Roland Kaminski | Torsten Schaub | Anne Siegel | Santiago Videla

Minimal intervention strategies in logical signaling networks with ASP

Suggested citation referring to the original publication:
Theory and Practice of Logic Programming 13 (2013) 4-5, pp. 675–690
DOI https://doi.org/10.1017/S1471068413000422
ISSN (print) 1471-0684
ISSN (online) 1475-3081

TLP **13** (4–5): Online Supplement, July 2013.

URL: http://dx.doi.org/10.1017/S1471068413000422

Minimal intervention strategies in logical signaling networks with ASP

ROLAND KAMINSKI
University of Potsdam

TORSTEN SCHAUB
University of Potsdam

ANNE SIEGEL
CNRS - IRISA, Rennes INRIA - Dyliss, Rennes

SANTIAGO VIDELA
CNRS - IRISA, Rennes INRIA - Dyliss, Rennes University of Potsdam

submitted 10 April 2013; revised 23 June 2013; accepted 5 July 2013

Abstract

Proposing relevant perturbations to biological signaling networks is central to many problems in biology and medicine because it allows for enabling or disabling certain biological outcomes. In contrast to quantitative methods that permit fine-grained (kinetic) analysis, qualitative approaches allow for addressing large-scale networks. This is accomplished by more abstract representations such as logical networks. We elaborate upon such a qualitative approach aiming at the computation of minimal interventions in logical signaling networks relying on Kleene's three-valued logic and fixpoint semantics. We address this problem within answer set programming and show that it greatly outperforms previous work using dedicated algorithms.

1 Introduction

Systems biology is a field at the crossover of biology, informatics, and mathematics. It aims at developing methods and models to elucidate the functioning of biological systems. Among them, signaling networks are crucial for the understanding of the fast response of a system to external perturbations. Importantly, they are involved in bio-medical processes and their control has a crucial impact on drug target identification and diagnosis.

During the last decade, many efforts have been made to develop relevant formalisms and modeling frameworks to take into account the specificities of such biological systems. In the lack of quantitative details, qualitative approaches, such as Boolean logical networks (Kauffman 1969; Thomas 1973), have become very popular (Wang *et al.* 2012). It has been proved that the early response of

signaling networks can be appropriately modeled with Boolean logical networks, as illustrated on several signal transduction pathways involved in diverse processes such as proliferation, cell cycle regulation, apoptosis, or differentiation (Saez-Rodriguez *et al.* 2007; Samaga *et al.* 2009; Calzone *et al.* 2010; Saez-Rodriguez *et al.* 2011). Moreover, inference of Boolean networks from experimental data can now be performed with several computational methods (Mitsos *et al.* 2009; Saez-Rodriguez *et al.* 2009; Sharan and Karp 2012; Videla *et al.* 2012).

As mentioned in the seminal paper for systems biology (Kitano 2002), a major challenge in this field is how to systematically control the state of the cell. From an application viewpoint, this means selecting appropriate drugs in order to force the system to reach a steady state with properties that were specified a priori. Thus, progress in this area would lead to hypothesis-driven research in biology. Nowadays, due to the lack of information, multiple hypotheses are usually generated from prior knowledge and computational models. Next, decision-making methods can be used to suggest new experiments in order to reduce ambiguous hypotheses. Finally, new experimental data is produced to test the generated hypotheses, the models are refined, and the loop is started over again (Kitano 2002; Kreutz and Timmer 2009; Sparkes *et al.* 2010).

The problem of identifying "key-players" in biological systems has been addressed for metabolic, gene and signaling networks. However, the underlying mathematical formalisms for each of these biological networks allows for different computational approaches. Moreover, while significant progress has been made for metabolic (Stelling *et al.* 2002; Kauffman *et al.* 2003; Klamt 2006; Acuña *et al.* 2012) and gene regulatory networks (Faryabi *et al.* 2008; Karlebach and Shamir 2009; Bouaynaya *et al.* 2011), controlling mechanisms in signaling networks remain poorly understood and only a few approaches can be found in the literature (Abdi *et al.* 2008; Samaga *et al.* 2010; Wang and Albert 2011). More precisely, apart from numerical methods (e.g. ordinary differential equations), computational formalisms for metabolic networks are based on linear algebra whereas the dynamics in gene regulatory networks are well captured either by probabilistic approaches or by discrete dynamical systems taking into account non-linear effects (Batt *et al.* 2008; Naldi *et al.* 2009). On the contrary, the nature of signaling transduction networks is closely related to that of digital circuits (Abdi *et al.* 2008; Morris *et al.* 2010; Wang and Buck 2012). Thus, logic-based models are particularly well suited to describe and study the (early)-response of such systems.

Among the few approaches addressing the challenge of controlling the state of the cell in the context of signaling transduction, we focus in what follows on (Samaga *et al.* 2010). Based on earlier work (Klamt 2006) on metabolic networks, the notion of *minimal intervention sets* was introduced in Samaga *et al.* (2010) and dedicated algorithms were developed to compute them. Intuitively, an "intervention" consists of an inclusion minimal set of knock-ins (activation drugs) and knock-outs (inhibition drugs) that force a set of target species or compounds into a desired state. Unfortunately, the dedicated algorithms are computationally demanding due to the highly combinatorial mechanisms in signaling networks. Therefore, they are limited to compute small intervention sets and fail to scale over large-scale

networks. In general, multiple interventions are necessary to cope with robustness and cellular complexity (Stelling *et al.* 2004). Moreover, previous work on the inference of logical networks suggests that, if the inherent noise is considered, there are multiple networks compatible with the experimental observations (Saez-Rodriguez *et al.* 2009). Concretely, the mentioned limitations make it hard to prove that the identified solutions are biologically robust to small perturbations of the system or its environment. Thus, in order to overcome such limitations, more elaborate and more powerful computational methods are needed towards large-scale systems and robust solutions.

