Type Inference in Systems Biology

François Fages, Sylvain Soliman

Firstname.Lastname@inria.fr
Projet Contraintes, INRIA Rocquencourt,
BP105, 78153 Le Chesnay Cedex, France.
http://contraintes.inria.fr

Abstract. Type checking and type inference are important concepts and methods of programming languages and software engineering. Type checking is a way to ensure some level of consistency, depending on the type system, in large programs and in complex assemblies of software components. Type inference provides powerful static analyses of pre-existing programs without types, and facilitates the use of type systems by freeing the user from entering type information. In this paper, we investigate the application of these concepts to systems biology. More specifically, we consider the Systems Biology Markup Language SBML and the Biochemical Abstract Machine BIOCHAM with their repositories of models of biochemical systems. We study three type systems: one for checking or inferring the functions of proteins in a reaction model, one for checking or inferring the activation and inhibition effects of proteins in a reaction model, and another one for checking or inferring the topology of compartments or locations. We show that the framework of abstract interpretation elegantly applies to the formalization of these abstractions and to the implementation of linear time type checking as well as type inference algorithms. Through some examples, we show that the analysis of biochemical models by type inference provides accurate and useful information. Interestingly, such a mathematical formalization of the abstractions used in systems biology already provides some guidelines for the extensions of biochemical reaction rule languages.

1 Introduction

Type checking and type inference are important concepts and methods of programming languages and software engineering [1]. Type checking is a way to ensure some level of consistency, depending on the type system, in large programs and in complex assemblies of software components. Type inference provides powerful static analyzes of pre-existing programs without types, and facilitates the use of type systems by freeing the user from entering type information.

In this paper, we investigate the application of these concepts to systems biology. More specifically, we consider the Systems Biology Markup Language SBML [2] and the Biochemical Abstract Machine BIOCHAM [3]. In both of these languages, the biochemical models are described through a set of reaction rules. We study three type systems:

1. one for checking or inferring the protein functions in a reaction model,
2. one for checking or inferring the activation and inhibition effects in a reaction model,
3. and another one for checking or inferring the topology of compartments or locations in reaction models with space considerations.

To this end, the formal framework of abstract interpretation will be used to provide type systems with a precise mathematical definition. Abstract interpretation is a theory of abstraction introduced by Cousot and Cousot in [4] as a framework for reasoning about programs, their semantics, and for designing static analysers, among which type inference systems [5]. Although not strictly necessary to the presentation of the type inference methods considered in this paper, we believe that that formal framework is very relevant to systems biology, as a formalism for providing a mathematical sense to modeling issues concerning multiple abstraction levels and their formal relationship.

We show that the framework of abstract interpretation elegantly applies to the formalization of the three abstractions considered in this paper and to the implementation of linear time type checking as well as type inference algorithms. Through examples of biochemical systems coming from the `biomodels.net` and BIOCHAM repositories of models, we show that the static analysis of reaction models by type inference provides both accurate and useful information. Interestingly, we show that these considerations also provide some guidelines concerning the extensions of biochemical reaction rule-based languages.

2 Preliminaries on Abstract Interpretation, Type Checking and Type Inference

2.1 Concrete Domain of Reaction Models

Following SBML and BIOCHAM conventions, a model of a biochemical system is a set of reaction rules of the form e `for` S `=>` S' where S is a set of molecules given with their stoichiometric coefficient, called a *solution*, S' is the transformed solution, and e is a kinetic expression involving the concentrations of molecules (which are not strictly required to appear in S). The set of molecules is noted $\mathcal{M}$. We will use the BIOCHAM operators `+` and `*` to denote solutions as `2*A + B`, as well as the syntax of catalyzed reactions `e for S =[C]=> S'` as an abbreviation for `e for S+C => S'+C`.

A set of reaction rules like $\{e_i$ `for` S_i `=>` $S'_i\}_{i=1,...,n}$ over molecular concentration variables $\{x_1, ..., x_m\}$, can be interpreted under different semantics. The traditional *differential semantics* interpret the rules by the following system of Ordinary Differential Equations (ODE):

$$dx_k/dt = \sum_{i=1}^{n} r_i(x_k) * e_i - \sum_{j=1}^{n} l_j(x_k) * e_j$$

where $r_i(x_k)$ (resp. l_i) is the stoichiometric coefficient of x_k in the right (resp. left) member of rule i. Thanks to its wide range of mathematical tools, this semantics is the most commonly used, when the data is available and the system of a reasonable size. The *stochastic semantics* interpret the kinetic expressions as transition probabilities (see for instance [6]), while the *boolean semantics* forget the kinetic expressions and interpret the rules as a non-deterministic (asynchronous) transition system over boolean states representing the absence or presence of molecules. In BIOCHAM these three semantics are implemented [7], while in the SBML exchange format, no particular semantics are defined.

For the simple analyzes considered in this paper, the concrete domain of reaction models will be the syntactic domain of formal reaction rules, with no other semantics than a data structure. A reaction model is thus a set of reaction rules, and the domain of reaction models is ordered by set inclusion, i.e. by the information ordering.

