Molecular interaction automated maps

Robert Demolombe, Luis Fariñas del Cerro, Naji Obeid

Molecular Interaction Automated Maps

Robert Demolombe, Luis Fariñas del Cerro, and Naji Obeid*

Université de Toulouse and CNRS, IRIT, Toulouse, France
robert.demolombe@orange.fr, luis.farinas@irit.fr, naji.obeid@irit.fr

Abstract. Determining the role of genomes and their interference in a cell life cycle has been at the center of metabolic network researches and experiments. Logical representations of such networks aim to guide scientists in their reasoning in general and to help them find inconsistencies and contradictions in their results in particular. This paper presents a new logical model capable of describing both positive (activation) and negative (inhibition) reactions of metabolic pathways based on a fragment of first order logic. An efficient automated deduction method will also be introduced, based on a translation procedure that transform first order formulas into quantifier free formulas. Then questions can either be answered by deduction to predict reaction results or by abductive reasoning to infer reactions and protein states.

Keywords: Metabolic pathways, logical model, inhibition, automated reasoning.

1 Introduction

Cells in general and human body cells in particular incorporates a large series of intracellular and extracellular signalings, notably protein activations and inhibitions, that specify how they should carry out their functions. Networks formed by such biochemical reactions, often referred as *pathways*, are at the center of a cell's existence and they range from simple and chain reactions and counter reactions to simple and multiple regulations and auto regulations, that can be formed by actions defined in Figure 1. Cancer, for example, can appear as a result of a pathology in the cell's pathway, thus, the study of signalization events appears to be an important factor in biological, pharmaceutical and medical researches [14, 11, 7]. However, the complexity of the imbrication of such processes makes the use of a physical model as a representation seem complicated.

In the last couple of decades, scientists that used artificial intelligence to model cell pathways [10, 9, 16, 17, 6, 21, 15] faced many problems especially because information about biological networks contained in knowledge bases is generally incomplete and sometimes uncertain and contradictory. To deal with such issues, abduction [3] as

* LNCSR Scholar

theory completion [12] is used to revise the state of existing nodes and add new nodes and arcs to express new observations. Languages that were used to model such networks had usually limited expressivity, were specific to special pathways or were limited to general basic functionalities. We, in this work, present a fragment of first order logic [19] capable of representing node states and actions in term of positive and negative relation between said nodes. Then an efficient proof theory for these fragments is proposed. This method can be extended to define an abduction procedure which has been implemented in SOLAR [13], an automated deduction system for consequence finding. A previous version of this work has been presented in BIOCOMP'13 [5].
For queries about the graph that contains negative actions, it is assumed that we have a complete representation of the graph. The consequence is that the negation is evaluated according to its definition in classical logic instead of some non-monotonic logic. This approach guarantees a clear meaning of answers. Since the completion of the graph is formalized a la Reiter we used the equality predicate. It is well known that equality leads to very expensive automated deductions. This problem has been resolved by replacing completed predicates by their extensions where these predicates are used to restrict the domain of quantified variables. The result of this translation is formulated without variables where consequences can be derived very fast. This is one of the main contributions of this paper.

The rest of this paper is organized as follows. Section 2 presents a basic language and axiomatic capable of describing general pathways, ans shows how it is possible to extends this language and axiomatic to address specific and real life examples. Section 3 defines a translation procedure capable of eliminating first order variables and equality predicates and shows how it can be applied to derive new axiomatic that can be used in the automated deduction process in SOLAR. Section 4 provide some case studies, and finally section 5 gives a summary and discusses future works.

2 Logical Model

In this section we will present a basic language capable of modeling some basic positive and negative interaction between two or more proteins in some pathway. We will first focus on the stimulation and inhibition actions, points (g) and (i) of Figure 1, and then show how this language can be modified to express the different other actions described in the same figure.

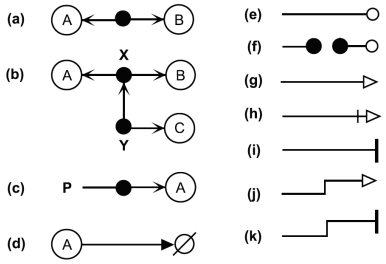

Fig. 1: Symbol definitions and map conventions.