The question of scalability of computational methods for the identification or the pruning of biological systems has already been successfully addressed with Answer Set Programming (ASP; (Baral 2003)) in various settings, among them (Baral *et al.* 2004; Erdem and Türe 2008; Gebser *et al.* 2010; Ray *et al.* 2010; Gebser *et al.* 2011; Videla *et al.* 2012). Of special interest is here ASP's expressive power to address problems of elevated complexity, in particular, for computing inclusion minimal models. In the same direction as these papers, we provide a precise characterization of the minimal intervention set problem relying on Kleene's three-valued logic and fixpoint semantics (similar to that of Fitting (1985)). We introduce an ASP encoding to solve this problem and we evaluate its performance on four real-world and biologically relevant benchmarks. [1] Interestingly, our fixpoint based characterisation of biological network response is in line with the investigations of Inoue (2011; Inoue and Sakama (2012) using different logic programming semantics for characterising systems behaviour. To be more precise, Inoue uses in Inoue (2011) a two-valued logic along with two-valued fixpoint semantics to characterise trajectories and stable states of Boolean networks (by translating them into logic programs). Hence such logic programming concepts appear to be appropriate tools for characterising these types of networks. Moreover, Boolean constraint solving technologies offer a powerful computational framework to face the highly combinatorial nature of signaling networks.

In what follows, we assume some familiarity with ASP, its semantics as well as its basic language constructs. A comprehensive treatment of ASP can be found in Baral (2003; Gebser *et al.* (2012). Our encodings are written in the input language of *gringo* 3 (Gebser *et al.*).

2 Intervention set strategies

This section provides a formal characterization of intervention strategies, as treated in Samaga *et al.* (2010). In doing so, we follow the popular approach to qualitative modeling in biology (Kauffman 1969; Thomas 1973) in representing biological networks as logical networks[2] and associating biological species or compounds (eg. receptors, kinases or phosphatases) with propositional variables. More formally, a

[1] The encodings are available at: `http://potassco.sourceforge.net/apps.html#interventions`
[2] We refrain from using the term *Boolean network* in view of our usage of three-valued semantics.

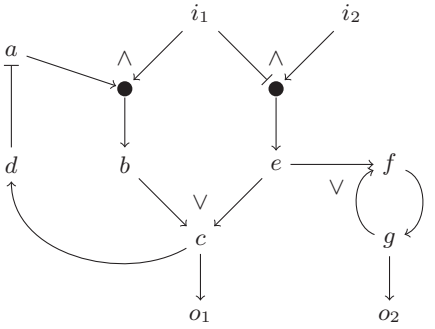

Fig. 1. Exemplary directed hypergraph representation of a logical network. We refer the reader to Klamt *et al.* (2009) for more details on hypergraphs and cellular networks.

logical network consists of a finite set V of propositional variables and a function ϕ mapping each variable $v \in V$ to a propositional formula $\phi(v)$ over V. We form propositional formulas from V with the connectives $\bot$, $\top$, $\neg$, $\vee$, and $\wedge$ in the standard way. For illustration, let us consider the logical network reproduced from (Samaga *et al.* 2010) in Figure 1. This network consists of the set V of species variables $\{i_1, i_2, a, \ldots, g, o_1, o_2\}$ along with the function ϕ defined as:

$$\phi = \left\{ \begin{array}{lll} i_1 \mapsto i_1 & a \mapsto \neg d & d \mapsto c \qquad g \mapsto f \\ i_2 \mapsto i_2 & b \mapsto a \wedge i_1 & e \mapsto \neg i_1 \wedge i_2 \quad o_1 \mapsto c \\ & c \mapsto b \vee e & f \mapsto e \vee g \qquad o_2 \mapsto g \end{array} \right\}$$

Note that ϕ leaves the specification of the (input) variables i_1 and i_2 open.

The steady states of such a network are given by truth assignments yielding identical values for v and $\phi(v)$ for all $v \in V$ (although not all are biological meaningful as we see below). Following (Samaga *et al.* 2010), we adapt a three-valued setting relying on Kleene's three-valued logic (Kleene 1950) and thus consider truth assignments mapping formulas to truth values $\{\boldsymbol{t}, \boldsymbol{f}, \boldsymbol{u}\}$ according to Kleene's semantics. Clearly, two-valued assignments are restricted to range $\{\boldsymbol{t}, \boldsymbol{f}\}$. Observe that any logical network has a trivial three-valued steady state obtained by assigned $\boldsymbol{u}$ to all variables. In fact, the major constituents of intervention strategies can be captured by means of partial two-valued assignments (designating the absence or presence of certain species). We sometimes represent assignments extensionally as sets, viz. $\{v \mapsto A(v) \mid v \in V\}$, for checking containment, difference, etc. To avoid conflicts when composing assignments, we define $A \circ B = (A \setminus \overline{B}) \cup B$ where $\overline{B} = \{v \mapsto \overline{s} \mid v \mapsto s \in B\}$ and $\overline{\boldsymbol{t}} = \boldsymbol{f}$, $\overline{\boldsymbol{f}} = \boldsymbol{t}$, $\overline{\boldsymbol{u}} = \boldsymbol{u}$.

For capturing the dynamics of a logical network (V, ϕ), we define the following operator on truth assignments over V: [3]

$$\Omega_{(V,\phi)}(A) = \{v \mapsto A(\phi(v)) \mid v \in V\}.$$

[3] The interested reader may notice the resemblance to Fitting's three-valued operator (Fitting 1985).

Among the (three-valued) steady states of (V, ϕ), we are interested in the fixpoint of $\Omega_{(V,\phi)}$ reachable from the "undefined assignment". As with Fitting's operator (1985), this fixpoint is unique and can be computed in polynomial time.

To this end, we define the iterative variant of $\Omega_{(V,\phi)}$ as

$$\Omega^0_{(V,\phi)}(A) = A \quad \text{and} \quad \Omega^{j+1}_{(V,\phi)}(A) = \Omega_{(V,\phi)}\big(\Omega^j_{(V,\phi)}(A)\big) \ .$$

In biological terms, a sequence $(\Omega^j_{(V,\phi)}(A))_{j\in J}$ represents the evolution of a system starting in state A.

For capturing modifications to a logical network, (Samaga *et al.* 2010) puts forward the notion of *clamping* variables to Boolean values, thereby overriding their original specification: The logical network $(V, \phi|_A)$ is obtained from (V, ϕ) by clamping assignment A over V, if

$$\phi|_A(v) = \begin{cases} \top & \text{if } A(v) = \boldsymbol{t} \\ \bot & \text{if } A(v) = \boldsymbol{f} \\ \phi(v) & \text{otherwise} \end{cases}$$

Given a network, the aim of an *intervention strategy* is to identify an intervention that leads to a steady state satisfying a given goal under some side constraints. The concepts of an *intervention* (I), *goal* (G), and side *constraints* (C) can be captured as partial two-valued assignments (indicating the absence or presence of certain species).

