrict ourselves to considering the following single interventions by exploiting the interdependencies: $D=0$, $I1=0$, $I1=1$, $A=1$, $B=1$, $C=1$, $E=1$, $O1=1$. Of the six equivalence classes given above, only $\{A=1, D=0\}$ and $\{C=1, B=1\}$ contain no interventions that have already been excluded. In addition, considering the s-a-1 fault $C=1$ we not only find the equivalent intervention $B=1$ but now also $E=1$ as the interaction $E \rightarrow F$ was removed from the interaction graph by exploiting the species dependencies and therefore condition 1.3 is fulfilled. Thus, the number of single interventions from which MISs are combined in the main algorithm can be further decreased from 8 to 5 by taking into account the equivalent interventions. We finally achieved an enormous reduction of the (potential) search space from $3^{11}$ candidates to $3 \cdot 2^3$ candidates: in each MIS, A, C, and O1 are either set to 1 or not part of the MIS, whereas I1 can be set to 0 or 1 or is not part of the MIS.

### *4.3. Exploiting dependencies in intervention goals with multiple targets*

If an intervention goal includes more than one target species, we can further reduce the search space by not testing those combinations of interventions (MIS candidates) that do not influence all targets in the appropriate direction. In contrast to the reduction methods described in Sections 4.1 and 4.2, this is not done in preprocessing, but during the iterations: before we compute for an MIS candidate the resulting $LSS_3$ we check whether its interventions can influence all the target species with appropriate sign. Consider the intervention goal with target species $T^+=\{v_1\}$ and $T^-=\{v_2\}$. In this case, each intervention set must contain at least one intervention that has a positive influence on $v_1$ and one intervention that has a negative influence on $v_2$. MIS candidates that do not fulfill this condition will not be checked in the current iteration but will be combined with further interventions in the next iteration.

## 5. INTERVENTION PROBLEMS WITH MULTIPLE INTERVENTION SCENARIOS

There are two main applications for intervention problems with more than one intervention scenario. First, when searching for suitable interventions that provoke a desired response in a signaling network one might consider different environmental situations with altered external stimuli (here: different side constraints $C_j$) the cell is exposed to. Intervention problems with multiple intervention scenarios then allow to demand that each intervention set leads to the fulfillment of each intervention goal given its associated set of external stimuli.

A second application for using multiple scenarios is the identification of internal failure modes (diagnosis problem) or of inconsistent parts in the network structure: consider a signaling network that has been treated in several experiments with different combinations of input stimuli for which the resulting activation level has been measured (and then discretized) at certain nodes. Usually, we will have some discrepancies between measurements and model-based predictions. We might ask the following question: "Are there nodes whose states are permanently switched on or off leading to the observed discrepancies?" For answering this question, we build for each experiment an intervention scenario where the input stimuli (conditions) are the fixed nodes $x_i^0$, whereas the associated intervention goal is given by the respective measurements for this experiment. The computed intervention sets then indicate ways in which the model would have to be changed to perfectly reproduce the experimental results. They indicate either possible "natural" failures in the network (e.g., constitutively activated nodes/pathways in cancer cells) or wrong model assumptions leading to systematic prediction errors. In particular for the latter application, it might be useful to allow a certain number of errors regarding the fulfillment of the intervention goals—this issue will be briefly discussed in Section 6.

As in the case of intervention problems with one intervention scenario, the dependency matrix is used for search space reduction in two different ways (cf. Sections 4.1 and 4.3): (i) during pre-processing, one excludes those interventions that do not affect at least one target node in at least one intervention scenario. Clearly, as the number of species being a target in at least one intervention goal will be usually larger than in a single intervention scenario, the potential problem reduction is lower compared to the single scenario problem. (ii) During the algorithm, we check whether the current candidate MIS may influence all targets of all intervention goals with appropriate sign (cf. Section 4.3). Compared to the single scenario problem, this reduction technique will become more important due to the typically larger overall number of target species.

Regarding fault equivalence classes determined during preprocessing it is necessary to treat all those species as target nodes that can be activated or inactivated in at least one scenario (cf. Section 4.2).