(a) Proteins A and B can bind to each other. The node placed on the line represents the A:B complex. (b) Multimolecular complexes: x is A:B and y is(A:B):C. (c) Covalent modification of protein A. (d) Degradation of protein A. (e) Enzymatic stimulation of a reaction. (f) Enzymatic stimulation in trans. (g) General symbol for stimulation. (h) A bar behind the arrowhead signifies necessity. (i) General symbol for inhibition. (j) Shorthand symbol for transcriptional activation. (k) Shorthand symbol for transcriptional inhibition.

2.1 Formal Language

Let's consider a fragment of first order logic with some basic predicates, boolean connectives ($\wedge$) and, ($\vee$) or, ($\neg$) negation, ($\rightarrow$) implication, ($\leftrightarrow$) equivalence, ($\exists$) existential and ($\forall$) universal quantifiers, and ($=$) equality.

The basic state predicates are $A(x)$, $I(x)$ and $P(x)$ respectively meaning that the protein x is *Active*, *Inhibited* or *Present*. And the basic state axioms that indicate that a certain protein x can never be in both *Active* and *Inhibited* states at the same time are:

$$A(x) \rightarrow \neg I(x) \ . \qquad (1) \qquad \neg A(x) \rightarrow I(x) \vee \neg P(x) \ . \qquad (2)$$

$$I(x) \rightarrow \neg A(x) \ . \qquad (3) \qquad \neg I(x) \rightarrow A(x) \vee \neg P(x) \ . \qquad (4)$$

An interaction between two or more different proteins is expressed by a predicate of the form $Action(protein_1, ..., protein_n)$. In our case we are interested by the simple *Activation* and *Inhibition* actions that are defined by the following predicates:

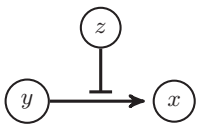

Fig. 2: Activation

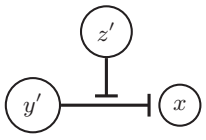

Fig. 3: Inhibition

– $CAP(y, x)$: the *Capacity of Activation* expresses that the protein y has the capacity to activate the protein x.

– $CICAP(z, y, x)$: the *Capacity to Inhibit the Capacity of Activation* expresses that the protein z has the capacity to inhibit the capacity of the activation of x by y.
– $CIP(y', x)$: the *Capacity to Inhibit a Protein* expresses that the protein y' has the capacity to inhibit the protein x.
– $CICIP(z', y', x)$: the *Capacity to Inhibit the Capacity of Inhibition of a Protein* expresses that the protein z' has the capacity to inhibit the capacity of inhibition of x by y'.

In the next section we will define the needed axioms that will be used to model the *Activation* and *Inhibition* actions.

2.2 Action axioms

Giving the fact that a node can acquire the state active or inhibited depending on different followed pathways, one of the issues answered by abduction is to know which set of proteins is required to be active of inhibited for our target protein be active or inhibited.

Axiomatic of activation: A protein x is active if there exists at least one *active* protein y such as $CAP(y, x)$ that has the capacity to activate x, and for every protein z that has the capacity to inhibit the capacity of activation of x by y, such as $CICAP(z, y, x)$, z is *not active*. (Figure 2)

$$\forall x(\exists y(A(y) \wedge CAP(y, x) \wedge \forall z(CICAP(z, y, x) \rightarrow \neg A(z)))\wedge \rightarrow A(x)) \ . \qquad (5)$$

Axiomatic of inhibition A protein x is inhibited if there exists at least one *active* protein y' such as $CIP(y', x)$ that has the capacity to inhibit x, and for every protein z' that has the capacity to inhibit the capacity of inhibition of x by y', such as $CICIP(z', y', x)$, z' is *not active*. (Figure 3)

$$\forall x(\exists y'(A(y') \wedge CIP(y', x) \wedge \forall z'(CICIP(z', y', x) \rightarrow \neg A(z')))\wedge \rightarrow I(x)) \ . \qquad (6)$$

2.3 Extension with new states and actions

The basic language defined in 2.1 and 2.2 can be easily extended to express different and more precise node statuses and actions. For example the action of *phosphorylation* can be defined by the following predicates:

– $CP(z, y, x)$: the *Capacity of Phosphorylation* expresses that the protein z has the capacity to phosphorylate the protein y on a certain site, knowing that x is the result of said phosphorylation.