To be more precise, given a logical network (V, ϕ), an *intervention scenario* is a pair (G, C) of partial two-valued assignments over V and an *intervention set* is a partial two-valued assignment I over a set of intervention variables $X \subseteq V$.

Intervention Strategy

Let (V, ϕ) be a logical network, let (G, C) be an intervention scenario, and $X \subseteq V$ be a set of intervention variables.

An intervention set I over X is an intervention strategy for (G, C) wrt (V, ϕ), if for some $j \geqslant 0$, we have

1. $\Omega^j_{(V,\phi|_{C\circ I})}(S_u) = \Omega^{j+1}_{(V,\phi|_{C\circ I})}(S_u)$ *where* $S_u = \{v \mapsto \boldsymbol{u} \mid v \in V\}$ *and*
2. $G \subseteq \Omega^j_{(V,\phi|_{C\circ I})}(S_u)$

In words, $\Omega^j_{(V,\phi|_{C\circ I})}(S_u)$ is a steady state of the clamped network $(V, \phi|_{C\circ I})$ satisfying the goal conditions in G. The important biological property of $\Omega^j_{(V,\phi|_{C\circ I})}(S_u)$ is that each of its variables remains undefined unless there is a cause to make it true or false.

The *intervention set problem* consists in deciding whether there is an intervention strategy for an intervention scenario (G, C) wrt a logical network (V, ϕ). Roughly speaking, the intervention set problem is a typical problem in NP: Once an intervention is guessed, it can be verified in polynomial time.

As an example, consider the above logical network along with the intervention scenarios $(G_1, C_1) = (\{o_1 \mapsto \boldsymbol{f}, o_2 \mapsto \boldsymbol{t}\}, \{i_1 \mapsto \boldsymbol{t}\})$ and $(G_2, C_2) = (\{a \mapsto \boldsymbol{t}\}, \emptyset)$. The intervention set $\{b \mapsto \boldsymbol{f}, e \mapsto \boldsymbol{f}, f \mapsto \boldsymbol{t}\}$ satisfies both scenarios yielding the two

steady states, respectively:

$$\{i_1 \mapsto \boldsymbol{t}, i_2 \mapsto \boldsymbol{u}, a \mapsto \boldsymbol{t}, b \mapsto \boldsymbol{f}, c \mapsto \boldsymbol{f}, d \mapsto \boldsymbol{f}, e \mapsto \boldsymbol{f}, f \mapsto \boldsymbol{t}, g \mapsto \boldsymbol{t}, o_1 \mapsto \boldsymbol{f}, o_2 \mapsto \boldsymbol{t}\}$$
$$\{i_1 \mapsto \boldsymbol{u}, i_2 \mapsto \boldsymbol{u}, a \mapsto \boldsymbol{t}, b \mapsto \boldsymbol{f}, c \mapsto \boldsymbol{f}, d \mapsto \boldsymbol{f}, e \mapsto \boldsymbol{f}, f \mapsto \boldsymbol{t}, g \mapsto \boldsymbol{t}, o_1 \mapsto \boldsymbol{f}, o_2 \mapsto \boldsymbol{t}\}.$$

Now, let us define further intervention strategies relying on a finite family $(G_j, C_j)_{j\in J}$ of intervention scenarios and k some positive integer.

- A *multi-scenario intervention strategy* for $(G_j, C_j)_{j\in J}$ wrt (V, ϕ) is an intervention strategy for each (G_j, C_j) wrt (V, ϕ) for each $j \in J$.
- A *bounded intervention strategy* for $(G_j, C_j)_{j\in J}$ wrt (V, ϕ) and k is a multi-scenario intervention strategy for $(G_j, C_j)_{j\in J}$ wrt (V, ϕ) of cardinality $k' \leqslant k$.
- A *minimal bounded intervention strategy* for $(G_j, C_j)_{j\in J}$ wrt (V, ϕ) and k is a $\subseteq$-minimal multi-scenario intervention strategy for $(G_j, C_j)_{j\in J}$ wrt (V, ϕ) of cardinality $k' \leqslant k$.

In the following, we are particularly interested in enumerating all minimal bounded intervention strategies. [4] We apply two different approaches to ASP solving. The first is *claspD*, which can enumerate all subset minimal solutions in polynomial space. The second one is *hclasp*, which utilizes (heuristic-driven) solution recording and hence runs in exponential space. Given that the intervention set problem has a potentially exponential number of solutions, both algorithms run in exponential time. However, by using *claspD*, it is possible to ask more complex queries. We can in principle put additional constraints on the solution candidates enumerating only subset minimal solutions that have a certain property. This is not possible with *hclasp* without embedding it into another algorithm (and then not in polynomial space).

3 Encoding

3.1 Instance representation

Let (V, ϕ) be a logical network. We represent the variables V as facts over predicate `variable/1`, namely `variable(`v`)` for all $v \in V$. Facts over predicate `candidate/1` denote the intervention variables that can be part of an intervention set. This allows us to control on which species interventions are permitted, for example one can exclude interventions over constrained or goal variables.

Without loss of generality, we assume that all formulas mapped by ϕ are in disjunctive normal form. Hence, $\phi(v)$ is a set of (dual) clauses and a clause a set of literals. We represent formulas using predicates `formula/2`, `dnf/2`, and `clause/3`. The facts `formula(`v`,`$s_{\phi(v)}$`)` map variables $v \in V$ to their corresponding formulas $\phi(v)$, facts `dnf(`$s_{\phi(v)}$`,`s_c`)` associate $\phi(v)$ with its clauses $c \in \phi(v)$, facts `clause(`s_c`,`v`,1)` associate clause c with its positive literals $v \in c \cap V$, and facts `clause(`s_c`,`v`,-1)` associate clause c with its negative literals $\neg v \in c$. Note that each $s_{(\cdot)}$ stands for some arbitrary but unique name in its respective context here.