The algorithm itself is then analogous to the algorithm used for single scenario problems (Section 3). The only difference is that one has to check for each candidate MIS the satisfaction of all intervention goals. The effort to test one MIS candidate thus increases linearly with the number of intervention scenarios.

We can now provide the scaffold of the full MIS algorithm in pseudo-code (Fig. 3).

## 6. FURTHER EXTENSIONS

The concept of MISs as described above can be extended in various ways:

a. Especially for intervention problems with multiple scenarios, it seems reasonable to allow a small number of errors regarding the desired values of the target nodes, i.e., to compute MISs that fulfill at least some parts of the intervention goal. For this purpose, one defines an upper boundary of permitted errors where the value of a target species needs not to be satisfied by the MIS.
b. Furthermore, we can remove some possible interventions a priori from the search space, for example to take into account that some species cannot (or should not) be knocked-in or knocked-out in the considered biological experiment. This includes in particular interventions at the target species itself as one is typically interested in finding alternative intervention strategies not including the trivial solutions. Another set of species for which we often exclude interventions are those for which a fixed value is given as side constraint. However, one can also think of certain problems where it is indeed useful to allow changes of the fixed states and we therefore referred to the most general case where the clamped values can be overwritten by an intervention when we introduced the MISs in Section 2.
c. To keep things simple, we restricted ourselves herein to considering target species rather than target reactions. Intervention goals that contain target reactions can be of interest if one does not want to block the activation of a species but one particular way of how it is activated (i.e., one of the hyperarcs pointing into the species). The concepts and reduction techniques introduced above can be extended to target reactions in a straightforward manner, for example by considering the influences on the input species of the target reactions when exploiting the dependencies for search space reduction as described in Section 4.1.

```
MIS = compute_mis(B,P,MAX_MIS_SIZE)
// INPUTS:
  // B: a Boolean network in LIH representation
  // P: an intervention problem for B
  // MAX_MIS_SIZE: maximum cardinality of the MISs to be computed
// OUTPUT:
  // MIS: the set of MISs
TARGETS_plus = all (target) nodes to be switched on in at least one intervention scenario in P;
TARGETS_minus = all (target) nodes to be switched off in at least one intervention scenario in P;
ALLOWED_INTERVENTIONS = dependency_analysis(B,P);
(ALLOWED_INTERVENTIONS,CLASSES) = fault_equ_classes(ALLOWED_INTERVENTIONS,B,P);
MIS = {};
MIS_CANDIDATES_NEW = {};
for i from 1 to MAX_MIS_SIZE
    MIS_CANDIDATES = MIS_CANDIDATES_NEW;
    MIS_CANDIDATES_NEW = {};
    for each M in MIS_CANDIDATES
        for each K in ALLOWED_INTERVENTIONS
            M_NEW = M ∪ K;
            if M_NEW is support-minimal w.r.t. MIS and MIS_CANDIDATES_NEW
                // checks that M_NEW does not contain any mis from MIS or any mis_candidate
                // from MIS_CANDIDATES_NEW
                if M_NEW can affect all TARGETS_plus and all TARGETS_minus in required directions
                    GOALS_OK = true;
                    for each intervention scenario S in P
                        LSS3 = computeLSS3(M_NEW,B,S); // S required for setting side constraints
                        if LSS3 does not fulfill the intervention goal of S
                            GOALS_OK = false;
                            break;
                        endif
                    endfor
                    if GOALS_OK
                        Append M_NEW to MIS;
                    else
                        Append M_NEW to MIS_CANDIDATES_NEW;
                    endif
                endif
            endif
        endfor
    endfor
endfor
MIS = expand(MIS,CLASSES);
END
```