– $CICP(t, z, y, x)$: the *Capacity to Inhibit the Capacity of Phosphorylation* expresses that the protein t has the capacity to inhibit the capacity of the phosphorylation of y by z leading to x.

We can now define the new phosphorylation axiom as:

$$\forall x(\exists y_1 \exists y_2(A(y_1) \wedge A(y_2) \wedge CP(y_1, y_2, x) \wedge \forall z(CICP(z, y_1, y_2, x) \rightarrow \neg A(z))) \wedge \rightarrow A(x)) \ .$$

3 Automated Deduction Method

In this section we define a fragment of first order logic with constants and equality, and without functions, that is a special case of *Evaluable* formulas [2] and *Domain Independent* formulas [22], and a generalization of *Guarded* formulas [1] called *Restricted* formulas. The properties of this fragment allow us to define a procedure capable of eliminating the quantifiers in this fragment, in other words to transform the first order formulas in formulas without variables, in order to obtain an efficient automated deduction procedure with these fragments.

Definition 1. *Domain formulas are defined by the following grammar:*

$$\delta ::= P(\overline{x}, \overline{c}) | \varphi \vee \psi | \varphi \wedge \psi | \varphi \wedge \neg \psi \ . \tag{7}$$

Where variables $\overline{x}$ and constants $\overline{c}$ denote $x_1, ..., x_n$ and $c_1, ..., c_m$ respectively. The set of free variables in φ is the same as the set of free variables in ψ for $\varphi \vee \psi$, and the set of free variables in ψ is included in the set of free variables in φ for $\varphi \wedge \neg \psi$.

Definition 2. *Restricted formulas are formulas without free variables defined by the following grammar:*

$$\delta ::= \forall \overline{x}(\varphi \rightarrow \psi) | \exists \overline{x}(\varphi \wedge \psi) \ . \tag{8}$$

Where φ is a domain formula and ψ is either a restricted formula or a formula without quantifiers, and every variable appearing in a restricted formula must appear in a domain formula. The set of variables in $\overline{x}$ is included in the set of free variables in φ; The same goes for ψ.
Examples: $\forall x(P(x) \rightarrow Q(x))$. $\forall x(P(x) \rightarrow \exists y(Q(y) \wedge R(x, y)))$.

Definition 3. *A completion formula is a formula of the following form:*

$$\begin{aligned} \forall x_1, ..., x_n \ (P(x_1, ..., x_n, c_1, ..., c_p) \leftrightarrow & ((x_1 = a_{1_1} \wedge ... \wedge x_n = a_{1_n}) \vee ... \vee \\ & (x_1 = a_{m_1} \wedge ... \wedge x_n = a_{m_n}))) \ . \end{aligned} \tag{9}$$

Where P is a predicate symbol of arity $n+p$. Completion formulas are similar to the completion axioms defined by Reiter in [18] where the implication is substituted by an equivalence.