[4] In Samaga *et al.* (2010), minimal bounded intervention strategies are called *minimal intervention sets.*

Finally, we represent the set of scenarios (G_i, C_i) for $1 \leqslant i \leqslant n$ using predicates `scenario/1`, `goal/3`, and `constrained/3`. The facts `scenario(`i`)` for $1 \leqslant i \leqslant n$ denote the scenarios to consider. The facts `goal(`i`,`v`,1)` and `constrained(`i`,`w`,1)` for positive goal literals $v \in G_i \cap V$ and positive constrained literals $w \in C_i \cap V$ and facts `goal(`i`,`v`,-1)` and `constrained(`i`,`w`,-1)` for negative goal literals $\neg v \in G_i$ and negative constrained literals $\neg w \in C_i$ denote the respective intervention goals and side constraints in each scenario.

Listing 1 shows the instance representation of our toy example logical network in Figure 1 together with the two intervention scenarios $(G_1, C_1) = (\{o_1 \mapsto \boldsymbol{f}, o_2 \mapsto \boldsymbol{t}\}, \{i_1 \mapsto \boldsymbol{t}\})$ and $(G_2, C_2) = (\{a \mapsto \boldsymbol{t}\}, \emptyset)$.

Listing 1. Toy example problem instance

```
variable(i1). variable(i2). variable(o2). variable(o1). variable(a).
variable(b).  variable(c).  variable(d).  variable(e).  variable(f).
variable(g).

candidate(i2). candidate(b).  candidate(c).  candidate(d).
candidate(e).  candidate(f).  candidate(g).

formula(a,0). formula(b,2). formula(c,1).  formula(d,4). formula(e,3).
formula(f,6). formula(g,5). formula(o1,4). formula(o2,7).

dnf(0,5). dnf(1,6). dnf(1,0). dnf(2,3). dnf(3,7).
dnf(4,1). dnf(5,2). dnf(6,4). dnf(6,6). dnf(7,4).

clause(0,b,1).  clause(1,c,1).  clause(2,f,1).  clause(3,a,1).
clause(3,i1,1). clause(4,g,1).  clause(5,d,-1). clause(6,e,1).
clause(7,i2,1). clause(7,i1,-1).

scenario(1). scenario(2).

constrained(1,i1,1).

goal(1,o1,-1). goal(1,o2,1). goal(2,a,1).
```

3.2 Logic program

Next we describe our encoding for solving the minimal intervention set problem as described in Section 2. Our ASP encoding is shown in Listing 2.

Listing 2. Logic program for solving the minimal intervention set problem

```
goal(T,S)        :- goal(_,T,S).
goal(T)          :- goal(T,_).
constrained(Z,E) :- constrained(Z,E,_).
constrained(E)   :- constrained(_,E).

satisfy(V,W,S) :- formula(W,D), dnf(D,C), clause(C,V,S).
closure(V,T)   :- goal(V,T).
closure(V,S*T) :- closure(W,T), satisfy(V,W,S), not goal(V,-S*T).

{ intervention(V,S) : closure(V,S) : candidate(V) }.
:- intervention(V,1), intervention(V,-1).
intervention(V) :- intervention(V,S).

eval(Z,V,S) :- scenario(Z), intervention(V,S).
eval(Z,E,S) :- constrained(Z,E,S), not intervention(E).
free(Z,V,D) :- formula(V,D), scenario(Z),
```

```
17                not constrained(Z,V), not intervention(V).
18
19  eval_clause(Z,C,-1) :- clause(C,V,S), eval(Z,V,-S).
20
21  eval(Z,V, 1) :- free(Z,V,D), eval(Z,W,T) : clause(C,W,T), dnf(D,C).
22  eval(Z,V,-1) :- free(Z,V,D), eval_clause(Z,C,-1) :        dnf(D,C).
23
24  :- goal(Z,T,S), not eval(Z,T,S).
25
26  #const max=0.
27  :- max>0, max + 1 #count{ intervention(_) }.
28
29  #minimize { intervention(_) }.
```

Note that in the following we use 1 and -1 for truth assignments to $\boldsymbol{t}$ and $\boldsymbol{f}$, respectively. Furthermore, undefined variables are represented by the absence of assignments to both $\boldsymbol{t}$ and $\boldsymbol{f}$. In Lines 1-4 we define auxiliary domain predicates used in the remainder of the encoding.

Lines 6-8 deserve closer attention since they allow us to reduce significantly the search space of candidate solutions. We incorporate a preprocessing step introduced in Samaga *et al.* (2010) that prunes variable assignments that can never be part of a minimal intervention set. The idea is to inductively collect all assignments that could be used to support a goal. First we gather all assignments that make a literal in a clause true and associate it with variable of the associated DNF (Line 6). Starting from the assignments that can satisfy a goal literal directly (Line 7), we inductively consider variable assignments (Line 8) that can support the assignments collected so far.

Let us illustrate this on our toy example. In order to satisfy `goal(1,o2,1)`, one would never consider to intervene variables f or g negatively. Since both reach $o2$ positively, only positive interventions on them could help. The same happens for variable e. However, since e also reaches $o1$ positively and we have `goal(1,o1,-1)`, a negative intervention of e could help for this goal.

Next, we use a choice rule in Line 10 to generate candidate solutions. We only choose interventions collected in the preprocessing step above. The integrity constraint in Line 11 eliminates contradictory interventions. Whereas Line 12 projects the intervention set to the intervened variables regardless of their signature. For example, one could generate the intervention set consisting of `intervention(e,1)` and `intervention(c,-1)`.