**FIG. 3.** Pseudo-code for the computation of minimal intervention sets (MISs).

d. Our logical framework can be extended in several ways to obtain a more appropriate description of real signaling networks. One possibility is to introduce an additional logical operator with incomplete truth table (ITT gate) that returns 1 if and only if all its non-negated arguments are 1 and all negated arguments are 0 and that returns 0 if all non-negated arguments are 0 and all negated arguments are 1 (Klamt et al., 2006). In all other cases, the operator returns the unknown value (*). The ITT gate can therefore be used to cope with uncertainties regarding the logical combination of different influences on a certain node (whether to use an AND or an OR gate). Obviously, the usage of ITT gates reduces the number of states that can be determined to be 1 or 0 when computing the $LSS_3$. For the computation of MISs this means that one usually has to intervene at more points of the network to fulfill the intervention goal safely and thus a higher computational effort is needed. Furthermore, no equivalence relations as described for AND and OR gates in Section 4.2 hold for the ITT gates leading to an additional increase of the computation time. In return, using ITT gates allows a safer interpretation of the results and makes it possible to detect properties of the network that are independent of the assumptions we make when choosing a logical function. In the extreme case, computing MISs in a network where in each node all inputs are combined by an ITT gate will reveal those sets of interventions that fulfill the intervention goal irrespective of the real Boolean (or even kinetic) concatenations. For example, in the toy model, the incoming edges in B, E, C and F would be concatenated by ITT gates. One MIS to enforce O1 to be 1 in this network would be $\{B = 1, E = 1\}$; only one of the two contained interventions would suffice for the original toy model.

## 7. BENCHMARKS

The calculation of MISs has been implemented in the *CellNetAnalyzer* (CNA), a MATLAB-toolbox that, among other things, can be used for the qualitative and logical analysis of signaling networks including the computation of MISs (Klamt et al., 2007). CNA has a graphical user interface that can be used to conveniently set up single-scenario intervention problems. In addition, CNA provides an application programming interface (API) so that MISs can be calculated directly without requiring user interaction; with this API, multiple scenario problems can be handled, which constitutes a major extension of CNA's functionality. All the extensions listed in Section 6 are also implemented both in the GUI and API.

CNA is freely available for academic use at www.mpi-magdeburg.mpg.de/projects/cna/cna.html.

In this section, we illustrate the power of the introduced reduction techniques for the computation of MISs by intervention problems related to concrete biological questions. The logical models used for these benchmarks are recently published models of T cell receptor signaling (Saez-Rodriguez et al., 2007) and of signaling cascades induced by the EGF receptor family (Samaga et al., 2009). Both models were implemented and analyzed in CNA.

For the T cell receptor model, we consider the intervention problem as described in Saez-Rodriguez et al. (2007), hereafter referred to as the "TCR problem": we search for MISs (with a maximum cardinality of 3) that lead to sustained T cell activation without external stimuli. The target species (9 to be activated, 3 to be inhibited) cover the complete output layer of the model, i.e., each non-output node influences at least one target node, whereas the three main inputs are set to zero as side constraints. The intervention problem we consider for the EGF receptor model (the "EGFR problem") is to activate two central players controlling proliferation and apoptosis (the kinases Akt and ERK) without external stimuli. Again, we restrict the maximum size of the MISs to 3. A detailed description of the intervention problems and resulting MISs can be found in the Supplementary Material (see online Supplementary Material at www.liebertonline.com).

Table 2 shows the results of the reduction techniques, together with the achieved reduction in computation time. We first consider the results when applying each reduction technique separately. All three techniques yield a decrease of the computation time by at least 49%, up to 99%. In the EGFR problem, in contrast to the TCR problem, the target species (Akt and ERK) are both situated in the intermediate layer of the EGFR network and many species or even whole pathways in the EGFR problem do not affect the target species Akt or ERK. According to these different structures of the intervention problems, the methods for search space reduction yield varying profit (Table 2). For the EGFR problem, exploiting the interdependencies is very beneficial: the number of possible single interventions can be reduced from 138 to 38 when only those single interventions are permitted that influence the target species properly (i.e., help activating Akt or/and ERK). In comparison, in the TCR problem, dependency analysis reduces the 130 possible single interventions only by 36 down to 94. In both models, we find a similar number of equivalence classes each containing at least two equivalent interventions (19 in TCR, 20 in EGFR); however, the equivalence classes for TCR are larger leading to a higher number of single interventions that can be excluded in the main algorithm. In most cases, the number of tested MIS candidates (for which the $LSS_3$ was calculated) is higher for the TCR problem, although more single interventions are possible in EGFR. This is due to the fact that many MISs of cardinality 1 and 2 exist for the EGFR problem, decreasing the search space considerably since supersets of these MISs need not to be considered in later iterations. In contrast, no MIS with cardinality smaller than 3 exists in the TCR problem.