Definition 4. *Given a domain formula φ and a set of completion formulas $\alpha_1, ..., \alpha_n$ such that for each predicate symbol in φ there exists a completion formula α for this predicate symbol, we say that the set of completion formulas $\alpha_1, ..., \alpha_n$ covers φ and will be noted $C(\varphi)$.*

Definition 5. *Given a domain formula φ and a saturated completion set denoted $C(\varphi)$ of φ, we define the domain of the variables of φ with respect to $C(\varphi)$, denoted $D(\mathcal{V}(\varphi), C(\varphi))$, where $\mathcal{V}(\varphi)$ represents the variables of φ, recursively as follows:*

- if φ is of the form $P(\overline{x}, \overline{c})$, and in $C(\varphi)$ we have a formula of the form:

$$\forall x_1, ..., x_n(P(x_1, ..., x_n, c_1, ..., c_p) \leftrightarrow ((x_1 = a_{1_1} \wedge ... \wedge x_n = a_{1_n}) \vee ... \vee (x_1 = a_{m_1} \wedge ... \wedge x_n = a_{m_n}))) \ .$$

$$\text{then } D(\mathcal{V}(\varphi), C(\varphi)) = \{< a_{1_1}, ..., a_{1_n} >, ..., < a_{m_1}, ..., a_{m_n} >\} \ . \quad (10)$$

- if φ is of the form $\varphi_1 \vee \varphi_2$ then:

$$D(\mathcal{V}(\varphi_1 \vee \varphi_2), C(\varphi_1 \vee \varphi_2)) = D(\mathcal{V}(\varphi_1), C(\varphi_1)) \cup D(\mathcal{V}(\varphi_2), C(\varphi_2)) \ . \quad (11)$$

- if φ is of the form $\varphi_1 \wedge \varphi_2$ then:

$$D(\mathcal{V}(\varphi_1 \wedge \varphi_2), C(\varphi_1 \wedge \varphi_2)) = D(\mathcal{V}(\varphi_1), C(\varphi_1)) \otimes_c D(\mathcal{V}(\varphi_2), C(\varphi_2)) \ . \quad (12)$$

  Where $\otimes_c$ [22] is a join operator and c is a conjunction of equalities of the form $i = j$ the same variable symbol appears in $\varphi_1 \wedge \varphi_2$ in position i in φ_1 and in position j in φ_2.
- if φ is of the form $\varphi_1 \wedge \neg\varphi_2$ then:

$$D(\mathcal{V}(\varphi_1 \wedge \neg\varphi_2), C(\varphi_1 \wedge \neg\varphi_2)) = D(\mathcal{V}(\varphi_1), C(\varphi_1)) \setminus D(\mathcal{V}(\varphi_1 \wedge \varphi_2), C(\varphi_1 \wedge \varphi_2)). \quad (13)$$

  Where $\setminus$ denotes the complement of the domain of each shared variable of φ_2 with respect to φ_1.

Example 1. Considering the three domains formulas $P(x)$, $Q(x)$, $R(x, y)$ and their corresponding completion formulas as following:

$\forall x(P(x) \rightarrow x = a \vee x = d)$ we have $D(\mathcal{V}(P(x)), C(P(x))) = \{< a >, < d >\}$.

$\forall x(Q(x) \rightarrow x = b \vee x = c)$ we have $D(\mathcal{V}(Q(x)), C(Q(x))) = \{< b >, < c >\}$.

$\forall x, y(R(x, y) \rightarrow (x = a \wedge y = b) \vee (x = a \wedge y = c) \vee (x = b \wedge y = e))$ we have $D(\mathcal{V}(R(x, y)), C(R(x, y))) = \{< a, b >, < a, c >, < b, e >\}$.

If we have:

$\varphi_1 = P(x) \vee Q(x)$ then $D(\mathcal{V}(\varphi_1), C(\varphi_1)) = \{< a >, < b >, < c >, < d >\}$.
$\varphi_2 = R(x, y) \vee (P(x) \wedge Q(x))$ then $D(\mathcal{V}(\varphi_2), C(\varphi_2)) = \{< a, b >, < a, c >\}$.
$\varphi_3 = R(x, y) \wedge \neg P(x)$ then $D(\mathcal{V}(\varphi_3), C(\varphi_3)) = \{< b, e >\}$.