Next, we describe in lines 14-15 which variables are clamped in the network according to the side constraints C in each scenario and the interventions I, i.e., $(V, \phi|_{C \circ I})$. Following the previous example, this will generate predicates `eval(1,i1,1)`, `eval(1,e,1)`, `eval(2,e,1)`, `eval(1,c,-1)` and `eval(2,c,-1)`. The rule in Line 16 captures the remaining variables that have not been clamped i.e., predicates `free/3`. Then in Line 19 we declare the fact that a conjunction evaluates to $\boldsymbol{f}$ if at least one of the literals occurring in it evaluates to $\boldsymbol{f}$. Afterwards, the rules in Lines 21 and 22 inductively propagate the truth values $\boldsymbol{t}$ and $\boldsymbol{f}$ in the standard way starting from the clamped truth values. Exploiting inductive definitions is ASP, this allows us to compute for each scenario the fixpoint of $\Omega_{(V,\phi|_{C \circ I})}$. For our example, we can

see how the positive intervention over e is propagated as follows. Since variable f is not intervened, its formula ($f \mapsto e \vee g$) described by predicates `formula(f,6)`, `dnf(6,4)`, `dnf(6,6)`, `clause(4,g,1)` and `clause(6,e,1)`, is "free" in all scenarios, i.e., `free(1,f,6)` and `free(2,f,6)`. Further, given that we have `dnf(6,6)` related only to `clause(6,e,1)` and e was intervened positively, the truth value for f is propagated in all scenarios regardless of g, i.e., `eval(1,f,1)` and `eval(2,f,1)`. Analogously, truth values are propagated through all the network until predicates `eval/3` describe the fixpoint in every scenario.

Line 24 declares an integrity constraint to eliminate solutions that do not satisfy all goals in each scenario. The statements in Line 26 and 27 allows us to optionally bound the problem by considering only intervention sets up to a given size. And finally, the minimize constraint in Line 29 denotes that we are interested in solutions involving a minimal number of interventions.

3.3 Solving

Using a standard ASP solver such as *clasp* (Gebser *et al.* 2007), our encoding will actually generate cardinality minimal models whereas we are interested in inclusion minimal models. Towards this end, we evaluate two alternative solvers derived from *clasp*, namely *claspD* (Gebser *et al.* 2013) and *hclasp* (Gebser *et al.* 2013). The disjunctive solver *claspD* together with the encodings of the *metasp* framework (Gebser *et al.* 2011) can easily be used to compute inclusion minimal models.[5] On the other hand, *hclasp* is a recently developed solver incorporating domain-specific heuristics into the input language allowing us to compute inclusion minimal models too. Thus, in order to use *hclasp*, the minimize constraint in the last line of the encoding in Listing 2 must be replaced by the rule in Listing 3.

Listing 3. Changes to Listing 2 in order to use *hclasp*

```
29 _heuristic(intervention(V),false,1) :- closure(V,S), candidate(V).
```

Together with *hclasp*'s option `--heuristic=domain`, the effect of the rule in Listing 3 is to tell the solver that atoms of the form `intervention(V)` must be chosen first and assigned truth value ***f***. This guarantees that the first answer set found is inclusion minimal regarding the set of atoms in focus (Castell *et al.* 1996). To further ensure inclusion minimality for all subsequent answer sets, a constraint must be added excluding the previous answer set (cf. (Di Rosa *et al.* 2010)); this is invoked by *hclasp*'s option `--enum-mode record`.

In Listing 4 we show the models found for the toy instance described in Listing 1 using *claspD*. Of course, the same models are found with *hclasp* using the following command line: `gringo enc-hclasp.lp toy.lp | hclasp --heuristic=domain --enum-mode record 0`

[5] The encodings are available at `http://www.cs.uni-potsdam.de/wv/metasp/`

Listing 4. Computing all MISs for the toy instance

```
$ gringo --reify encoding.lp toy.lp |\
  gringo - meta.lp metaD.lp meta0.lp <(echo "optimize(1,1,incl).") |\
  claspD 0
claspD version 2
Reading from stdin
Solving...
Answer: 1
intervention(i2,-1) intervention(b,-1) intervention(f,1)
Answer: 2
intervention(e,-1) intervention(b,-1) intervention(f,1)
Answer: 3
intervention(e,-1) intervention(b,-1) intervention(g,1)
Answer: 4
intervention(i2,-1) intervention(b,-1) intervention(g,1)
Answer: 5
intervention(b,-1) intervention(f,1) intervention(d,-1)
Answer: 6
intervention(b,-1) intervention(d,-1) intervention(g,1)
Answer: 7
intervention(g,1) intervention(c,-1)
Answer: 8
intervention(e,1) intervention(c,-1)
Answer: 9
intervention(f,1) intervention(c,-1)
SATISFIABLE

Models       : 9
Time         : 0.034s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.010s
```

4 Benchmarks

We evaluate the performance and scalability of our ASP solution over four real-world and biologically relevant benchmarks (Table 1). Three of these benchmarks (*EGFR*, *EGFR multiple* and *TCR*) were used in Samaga *et al.* (2010) and their corresponding logical networks were recently published (Saez-Rodriguez *et al.* 2007; Samaga *et al.* 2009). Further, we also use a larger unpublished logical network (*TBH6b*) provided by Axel von Kamp and Steffen Klamt. While the authors in Samaga *et al.* (2010) restricted their study to a maximum cardinality of 3, herein we extend this limitation to a maximum cardinality of 10 or no limit at all. For each case we report the number of $\subseteq$-minimal intervention sets (MISs) and CPU time in seconds for each solver. Computations were run on a MacBook Pro, Intel Core i7, 2.7 GHz and 4 GB of RAM using *gringo*-3.0.3 together with *claspD*-2 and *hclasp*, respectively. In Table 1 we describe our benchmark problem instances. For each case we report the number of variables (i.e. species) in the logical network, the number of intervention scenarios, the number of intervention constraints, the number of intervention goals, and the number of candidate intervention sets. The number of candidate intervention sets is computed as follows. Let n be the number of variables that could be intervened positively and negatively. Let m be the number of variables that could be intervened either positively or negatively, but not both. Then, the number of intervention sets is given by $3^n \times 2^m$. We note that for *EGFR*, *TCR* and *TBH6b* interventions are forbidden over constrained and goal variables, whereas for *EGFR multiple* this is not

Table 1. *Benchmark instances of the multi-scenario intervention set problem*

Instance $(G_j, C_j)_{j\in J}$ wrt (V, ϕ)	Variables $\lvert V\rvert$	Scenarios $\lvert J\rvert$	Side constraints $\sum_{j\in J}\lvert C_j\rvert$	Goals $\sum_{j\in J}\lvert G_j\rvert$	Candidate MISs
EGFR	103	1	28	2	$3^{13}\times 2^{19}$
EGFR multiple	103	34	974	408	$3^{52}\times 2^{15}$
TCR	94	1	17	12	$3^{35}\times 2^{28}$
TBH6b	203	1	17	3	$3^{85}\times 2^{24}$