When combining the three reduction techniques, as already mentioned above, we first exploit the dependencies and then compute the equivalence classes in the reduced model providing the final set of

TABLE 2. BENCHMARKS

| | | *Exploiting* | | | | |
|---|---|---|---|---|---|---|
| | *Naive algorithm* | *sDep* | *EqClass* | *mDep* | *sDep and EqClass* | *Algorithm exploiting all* |
| **EGFR problem** | | | | | | |
| Computation time [sec] | 750 | 4 | 307 | 257 | 2 | 2 |
| (% of naive algorithm) | (100%) | (0.5%) | (41%) | (34%) | (0.3%) | (0.3%) |
| Checked candidates | 302975 | 1951 | 122817 | 93483 | 970 | 960 |
| Possible single interventions | 138 | 38 | 105 | 138 | 32 | 32 |
| Species that can | | | | | | |
| be set to 1 and 0 | 69 | 11 | 42 | 69 | 10 | 10 |
| only be set to 1 | 0 | 15 | 14 | 0 | 11 | 11 |
| only be set to 0 | 0 | 1 | 7 | 0 | 1 | 1 |
| Single interventions excluded | | | | | | |
| by sDep | — | 100 | — | — | 100 | 100 |
| by EqClass (# EqClass) | — | — | 33 (20) | — | 6 (4) | 6 (4) |
| Candidates excluded by mDep | — | — | — | 209492 | — | 10 |
| **TCR problem** | | | | | | |
| Computation time [sec] | 655 | 249 | 212 | 336 | 91 | 66 |
| (% of naive algorithm) | (100%) | (38%) | (32%) | (51%) | (15%) | (10%) |
| Checked candidates | 357890 | 135626 | 114577 | 169189 | 49061 | 34991 |
| Possible single interventions | 130 | 94 | 89 | 130 | 67 | 67 |
| Species that can | | | | | | |
| be set to 1 and 0 | 65 | 31 | 34 | 65 | 17 | 17 |
| only be set to 1 | 0 | 27 | 15 | 0 | 26 | 26 |
| only be set to 0 | 0 | 5 | 6 | 0 | 7 | 7 |
| Single interventions excluded | | | | | | |
| by sDep | — | 36 | — | — | 36 | 36 |
| by EqClass (# EqClass) | — | — | 41 (19) | — | 27 (13) | 27 (13) |
| Candidates excluded by mDep | — | — | — | 188701 | — | 14070 |
| **EGFR multiple scenarios problem** | | | | | | |
| Computation time [sec] | 899 | 260 | 453 | 677 | 174 | 131 |
| (% of naive algorithm) | (100%) | (29%) | (50%) | (75%) | (19%) | (15%) |
| Checked candidates | 375447 | 79587 | 188865 | 168307 | 53041 | 38076 |
| Possible single interventions | 132 | 79 | 105 | 132 | 69 | 69 |
| Species that can | | | | | | |
| be set to 1 and 0 | 61 | 34 | 40 | 61 | 26 | 26 |
| only be set to 1 | 5 | 6 | 15 | 5 | 7 | 7 |
| only be set to 0 | 5 | 5 | 10 | 5 | 10 | 10 |
| Single interventions excluded | | | | | | |
| by sDep | — | 53 | — | — | 53 | 53 |
| by EqClass (# EqClass) | — | — | 27 (18) | — | 10 (8) | 10 (8) |
| Candidates excluded by mDep | — | — | — | 207140 | — | 14965 |

All computations were made on an Intel$^{(R)}$Core$^{(TM)}$2 Duo CPU E6850 at 3.00 GHz.

sDep, dependency analysis to exclude single interventions; mDep, dependency analysis to exclude MIS candidates not influencing all targets; EqClass, fault equivalence classes; MIS, minimal intervention set.

permitted single interventions. During the main algorithm, the number of MIS candidates that are checked (i.e., for which the $LSS_3$ is computed) is then further reduced by excluding those candidate MIS whose interventions do not influence all target species properly (see Section 4.3). Both problems benefit when incorporating the equivalence classes, for the TCR problem even more than for the EGFR problem, because in the latter many interventions were already excluded by dependency analysis. Exploiting the dependencies for multiple targets finally leads to an appreciable improvement only for the TCR problem—what can be explained by the fact that there are many target species in this setting.