Definition 1. *The universe of reactions is the set of possible rules*
$\mathcal{R} = \{e \texttt{ for } S \texttt{ => } S' \mid e$ *is a kinetic expression,*
and S and S' are solutions $\}$.
The concrete domain $\mathcal{D}_{\mathcal{R}}$ of reaction models is the power-set of reaction rules ordered by inclusion $\mathcal{D}_{\mathcal{R}} = (\mathcal{P}(\mathcal{R}), \subseteq)$.

2.2 Abstract Domains, Abstractions and Galois Connections

In the general setting of abstract interpretation, an abstract domain is a lattice $L(\sqsubseteq, \bot, \top, \sqcup, \sqcap)$ defined by the set L and the partial order $\sqsubseteq$, and where $\bot$, $\top$, $\sqcup$, $\sqcap$ denote the least element, the greatest element, the least upper bound and the greatest lower bound respectively.

As often the case in program analysis, the concrete domain and the abstract domains considered for analyzing biochemical models, are power-sets, that is set lattices $\mathcal{P}(\mathcal{S})(\subseteq, \emptyset, \mathcal{S}, \cup, \cap)$ ordered by inclusion, with the empty set as $\bot$ element, and the base set $\mathcal{S}$ (such as the universe of reaction rules here) as $\top$ element. An abstraction is formalized by a Galois connection as follows [4]:

Definition 2. *A Galois connection $C \to_\alpha A$ between two lattices C and A is defined by abstraction and concretization functions $\alpha : C \to A$ and $\gamma : A \to C$ that satisfy $\forall c \in C, \forall y \in A : x \sqsubseteq_C \gamma(y) \Leftrightarrow \alpha(x) \sqsubseteq_A y$.*

For any Galois connection, we have the following properties:

1. $\gamma \circ \alpha$ is extensive (i.e. $x \sqsubseteq_C \gamma \circ \alpha(x)$) and represents the information lost by the abstractions;
2. α preserves $\sqcup$, and γ preserves $\sqcap$;
3. α is one-to-one *iff* γ is onto *iff* $\gamma \circ \alpha$ is the identity.

If $\gamma \circ \alpha$ is the identity, the abstraction α loses no information, and C and A are isomorphic from the information standpoint (although γ may not be one-to-one).

We will consider three abstract domains: one for protein functions, where molecules are abstracted into categories such as kinases and phosphatases, one for the influence graph, where the biochemical reaction rules are abstracted in activation and inhibition binary relations between molecules, and one for location topologies, where the reaction (and transport) rules are abstracted retaining only the neighborhood information between locations.

2.3 Type Checking and Type Inference by Abstract Interpretation

In this setting, a type system A for a concrete domain C is simply a Galois connection $C \rightarrow_\alpha A$. The *type inference* problem is, given a concrete element $x \in C$ (e.g. a reaction model) to compute $\alpha(x)$ (e.g. the protein functions that can be inferred from the reactions). The *type checking* problem is, given a concrete element $x \in C$ and a typing $y \in A$ (e.g. a set of protein functions), to determine whether $x \sqsubseteq_C \gamma(y)$ (i.e. whether the reactions provide less and compatible information on the protein functions) which is equivalent to $\alpha(x) \sqsubseteq_A y$ (i.e. whether the typing contains the inferred types).

The simple type systems considered in this paper will be implemented with type checking and type inference algorithms that basically browse the reactions, and check or collect the type information for each rule independently and in linear time.

3 A Type System for Protein Functions

To investigate the use of type inference in the domain of protein functions we first restrict ourselves to two simple functions: kinase and phosphatase. These correspond to the action of adding (resp. removing) a phosphate group to (resp. from) a compound.

For the sake of simplicity, we do not consider other categories such as protease (in degradation rules), acetylase and deacetylase (in modification rules), etc. This choice is in accordance with the BIOCHAM syntax which allows to mark the modified sites of a protein with the operator `~`, as in `P~{p,q}` without distinguishing however between a phosphorylation and an acetylation for instance. We thus consider BIOCHAM models containing compounds with different levels of phosphorylation or acetylation, without distinguishing the different forms of modification, and call them phosphorylation, by abuse of terminology.

The analysis of protein functions in a reaction model is interesting for several reasons. First, the kind of information (kinase activity) collected on proteins can be checked using online databases like GO, the Gene Ontology [8]. Second, in the context of the machine learning techniques implemented in BIOCHAM for completing or revising a model w.r.t. a temporal logic specification [7], the information that an enzyme acts as a kinase or as a phosphatase drastically reduce the search space for reaction additions, and help find more biologically plausible model revisions.

3.1 Abstract Domain of Protein Functions

Definition 3. *The abstract domain of protein functions $\mathcal{D}_\mathcal{F}$ is the domain of functions from molecules $\mathcal{M}$ to pairs of booleans, representing "has kinase function" (true/false) and "has phosphatase function" (true/false).*

Definition 4. $\alpha : \mathcal{D}_\mathcal{R} \rightarrow \mathcal{D}_\mathcal{F}$ *is defined for each molecule as the disjunction of α on each single rule and each pair of rules:*

$\alpha($`A =[B]=> C`$) =$ *where* `C` *is more phosphorylated than* `A` *(i.e. its set of active phosphorylation sites strictly includes that of* `A`*) is abstracted as* B has kinase function.