Table 2. *MISs calculation up to a given size (k) or unbounded (∞) with a timeout set to 1000 sec. For each instance we report the number of intervention sets (I) and CPU time (sec) of both, claspD and hclasp*

	EGFR			EGFR multiple			TCR			TBH6b		
k	*I*	*claspD*	*hclasp*	*I*	*claspD*	*hclasp*	*I*	*claspD*	*hclasp*	*I*	*claspD*	*hclasp*
1	15	0.02	0.00	0	0.86	0.07	0	0.04	0.00	0	0.75	0.00
2	21	0.11	0.00	0	0.93	0.08	0	0.05	0.00	0	0.81	0.00
3	21	0.13	0.00	8	1.15	0.10	7	0.05	0.00	6	0.92	0.00
4	21	0.14	0.00	38	11.16	0.15	26	0.36	0.03	6	18.00	0.00
5	21	0.17	0.00	74	41.57	0.21	1196	2.04	0.08	6	17.40	0.00
6	21	0.16	0.00	83	73.14	0.19	4290	60.38	0.37	15	20.28	0.00
7	21	0.16	0.00	83	90.66	0.20	7258	198.19	0.35	15	47.19	0.00
8	21	0.16	0.00	83	90.56	0.24	8776	465.67	0.75	24	52.11	0.01
9	21	0.18	0.00	83	88.95	0.25	9316	655.22	1.09	207	90.67	0.04
10	21	0.19	0.00	83	84.60	0.23	9704	669.64	0.94	1248	667.85	0.23
∞	21	0.13	0.00	83	91.83	0.19	13016	512.52	1.19	-[a]	-	-

[a] Both solvers have reached a timeout for this case, however their behaviors are very different. After 1000 seconds, *claspD* found only 648 models whereas *hclasp* found 894483 models

the case (there are no forbidden variables). This relaxation is necessary in order to reproduce the results reported in Samaga *et al.* (2010) and compare both methods. Nonetheless, it worth noting that there are no solutions satisfying the 34 scenarios. In fact, depending on the application at hand, finding no solution could be more interesting than relaxing the problem. In Table 2 we show the performance over the four problem instances. First, we were able to compute all MISs of up to size 10 for all instances regardless of the solver used. This represents a significant improvement compared to previous approaches limited in practice to compute intervention sets having maximum cardinality of 3 or 4. Moreover, for *EGFR*, *EGFR multiple* and *TCR* we were able to solve the unbounded problem, i.e. $k = \infty$. That is, for the mentioned instances, we are able to completely characterize all feasible inclusion minimal intervention sets. Further, while for $k \leqslant 4$ the difference between *claspD* and *hclasp* may not be very evident, for $5 \leqslant k \leqslant 10$ computation times for *claspD* tend to grow significantly whereas computation times for *hclasp* remain relatively constant. Finally, for the unbounded case of *TBH6b* where we found a timeout after 1000 seconds, it is interesting to note the different behavior of each solver (see footnote in Table 2). One reason for the worse performance of *claspD* is related to

the way *metasp* implements subset minimization. Currently, for non-tight programs, the encoding of inclusion minimality in *metasp* leads to a quadratic blowup in the propositional program passed to *claspD*. Compare with the *EGFR* instance where the underlying network is acyclic and hence the resulting propositional program is tight. The performance of *claspD* is better on this instance. Overall, *hclasp*'s solution recording based algorithm appears to be the better choice for enumerating subset minimal answers.

5 Discussion and conclusion

Logic modeling is an emerging qualitative approach to capture mechanistic behavior in large-scale biological systems. Despite of its relative simplicity, it allows for addressing relevant problems related to drug target identification, experimental design, and diagnosis. In this context, finding minimal intervention strategies in a logic signaling network with desired outcomes leads to challenging combinatorial problems that require advanced solving technologies. Previous work on this subject consists of dedicated algorithms and special purpose search space reduction techniques for coping with combinatorial explosions. In fact, in practice, such algorithms are limited to searching intervention sets having only a small number of interventions (eg. $\leqslant 3$).

In this work, we have provided a precise characterization of the *minimal intervention set* problem relying on Kleene's three-valued logic and fixpoint semantics (close to traditional logic programming concepts). In this context, our fixpoint characterization allows us to capture the steady states of a logical network following from a set of clamped values. We have proposed an ASP encoding for this problem and we have evaluated its performance using real-world biological benchmarks. Negation by default and the recursive definition of reachability make of ASP a very suitable framework for this problem. For addressing our problem's complexity, we have used and compared the ASP solvers *claspD* and *hclasp*.

Our ASP encoding incorporates a search space reduction based on the interaction graph underlying a logical network (Samaga *et al.* 2010). This can be exploited during the grounding phase, significantly reducing the number of candidate solutions. In fact, we considered also other special purpose techniques from the aforecited work but did not observe any improvements in performance on the available instances. Experiments have shown that our approach outperforms the previous dedicated algorithms in up to four orders of magnitude (for small number of interventions ($\leqslant 3$) still feasible for the algorithm). This was not very surprising since such algorithms are based on a standard breadth-first search using additional techniques for search space reduction. More importantly, we are able to search for significantly larger intervention sets or even solve the unbounded problem (ie. no limit in the number of interventions). While considering a small number of interventions the number of solutions (i.e. intervention sets) is in the order of tens, with a larger number of interventions we have found thousands of feasible solutions. Furthermore, being able to solve the unbounded problem allows us to completely characterize the set of solutions.

Nowadays, in signaling networks, large-scale (> 10) interventions combining different inhibitors can be considered as a long-term technological perspective. Nevertheless, in 2 out of 4 benchmarks (EGFR and EGFR mutiple), we did not find intervention sets of size bigger than 10. This suggests that extending the set of interventions may not be interesting for such models. Similarly, for the remaining benchmark where we have solved the unbounded problem (TCR), $\sim 70\%$ of the minimal intervention sets have size smaller than 10. This provides also a useful information on the flexibility of the system. On the contrary, knowing that a large number of interventions are required to reach certain state could help to understand (at least theoretically) the systems' robustness. Altogether, being able to compute both small and large admissible intervention sets, appears as an interesting and relevant feature of our approach.