We complete the discussion of the introduced improvements by studying an intervention problem with multiple intervention scenarios (Table 2). As mentioned in Section 5, the concept of MISs can be used to identify inconsistencies in the network structure that account for discrepancies between model predictions and measurements (which can be seen as an effect-cause analysis or diagnosis problem). To illustrate this, we installed manually three failures in the EGF receptor model (we fixed the state of three species) and generated by this "malfunctioning" network a synthetic data set that mimics the results of 34 experiments where the state of 12 species of the network (on or off) was measured in response to 34 different combinations of ligands and inhibitors. We thus consider an intervention problem with 34 intervention scenarios where the intervention goal for each scenario is that the 12 species reach the measured state and the side constraint is given by the stimulus (ligand/inhibitor). Searching for MISs that solve this intervention problem in the original network shows which changes of the network structure can explain the (artificial) experimental data. We found six MISs (up to size 3), among them the one that contains the interventions we installed as failures in the model (see online Supplementary Material at www.liebertonline.com). The structure of the intervention problem—albeit with multiple intervention scenarios—is similar to the TCR problem: we have a large number of target species that are situated in the lower part of the network. Thus, the relative decrease of the computation time that can be achieved by the three reduction techniques is similar to the TCR problem.

In conclusion, the examples show that the methods for search space reduction lead to a considerable decrease of the computation time, whereby the actual benefit and the contribution of each of the three techniques is dependent on the structure of the intervention problem as well as the network structure itself: for the EGFR problem, we could reduce the computation time by 99.5% compared to the naive algorithm without any reduction, whereas for the other two examples at least 85% reduction could be achieved. Even for the complicated multiple scenario problem in the EGFR network, all MISs up to size 3 could thus be calculated in less than 3 minutes on a typical PC.

## 8. DISCUSSION

Minimal intervention sets can be used to gain important insights into signaling models that are based on a Boolean description of signaling and regulatory networks. They can be used for drug target identification, to design experiments leading to a desired result, or for diagnosis, i.e., for generating hypotheses that explain experimental data in the context of a given network (Saez-Rodriguez et al., 2007; Franke et al., 2008; Samaga et al., 2009).

Due to the complex combinatorial problem, determination of MISs is computationally challenging what calls for algorithmic improvements. In this study, we introduced three search space reduction techniques (dependency analysis for single interventions and multiple targets, fault equivalence classes) that may significantly speed up the calculation procedure. As the benchmark demonstrated, the achievable performance gain as well as the contributions of the different reduction techniques depends on the actual intervention problem and network size. The computation times for typical intervention problems show that realistic problems can be treated in reasonable time. We have implemented the procedure for computing MISs (GUI- and API-based) with various options in our software tool CellNetAnalyzer, which is publicly available.

With fault equivalence classes, we adopted a well-known technique from electrical engineering illustrating relationships between electrical circuits and cellular networks with information flow. The concept of fault equivalence classes is not only relevant for computing MISs: they also reveal sets of failures in the network that are indistinguishable with a given set of readouts. The importance of the dependency matrix for elucidating network-wide activatory/inhibitory interdependencies, not only in the context of MISs, has been shown in previous studies (Saez-Rodriguez et al., 2007; Franke et al., 2008; Samaga et al., 2009).

The framework of LIHs used herein is based on Boolean networks where nodes are either on or off. In the future, it is planned to extend the framework to multi-valued logic where the "on" state can be differentiated into several levels, which can, for example, represent different levels of activation. The $LSS_3$ calculation can be extended in a straightforward manner to support multiple values. Obviously, the computational effort for the determination of MISs increases dramatically if more than two states have to be considered for each node. Therefore, techniques for improving the algorithms will be even more essential than in the Boolean case. The concepts we introduced herein can, in large part, be adapted to multi-valued logic. However, whether these efforts then suffice to compute MISs in large-scale multi-valued logical networks in suitable time remains to be seen.