$\alpha($`A =[B]=> C`$) =$ *where, on the contrary,* `A` *is more phosphorylated than* `C`*, we abstract that* B has phosphatase function.

$\alpha($`A + B => A-B, A-B => C + B`$) =$ *where* `C` *is more phosphorylated than* `A` *is abstracted as* B has kinase function.

$\alpha($`A + B => A-B, A-B => C + B`$) =$ *where, on the contrary,* `A` *is more phosphorylated than* `C`*, we abstract that* B has phosphatase function.

3.2 Evaluation Results

MAPK model. On a simple example of the MAPK cascade extracted from the SBML repository and originally based on [9], the type inference algorithm determines that `RAFK`, `RAF~{p1}` and `MEK~{p1,p2}` have a kinase function; `RAFPH`, `MEKPH` and `MAPKPH` have a phosphatase function; and the other compounds have no function inferred.

If we wanted to type-check such a model, we would correctly check all phosphatases but would miss an example of the kinase function of `MAPK~{p1,p2}`, since its action is not visible in the above model.

Kohn's Map. Kohn's map of the mammalian cell cycle control [10] has been transcribed in BIOCHAM to serve as a large benchmarking example of 500 species and 800 rules [11]. To check if this abstraction scales up we tried it on this model, and indeed obtain the answer in less than one second CPU time (on a PC 1,7GHz). Here is an excerpt of the output of the type inference:

```
cdk7-cycH is a kinase
Wee1 is a kinase
Myt1 is a kinase
cdc25C~{p1} is a phosphatase
cdc25C~{p1,p2} is a phosphatase
Chk1 is a kinase
C-TAK1 is a kinase
Raf1 is a kinase
cdc25A~{p1} is a phosphatase
cycA-cdk1~{p3} is a kinase
cycA-cdk2~{p2} is a kinase
cycE-cdk2~{p2} is a kinase
cdk2~{p2}-cycE~{p1} is a kinase
cycD-cdk46~{p3} is a kinase
cdk46~{p3}-cycD~{p1} is a kinase
cycA-cdk1~{p3} is a kinase
cycB-cdk1~{p3} is a kinase
cycA-cdk2~{p2} is a kinase
cycD-cdk46~{p3} is a kinase
cdk46~{p3}-cycD~{p1} is a kinase
```

```
Plk1 is a kinase
pCAF is a kinase
p300 is a kinase
HDAC1 is a phosphatase
```

It is worth noticing that in these results no compound is both a kinase and a phosphatase. The `cdc25 A` and `C` are the only phosphatases found in the whole map with `HDAC1`). The type inference also tells us that the cyclin-dependant kinases have a kinase function when in complex with a cyclin. Finally the acetylases `pCAF`, `p300` and the deacetylase `HDAC1` are detected but identified to kinases and phosphatases respectively, since the BIOCHAM syntax does not distinguish between phosphorylation and acetylation.

4 A Type System for Activation and Inhibitory Influences

4.1 Abstract Domain of Influences

Influence networks for activation and inhibition have been introduced for the analysis of gene expression in the setting of gene regulatory networks [12]. Such influence networks are in fact an abstraction of complex reaction networks, and can be applied as such to protein interaction networks. However the distinction between the influence network and the reaction network is crucial to the application of Thomas's conditions of multistationarity and oscillations [12, 13] to protein interaction network, and there has been some confusion between the two kinds of networks [14]. Here we precisely define influence networks as an abstraction of (or a type system for) reaction networks.

Definition 5. *The abstract domain of influences is the powerset of the binary relations of activation and inhibition between compounds* $\mathcal{D_I} = \mathcal{P}(\{A \textit{ activates } B \mid A, B \in \mathcal{M}\} \cup \{A \textit{ inhibits } B \mid A, B \in \mathcal{M}\})$.

The influence abstraction $\alpha : \mathcal{D_R} \rightarrow \mathcal{D_I}$ *is the function*

$$\begin{aligned}\alpha(x) = \{A \textit{ inhibits } B \quad & \mid \exists (e_i \textit{ for } S_i \Rightarrow S'_i) \in x, \\ & \quad l_i(A) > 0 \textit{ and } r_i(B) - l_i(B) < 0\} \\ \cup\{A \textit{ activates } B & \mid \exists (e_i for S_i \Rightarrow S'_i) \in x, \\ & \quad l_i(A) > 0 \textit{ and } r_i(B) - l_i(B) > 0\}\end{aligned}$$

In particular, we have the following influences for elementary reactions of complexation, modification, synthesis and degradation:

$$\begin{aligned}\alpha(\{\mathtt{A} + \mathtt{B} \Rightarrow \mathtt{C}\}) &= \{ \text{ A inhibits B, A inhibits A, B inhibits A,} \\ & \quad\ \text{B inhibits B, A activates C, B activates C}\} \\ \alpha(\{\mathtt{A} = [\mathtt{C}] \Rightarrow \mathtt{B}\}) &= \{ \text{ C inhibits A, A inhibits A, A activates B, C activates B}\} \\ \alpha(\{\mathtt{A} = [\mathtt{B}] \Rightarrow \_\}) &= \{ \text{ B inhibits A, A inhibits A}\} \\ \alpha(\{\_ = [\mathtt{B}] \Rightarrow \mathtt{A}\}) &= \{ \text{ B activates A}\}\end{aligned}$$

The inhibition loops on the reactants are justified by the negative sign in the Jacobian matrix of the differential semantics of such reactions. It is worth noting however that they are often omitted in the influence graphs considered in the literature, as well as with some other influences, according to functionality, kinetic and non-linearity considerations.