Both computational and biological perspective tracks are open. On the computational side, a precise estimation of the empirical complexity appears to be non trivial and very specific to each instance. Given a problem instance, the number of candidate solutions can be computed analytically. However, experiments have shown that this is not the only parameter determining the empirical complexity. The number of goal variables and their location in the logical network, the number of intervention scenarios, and topological properties of the network may have an impact on the computational efforts required in practice. More generally, it will be interesting to investigate in how far the employed concepts from logic programming furnish an adequate and general tool for addressing problems in (Boolean) networks, as also indicated by the work of Inoue and Sakama in Inoue (2011; Inoue and Sakama (2012). On the biological side, in the light of such a large number of solutions, the way to select among them arises. In general, when the inherent noise is taken into account, several logical networks can describe a biological system equally (or similarly) well (Saez-Rodriguez *et al.* 2009). Thus, one could extend the intervention set problem to a family of plausible logical networks. This way, we would reduce the number of solutions by selecting the more robust of them. That is, intervention sets satisfying each scenario in all networks.

Acknowledgements

We would like to thank Steffen Klamt, Axel von Kamp, and Regina Samaga for their help in providing the benchmarks, reading the manuscript, and many valuable discussions. The work of SV was supported by the project ANR-10-BLANC-0218. The work of RK and TS was partly funded by DFG grant SCHA 550/9-1.

References

Abdi, A., Tahoori, M. B. and Emamian, E. S. 2008. Fault diagnosis engineering of digital circuits can identify vulnerable molecules in complex cellular pathways. *Science Signaling 1,* 42, ra10.

Acuña, V. V., Milreu, P. V. P., Cottret, L. L., Marchetti-Spaccamela, A. A., Stougie, L. L. and Sagot, M.-F. M. 2012. Algorithms and complexity of enumerating minimal precursor sets in genome-wide metabolic networks. *Bioinformatics 28,* 19 (September), 2474–2483.

Baral, C. 2003. *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press, Cambridge, UK.

Baral, C., Chancellor, K., Tran, N., Tran, N., Joy, A. and Berens, M. 2004. A knowledge based approach for representing and reasoning about signaling networks. In *Proceedings of the Twelfth International Conference on Intelligent Systems for Molecular Biology/Third European Conference on Computational Biology (ISMB'04/ECCB'04)*, 15–22.

Batt, G., de Jong, H., Page, M. and Geiselmann, J. 2008. Symbolic reachability analysis of genetic regulatory networks using discrete abstractions. *Automatica 44,* 4 (March), 982–989.

Bouaynaya, N. N., Shterenberg, R. and Schonfeld, D. 2011. Methods for optimal intervention in gene regulatory networks. *Signal Processing Magazine, IEEE 29,* 1 (December), 158–163.

Calzone, L. L., Tournier, L. L., Fourquet, S. S., Thieffry, D. D., Zhivotovsky, B. B., Barillot, E. E. and Zinovyev, A. A. 2010. Mathematical modelling of cell-fate decision in response to death receptor engagement. *PLoS Computational Biology 6,* 3 (February), e1000702–e1000702.

Castell, T., Cayrol, C., Cayrol, M. and Le Berre, D. 1996. Using the Davis and Putnam procedure for an efficient computation of preferred models. In *Proceedings of the Twelfth European Conference on Artificial Intelligence (ECAI'96)*, W. Wahlster, Ed. John Wiley & sons, 350–354.

Di Rosa, E., Giunchiglia, E. and Maratea, M. 2010. Solving satisfiability problems with preferences. *Constraints 15,* 4, 485–515.

Erdem, E. and Türe, F. 2008. Efficient haplotype inference with answer set programming. In *Proceedings of the Twenty-third National Conference on Artificial Intelligence (AAAI'08)*, D. Fox and C. Gomes, Eds. AAAI Press, 436–441.

Faryabi, B., Vahedi, G., Chamberland, J.-F., Datta, A. and Dougherty, E. R. 2008. Optimal constrained stationary intervention in gene regulatory networks. *EURASIP Journal of Bioinformatics and Systems Biology 2008.*

Fitting, M. 1985. A Kripke-Kleene semantics for logic programs. *Journal of Logic Programming 2,* 4, 295–312.

Gebser, M., Guziolowski, C., Ivanchev, M., Schaub, T., Siegel, A., Thiele, S. and Veber, P. 2010. Repair and prediction (under inconsistency) in large biological networks with answer set programming. In *Proceedings of the Twelfth International Conference on Principles of Knowledge Representation and Reasoning (KR'10)*, F. Lin and U. Sattler, Eds. AAAI Press, 497–507.

Gebser, M., Kaminski, R., Kaufmann, B., Ostrowski, M., Schaub, T. and Thiele, S. A user's guide to `gringo`, `clasp`, `clingo`, and `iclingo`.

Gebser, M., Kaminski, R., Kaufmann, B. and Schaub, T. 2012. *Answer Set Solving in Practice*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool Publishers.

Gebser, M., Kaminski, R. and Schaub, T. 2011. Complex optimization in answer set programming. *Theory and Practice of Logic Programming 11,* 4-5, 821–839.

Gebser, M., Kaufmann, B., Neumann, A. and Schaub, T. 2007. clasp: A conflict-driven answer set solver. In *Proceedings of the Ninth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR'07)*, C. Baral, G. Brewka, and J. Schlipf, Eds. Lecture Notes in Artificial Intelligence, vol. 4483, Springer-Verlag, 260–265.

Gebser, M., Kaufmann, B., Otero, R., Romero, J., Schaub, T. and Wanko, P. 2013. Domain-specific heuristics in answer set programming. In *Proceedings of the Twenty-Seventh National Conference on Artificial Intelligence (AAAI'13)*, M. desJardins and M. Littman, Eds. AAAI Press. To appear.

Gebser, M., Kaufmann, B. and Schaub, T. 2013. Advanced conflict-driven disjunctive answer set solving. In *Proceedings of the Twenty-third International Joint Conference on Artificial Intelligence (IJCAI'13)*, F. Rossi, Ed. IJCAI/AAAI. To appear.