## ACKNOWLEDGMENTS

We are grateful to Oliver Hädicke for helpful comments during the preparation of the manuscript. This work was supported by the German Federal Ministry of Education and Research (HepatoSys and FORSYS-Centre MaCS [Magdeburg Centre for Systems Biology]) and the Ministry of Education and Research of Saxony-Anhalt (Research Center "Dynamic Systems").

## DISCLOSURE STATEMENT

No competing financial interests exist.

## REFERENCES

Abramovici, M., Breuer, M.A., and Friedman, A.D. 1990. *Digital Systems Testing and Testable Design*. Computer Science Press, New York.

Albert, R., and Othmer, H.G. 2003. The topology of the regulatory interactions predicts the expression pattern of the segment polarity genes in *Drosophila melanogaster. J. Theor. Biol.* 223, 1–18.

Christensen, T.S., Oliveira, A.P., and Nielsen, J. 2009. Reconstruction and logical modeling of glucose repression signaling pathways in *Saccharomyces cerevisiae. BMC Syst. Biol.* 3, 7.

Fitzgerald, J.B., Schoeberl, B., Nielsen, U.B., et al. 2006. Systems biology and combination therapy in the quest for clinical efficacy. *Nat. Chem. Biol.* 2, 458–466.

Franke, R., Müller, M., Wundrack, N., et al. 2008. Host-pathogen systems biology: logical modelling of hepatocyte growth factor and *Helicobacter pylori* induced c-Met signal transduction. *BMC Syst. Biol.* 2, 4.

Kauffman, S.A. 1969. Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.* 22, 437–467.

Klamt, S. 2006. Generalized concept of minimal cut sets in biochemical networks. *Biosystems* 83, 233–247.

Klamt, S., Saez-Rodriguez, J., Lindquist, J.A., et al. 2006. A methodology for the structural and functional analysis of signaling and regulatory networks. *BMC Bioinform.* 7, 56.

Klamt, S., Saez-Rodriguez, J., and Gilles, E.D. 2007. Structural and functional analysis of cellular networks with CellNetAnalyzer. *BMC Syst. Biol.* 1, 2.

Klamt, S., and von Kamp, A. 2009. Computing paths and cycles in biological interaction graphs. *BMC Bioinformatics* 10, 181.

McCluskey, E.J., and Clegg, F.W. 1971. Fault equivalence in combinational logic networks. *IEEE Trans. Comput.* C20, 1286–1293.

Mendoza, L., Thieffry, D., and Alvarez-Buylla, E.R. 1999. Genetic control of flower morphogenesis in Arabidopsis thaliana: a logical analysis. *Bioinformatics* 15, 593–606.

Saez-Rodriguez, J., Simeoni, L., Lindquist, J.A., et al. 2007. A logical model provides insights into T cell receptor signaling. *PLoS Comput. Biol.* 3, e163.

Sahin, O., Fröhlich, H., Löbke, C., et al. 2009. Modeling ERBB receptor-regulated G1/S transition to find novel targets for de novo trastuzumab resistance. *BMC Syst. Biol.* 3, 1.

Samaga, R., Saez-Rodriguez, J., Alexopoulos, L.G., et al. 2009. The logic of EGFR/ErbB signaling: theoretical properties and analysis of high-throughput data. *PLoS Comput. Biol.* 5:e1000438.

Thomas, R. 1973. Boolean formalization of genetic control circuits. *J. Theor. Biol.* 42, 563–585.

Thomas, R., and D'Ari, R. 1990. *Biological Feedback.* CRC Press, Boca Raton, FL.

Address correspondence to:
*Dr. Steffen Klamt*
*Max Planck Institute for Dynamics of Complex Technical Systems*
*Sandtorstraße 1*
*D-39106 Magdeburg, Germany*

*E-mail:* klamt@mpi-magdeburg.mpg.de