4.2 Evaluation Results

MAPK model. Let us first consider the MAPK signalling model of [9]. Fig. 1 depicts the reaction graph as a bipartite graph with round boxes for molecules and rectangular boxes for rules. Fig. 2 depicts the inferred influence graph, where activation (resp. inhibition) is materialized by plain (resp. dashed) arrows. The graph layouts of the figures have been computed in BIOCHAM by the Graphviz suite[1].

p53-Mdm2 model. In the p53-Mdm2 model of [15], the protein $Mdm2$ is localized explicitly in two possible locations: the nucleus and in the cytoplasm, and transport rules are considered. Fig. 4 depicts the reaction graph of the model.

Fig. 3 depicts the inferred influence graph. Note that $Mdm2$ in the nucleus has both an activation and an inhibitory effect on $p53\ u$. This corresponds to different influences in different regions of the phase space.

Fig. 5 depicts the core influence graph considered for the logical analysis of this model [16]. In the core influence graph, some influence are neglected, as expected, however some inhibitions, such the inhibitory effect of $p53$ on $Mdm2$ in the nucleus, are considered while they do not appear in the inferred influence graph. The reason for these omissions is the way the reaction model is written. Some inhibitory effects are indeed expressed in the kinetic expression by subtraction of, or division by, the molecular concentration of some compounds that do not appear in the rule itself. Those inhibitions are thus missed by the type inference algorithm. An example of such a rule is the following one for the inhibition of $Mdm2$ by $p53$:

```
macro(p53tot,[p53]+[p53~{u}]+[p53~{uu}]).
(kph*[Mdm2::c]/(Jph+p53tot),MA(kdeph)) for Mdm2::c <=> Mdm2~{p}::c.
```

Obviously, we cannot expect to infer such inhibitory effects from the kinetic expressions with all generality, however the model being written that way without fully decomposing all influences by reaction rules, a refinement of the abstraction function taking into account the kinetic expression is worth investigating. As an alternative, one could extend the syntax of reaction rules in order to indicate the inhibitors of the reaction, in a somewhat symmetric fashion to catalysts.

Kohn's Map. On Kohn's map, the type inference of activation and inhibition influences takes less than one second CPU time (on a PC 1,7GHz) for the complete model, showing again the efficiency of the type inference algorithm.

5 A Type System for Location Topologies

To date, models of biochemical systems generally abstract from space considerations. Models taking into account cell compartments and transport phenomena

[1] `http://www.graphviz.org/`

are thus much less common. Nevertheless, with the advent of systems biology computational tools, more and more models are refined with space considerations and transport delays, e.g. [15]. In SBML [2] level 1 version 1, locations have been introduced as purely symbolic compartments without topology. We show in this section how the topology can be inferred from the reaction rules, and checked in different models.

5.1 Abstract Domain of Location Topologies

Definition 6. *Abstract domain of neighborhood relation $\mathcal{D}_{\mathcal{N}}$ is a relation on pairs of molecules $\mathcal{M} \times \mathcal{M}$.*

Definition 7. *$\alpha : \mathcal{D}_{\mathcal{R}} \to \mathcal{D}_{\mathcal{N}}$ is defined by the union of its definition on single rules:*

$\alpha(\mathtt{E\ for\ A}_1 + \cdots + \mathtt{A}_n \mathtt{\ =>\ B}_1 + \cdots + \mathtt{B}_m)$ = *All A_i and all B_j are pairwise neighbors, and for all C_k such that $[\mathtt{C}_k]$ appears in* `E`*, C_k is a neighbor of all A_i and all B_j.*

5.2 Evaluation Results

Models from `biomodels.net`. We have taken models from the literature through the `biomodels.net` database. Of the 50 models in the current version (dated January 2006) only 13 have more than one compartment, and only 7 of those use the *outside* attribute of SBML to provide more topological insight.

The neighboring relation is inferred in these models imported in BIOCHAM, and then checked consistent with the provided *outside* relation.

For instance for calcium oscillations, we tried both the Marhl et al. model of [17] and the Borghans et al. model of [18].

In the first case (model `BIOMD0000000039.xml`), three locations are defined: the cytosol, the endoplasmic reticulum and a mitochondria, from the reactions the inferred topology is that the cytosol is neighbor of the two other locations. This correspond exactly to the information obtained from the *outside* annotations (the cytosol being marked as the outside of the two other locations).

In the second case (models BIOMD0000000043.xml to `BIOMD0000000045.xml`) we focused on the last model (*two-pool*) since it is the only one with 4 different locations: the extracellular space, the cytosol and two internal vesiculae. The location inference produces a topology where the cytosol is neighbor of all other locations. Once again this is correct w.r.t. the outside information provided in the SBML file: both vesiculae have the cytosol as outside location and the cytosol itself has the extracellular space as outside location.