Gebser, M., Schaub, T., Thiele, S. and Veber, P. 2011. Detecting inconsistencies in large biological networks with answer set programming. *Theory and Practice of Logic Programming 11,* 2-3, 323–360.

Inoue, K. 2011. Logic programming for boolean networks. In *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence (IJCAI'11)*, T. Walsh, Ed. IJCAI/AAAI, 924–930.

Inoue, K. and Sakama, C. 2012. Oscillating behavior of logic programs. In *Correct Reasoning*, E. Erdem, J. Lee, Y. Lierler and D. Pearce, Eds. Lecture Notes in Computer Science, vol. 7265. Springer, 345–362.

Karlebach, G. G. and Shamir, R. R. 2009. Minimally perturbing a gene regulatory network to avoid a disease phenotype: the glioma network as a test case. *BMC Systems Biology 4*, 15–15.

Kauffman, S. 1969. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology 22,* 3 (February), 437–467.

Kauffman, K. J., Prakash, P. and Edwards, J. S. 2003. Advances in flux balance analysis. *Current opinion in biotechnology 14,* 5 (October), 491–496.

Kitano, H. 2002. Systems biology: a brief overview. *Science 295,* 5560, 1662–1664.

Klamt, S. S. 2006. Generalized concept of minimal cut sets in biochemical networks. *Biosystems 83,* 2-3 (January), 233–247.

Klamt, S., Haus, U.-U. and Theis, F. J. 2009. Hypergraphs and cellular networks. *PLoS Computational Biology 5,* 5 (May), e1000385.

Kleene, S. 1950. *Introduction to Metamathematics*. Princeton, NJ, 1950.

Kreutz, C. and Timmer, J. 2009. Systems biology: experimental design. *FEBS Journal 276,* 4 (January), 923–942.

Mitsos, A., Melas, I., Siminelakis, P., Chairakaki, A., Saez-Rodriguez, J. and Alexopoulos, L. G. 2009. Identifying Drug effects via pathway alterations using an integer linear programming optimization formulation on phosphoproteomic data. *PLoS Computational Biology 5,* 12, e1000591.

Morris, M., Saez-Rodriguez, J., Sorger, P. and Lauffenburger, D. A. 2010. Logic-based models for the analysis of cell signaling networks. *Biochemistry 49,* 15, 3216–3224.

Naldi, A., Carneiro, J., Chaouiya, C. and Thieffry, D. 2009. Diversity and plasticity of th cell types predicted from regulatory network modelling. *PLoS Computational Biology 6,* 9 (December), e1000912.

Ray, O., Whelan, K. and King, R. 2010. Logic-based steady-state analysis and revision of metabolic networks with inhibition. In *Proceedings of the 2010 International Conference on Complex, Intelligent and Software Intensive Systems (CISIS '10) 0*, 661–666.

Saez-Rodriguez, J., Alexopoulos, L. G., Epperlein, J., Samaga, R., Lauffenburger, D. A., Klamt, S. and Sorger, P. 2009. Discrete logic modelling as a means to link protein signalling networks with functional analysis of mammalian signal transduction. *Molecular Systems Biology 5,* 331.

Saez-Rodriguez, J., Alexopoulos, L. G., Zhang, M., Morris, M., Lauffenburger, D. A. and Sorger, P. 2011. Comparing signaling networks between normal and transformed hepatocytes using discrete logical models. *Cancer Research 71,* 16, 5400.

Saez-Rodriguez, J., Simeoni, L., Lindquist, J., Hemenway, R., Bommhardt, U., Arndt, B., Haus, U.-U., Weismantel, R., Gilles, E., Klamt, S. and Schraven, B. 2007. A logical model provides insights into T cell receptor signaling. *PLOS Computational Biology 3,* 8 (August), e163.

Samaga, R., Saez-Rodriguez, J., Alexopoulos, L. G., Sorger, P. and Klamt, S. 2009. The logic of EGFR/ErbB signaling: theoretical properties and analysis of high-throughput data. *PLoS Computational Biology 5,* 8 (August), e1000438.

Samaga, R., Von Kamp, A. and Klamt, S. 2010. Computing combinatorial intervention strategies and failure modes in signaling networks. *Journal of Computational Biology 17,* 1 (Jan.), 39–53.

Sharan, R. and Karp, R. M. 2012. Reconstructing boolean models of signaling. In *Research in Computational Molecular Biology*. Springer Berlin Heidelberg, Berlin, Heidelberg, 261–271.

Sparkes, A., Aubrey, W., Byrne, E., Clare, A., Khan, M. N., Liakata, M., Markham, M., Rowland, J., Soldatova, L. N., Whelan, K. E., Young, M. and King, R. D. 2010. Towards robot scientists for autonomous scientific discovery. *Automated Experimentation 2*, 1–1.

Stelling, J. J., Klamt, S. S., Bettenbrock, K. K., Schuster, S. S. and Gilles, E. D. E. 2002. Metabolic network structure determines key aspects of functionality and regulation. *Nature 420,* 6912 (November), 190–193.

Stelling, J., Sauer, U., Szallasi, Z., Doyle, F. and Doyle, J. 2004. Robustness of Cellular Functions. *Cell 118,* 6, 675–685.

Thomas, R. R. 1973. Boolean formalization of genetic control circuits. *Journal of Theoretical Biology 42,* 3 (November), 563–585.

Videla, S., Guziolowski, C., Eduati, F., Thiele, S., Grabe, N., Saez-Rodriguez, J. and Siegel, A. 2012. Revisiting the training of logic models of protein signaling networks with ASP. In *Computational Methods in Systems Biology 2012*, D. Gilbert and M. Heiner, Eds. Springer Berlin / Heidelberg, 342–361.

Wang, R.-S. and Albert, R. 2011. Elementary signaling modes predict the essentiality of signal transduction network components. *BMC Systems Biology 5*, 44.

Wang, B. and Buck, M. 2012. Customizing cell signaling using engineered genetic logic circuits. *Trends in Microbiology 20,* 8 (August), 376–384.

Wang, R.-S. R., Saadatpour, A. A. and Albert, R. R. 2012. Boolean modeling in systems biology: an overview of methodology and applications. *Physical biology 9,* 5 (September), 055001–055001.