These considerations show that there is some mismatch between the SBML reaction models and the choice of expressing outside vs neighborhood properties of locations. In the perspective of type checking and type inference, neighborhood relations should be preferred as they can be checked, or inferred from the reaction model, whereas the outside relation contain more information that, while helpful for the modeler as meta-data, cannot be handled automatically without abstracting it first in neighbors properties.

P53/Mdm2. The first example comes from [15]: a model of the p53/Mdm2 interaction with two locations where the transport between cytoplasm and nucleus is necessary to explain some time delays observed in the mutual repression of these proteins.

```
biocham: load_biocham('EXAMPLES/locations/p53Mdm2.bc').
...
biocham: show_neighborhood.
c and n are neighbors
```

In this precise case, the model as published does not systematically use the volume ratio in the kinetics. The transcription and type-checking of the model showed that if one wanted to keep the background degradation rate of $Mdm2$ (without DNA damage) independent of the location, one obtains different kinetics than those of the published model. In this case a formal transcription in BIOCHAM (or SBML) provided a supplementary model-validation step.

Delta and Notch Model. The next example is adapted from [19]. The Delta and Notch proteins are crucial to the cell fate in several different organisms. A population of neighboring cells (here we chose a square grid) is represented through locations and the model allows to observe the salt-and-pepper coloring (corresponding to high Delta-low Notch/low Delta-high Notch) typical of the Delta-Notch lateral inhibition based differentiation. The signaling pathways are simplified to the extreme to take into account only the direct effect of Delta and Notch expression on the local and neighboring cells. This example would thus not provide a good basis for the abstraction of section 4.

Depending on the abstraction chosen we obtain figure 6 and 7. In the first case the abstraction used is not the one given in section 5.1 but

Definition 8. *$\alpha : \mathcal{D}_{\mathcal{R}} \to \mathcal{D}_{\mathcal{N}}$ is defined by the union of its definition on single rules:*
$\alpha(\texttt{E for A}_1 + \cdots + \texttt{A}_n \texttt{ => B}_1 + \cdots + \texttt{B}_m) =$ *All A_i, all B_j, and all C_k such that* `[C`$_k$`]` *appears in* `E`*, are pairwise neighbors.*

This was indeed a reasonable candidate for an abstraction, but proved too coarse on some examples since co-modifiers are often put in the kinetic expression of a single rule for simplification purposes.

6 Conclusion

We have shown that the framework of abstract interpretation applies to the formalization of some abstractions commonly used in systems biology, and to the implementation of linear-time type checking as well as type inference algorithms.

In the three type systems studied in this paper, for protein functions, activation and inhibitory influences, and location topologies respectively, the analyses are based on static information gained directly from the syntax of reaction rules, without considering their formal semantics, nor their precise dynamics. It is worth noting that this situation also occurs in program analysis where the syntax of programs may capture a sufficient part of the semantics for many analyses. Here, it is remarkable that such simple analyses already provide useful information on biological models, independently from their dynamics for which different definitions are considered (discrete, continuous, stochastic, etc.) [7].

The formal definition of the influence graph as an abstraction of the reaction model eliminates some confusion that exists in the use of Thomas's conditions [12, 13] for the analysis of reaction models [14]. Such a formalization shows also that the influence graphs usually considered in the literature are further abstractions obtained by forgetting some influences, based on non-linearity considerations [20]. Some inhibitions may also be missing in the inferred influences when they are hidden in the kinetic expressions of the reactions and do not appear explicitly in the reactants. This suggests either to refine the abstraction function to take into account the kinetic expression when possible, or to extend the syntax of reactions in order to make explicit such inhibitory effects, in a symmetric fashion to catalysts for activations. In SBML there is actually an unique symmetrical notion of *Modifiers* which is not sufficient to infer the influence graph.

Similarly, the inference of protein functions and of location neighborhood have shown that the static analysis of reaction models by type inference provides both accurate and useful information. They also provide some guidelines for the extensions of biochemical reaction languages, like for instance in SBML considering neighborhood rather than outside properties, and introducing a syntax for the modification of compounds, and in BIOCHAM differentiating phosphorylation from other forms of modifications like acetylation.

Acknowledgement. This work benefited from partial support of the Network of Excellence REWERSE of the European Union.

References

1. Cardelli, L.: Typeful programming. In Neuhold, E.J., Paul, M., eds.: Formal Description of Programming Concepts. Springer-Verlag, Berlin (1991) 431–507
2. Hucka, M., et al.: The systems biology markup language (SBML): A medium for representation and exchange of biochemical network models. Bioinformatics **19** (2003) 524–531
3. Fages, F., Soliman, S., Chabrier-Rivier, N.: Modelling and querying interaction networks in the biochemical abstract machine BIOCHAM. Journal of Biological Physics and Chemistry **4** (2004) 64–73
4. Cousot, P., Cousot, R.: Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: POPL'77: Proceedings of the 6th ACM Symposium on Principles of Programming Languages, New York, ACM Press (1977) 238–252 Los Angeles.
5. Cousot, P.: Types as abstract interpretation (invited paper). In: POPL'97: Proceedings of the 24th ACM Symposium on Principles of Programming Languages, New York, ACM Press (1997) 316–331 Paris.

6. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. Journal of Physical Chemistry **81** (1977) 2340–2361
7. Calzone, L., Chabrier-Rivier, N., Fages, F., Soliman, S.: Machine learning biochemical networks from temporal logic properties. Transactions on Computational Systems Biology (2006) CMSB'05 Special Issue (to appear).
8. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G.: Gene ontology: tool for the unification of biology. Nature Genetics **25** (2000) 25–29
9. Levchenko, A., Bruck, J., Sternberg, P.W.: Scaffold proteins may biphasically affect the levels of mitogen-activated protein kinase signaling and reduce its threshold properties. PNAS **97** (2000) 5818–5823
10. Kohn, K.W.: Molecular interaction map of the mammalian cell cycle control and DNA repair systems. Molecular Biology of the Cell **10** (1999) 2703–2734
11. Chabrier-Rivier, N., Chiaverini, M., Danos, V., Fages, F., Schächter, V.: Modeling and querying biochemical interaction networks. Theoretical Computer Science **325** (2004) 25–44
12. Thomas, R., Gathoye, A.M., Lambert, L.: A complex control circuit : regulation of immunity in temperate bacteriophages. European Journal of Biochemistry **71** (1976) 211–227
13. Soulé, C.: Graphic requirements for multistationarity. ComplexUs **1** (2003) 123–133
14. Markevich, N.I., Hoek, J.B., Kholodenko, B.N.: Signaling switches and bistability arising from multisite phosphorylation in protein kinase cascades. Journal of Cell Biology **164** (2005) 353–359
15. Ciliberto, A., Novák, B., Tyson, J.J.: Steady states and oscillations in the p53/mdm2 network. Cell Cycle **4** (2005) 488–493
16. Kaufman, M.: Private communication. (2006)
17. Marhl, M., Haberichter, T., Brumen, M., Heinrich, R.: Complex calcium oscillations and the role of mitochondria and cytosolic proteins. BioSystems **57** (2000) 75–86
18. Borghans, J., Dupont, G., Goldbeter, A.: Complex intracellular calcium oscillations: a theoretical exploration of possible mechanisms. Biophysical Chemistry **66** (1997) 25–41
19. Ghosh, R., Tomlin, C.: Lateral inhibition through delta-notch signaling: A piecewise affine hybrid model. In Springer-Verlag, ed.: Proceedings of the 4th International Workshop on Hybrid Systems: Computation and Control, HSCC'01. Volume 2034 of Lecture Notes in Computer Science., Rome, Italy (2001) 232–246
20. Thomas, R., Kaufman, M.: Multistationarity, the basis of cell differentiation and memory. Chaos **11** (2001) 170–195

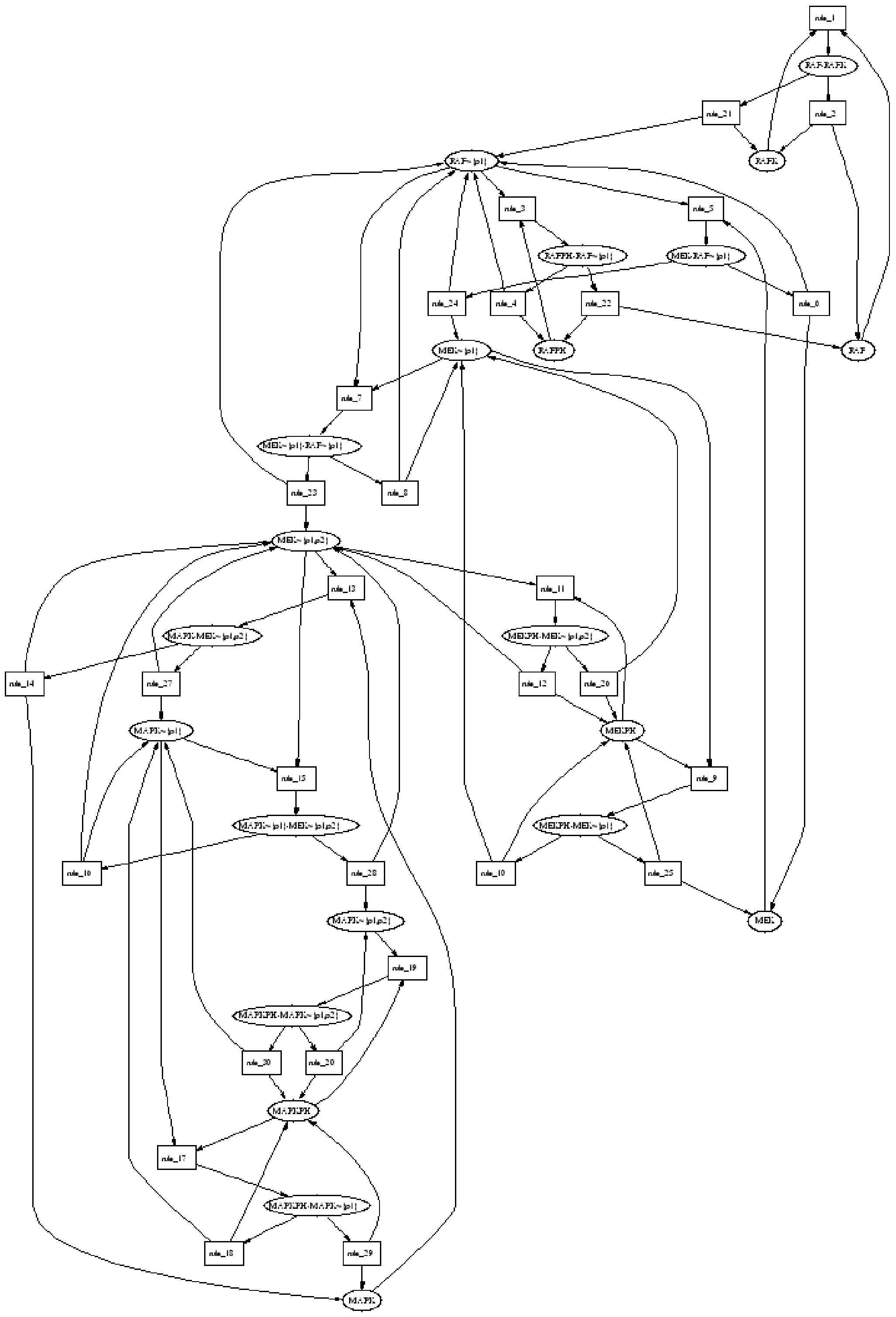

Fig. 1. Reaction graph of the MAPK model

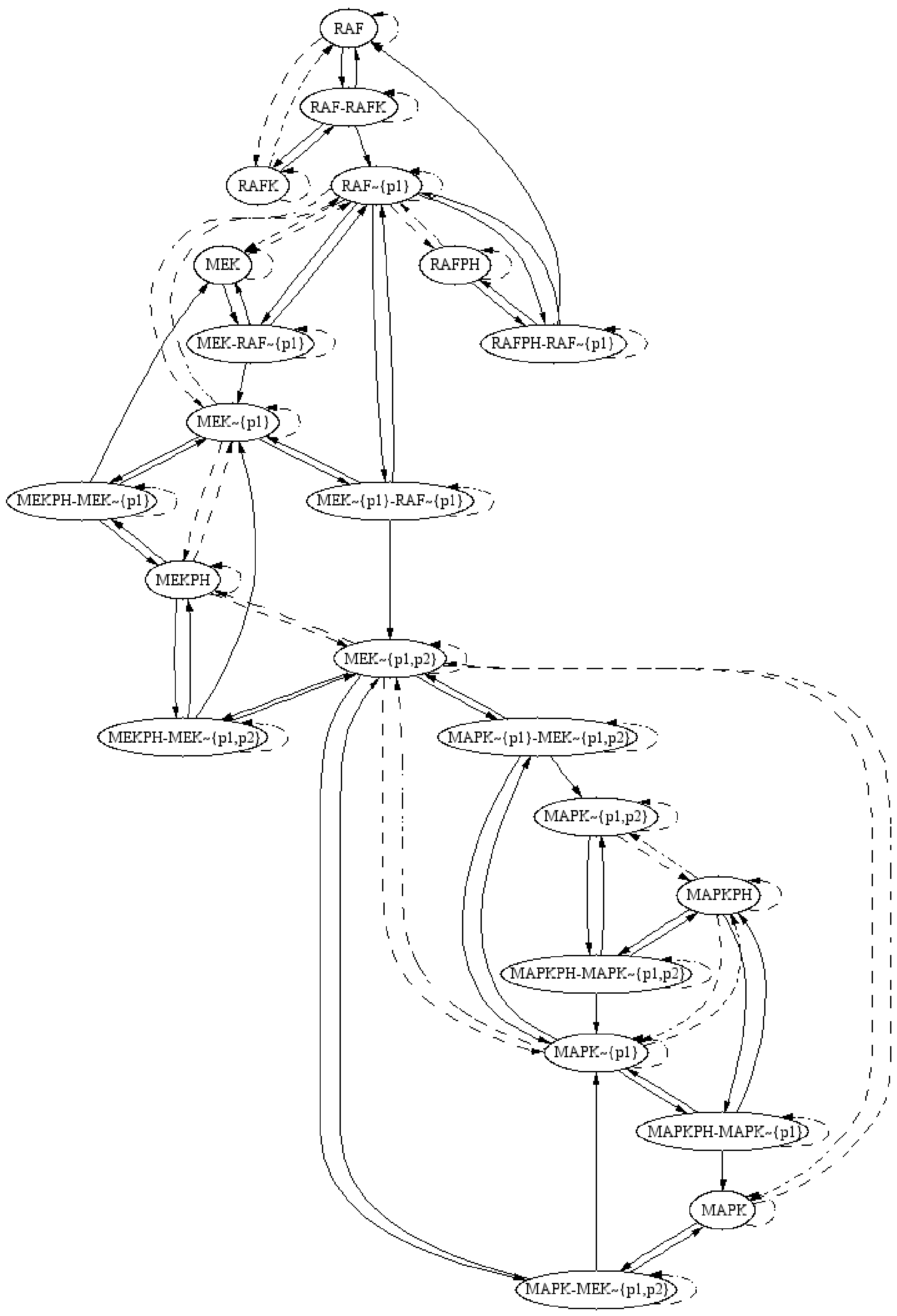

Fig. 2. Inferred influence graph of the MAPK model

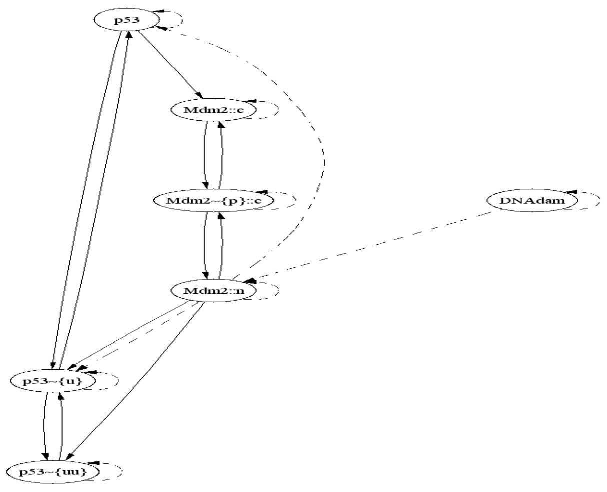

Fig. 3. Inferred influence graph of the p53-Mdm2 model

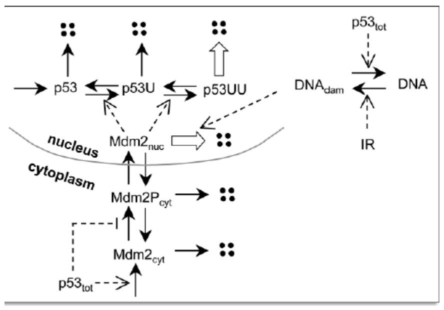

Fig. 4. Original reaction graph considered in [15] for the p53-Mdm2 model.

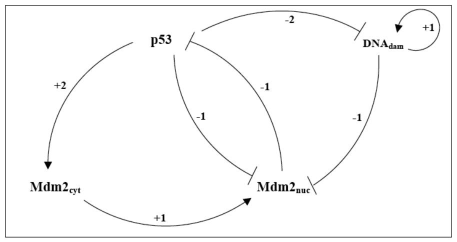

Fig. 5. Core influence graph.

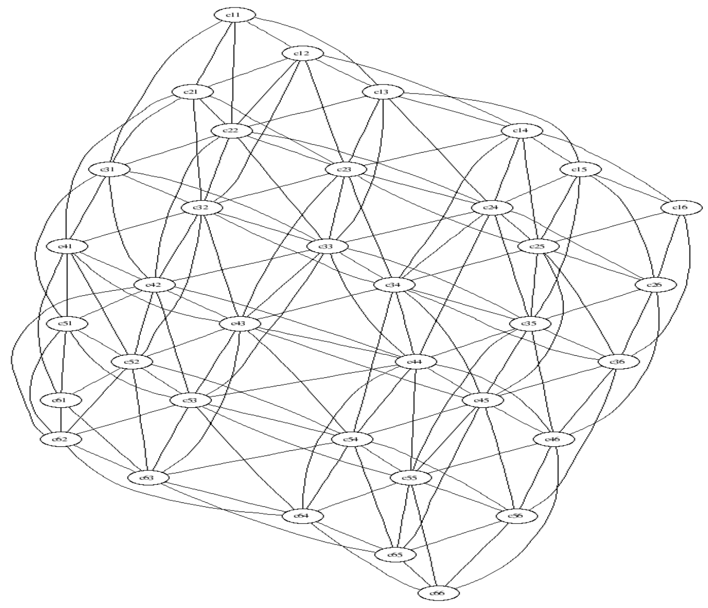

Fig. 6. Delta-Notch square cell grid inferred in a 6x6 model, with modifiers, reactants and products as pairwise neighbors

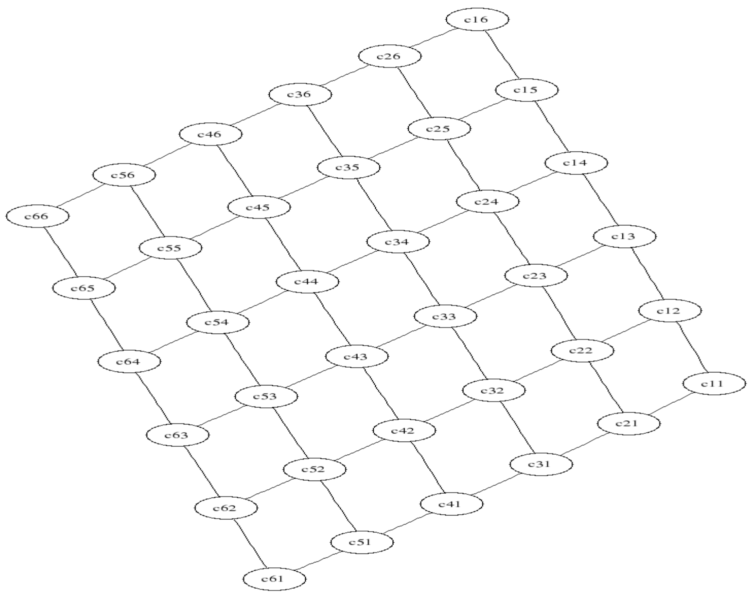

Fig. 7. Delta-Notch square cell grid inferred in a 6x6 model, without modifier-modifier neighborhood