Quantifier elimination procedure

Let φ be a restricted formula of the following forms: $\forall \overline{x}(\varphi_1(\overline{x}) \rightarrow \varphi_2(\overline{x}))$ or $\exists \overline{x}(\varphi_1(\overline{x}) \wedge \varphi_2(\overline{x}))$, let $C(\varphi_1(\overline{x}))$ a set of completion formulas for φ_1, then we define recursively a translation $T(\varphi, C(\varphi))$, allowing to replace universal (existential) quantifiers by conjunction (disjunction) of formulas where quantified variables are substituted by constants as follows:

– if $D(\mathcal{V}(\varphi_1), C(\varphi_1)) = \{< \overline{c_1} >, ..., < \overline{c_n} >\}$ with $n > 0$:

$T(\forall \overline{x}(\varphi_1(\overline{x}) \rightarrow \varphi_2(\overline{x})), C(\varphi)) = T(\varphi_2(\overline{c_1}), C(\varphi_2(\overline{c_1}))) \wedge ... \wedge T(\varphi_2(\overline{c_n}), C(\varphi_2(\overline{c_n})))$.
$T(\exists \overline{x}(\varphi_1(\overline{x}) \wedge \varphi_2(\overline{x})), C(\varphi)) = T(\varphi_2(\overline{c_1}), C(\varphi_2(\overline{c_1}))) \vee ... \vee T(\varphi_2(\overline{c_n}), C(\varphi_2(\overline{c_n})))$.

– if $D(\mathcal{V}(\varphi_1), C(\varphi_1)) = \varnothing$:
$T(\forall \overline{x}\ (\varphi_1(\overline{x}) \rightarrow \varphi_2(\overline{x}))\ ,\ C(\varphi)) = True.$ $\quad$ $T(\exists \overline{x}\ (\varphi_1(\overline{x}) \wedge \varphi_2(\overline{x}))\ ,\ C(\varphi)) = False.$

Note 1. It is worth nothing that in this translation process each quantified formula is replaced in the sub formulas by constants. The consequence is that if a sub formula of a restricted formula is of the form $\forall \overline{x}(\varphi_1(\overline{x}) \rightarrow \varphi_2(\overline{x}, \overline{y}))$ or $\exists \overline{x}(\varphi_1(\overline{x}) \wedge \varphi_2(\overline{x}, \overline{y}))$ where the quantifiers $\forall \overline{x}$ or $\exists \overline{x}$ are substituted by their domain values, the variables in $\overline{y}$ must have been already substituted by its corresponding constants.

Then in the theory $\mathcal{T}$ in which we have the axioms of equality and axioms of the form $\neg(a = b)$ for each constant a and b representing different objects, which are called unique name axioms by Reiter in [18], we have the following main theorem:

Theorem 1. *Let φ be a restricted formula, and $C(\varphi)$ a completion set of formulas of the domain formulas of φ, then:*

$$\mathcal{T},\ C(\varphi) \vdash \varphi \leftrightarrow T(\varphi, C(\varphi))\ . \tag{14}$$

Proof. The proof consists of applying induction on the number of domain formulas in a restricted formula to prove that the theorem holds for any number domain formulas.

Example 2.

Let's consider the case where a protein b has the capacity to activate another protein a, and that two other proteins c_1 and c_2 have the capacity to inhibit the capacity of activation of a by b. This proposition can be expressed by the following completion axioms:

- $\forall y(CAP(y, a) \leftrightarrow y = b)$: Where b is the only protein that has the capacity to activate a.
- $\forall z(CICAP(z, b, a) \leftrightarrow z = c_1 \vee z = c_2)$: Where c_1 and c_2 are the only proteins that have the capacity to inhibit the capacity of activation of a by b.

Using the activation axiom defined in section 2 and the translation procedure, we can deduce $A(b) \wedge \neg A(c_1) \wedge \neg A(c_2) \wedge \rightarrow A(a)$. Which means that the protein a is active if the protein b is active and the proteins c_1, c_2 are not active.

Let's also consider that a protein d has the capacity to inhibit the protein a and that there is no proteins capable of inhibiting the capacity of inhibition of a by d. This proposition can be expressed by the following completion axioms:

- $\forall y(CIP(y, a) \leftrightarrow y = d)$: Where d is the only protein that has the capacity to inhibit a.
- $\forall z(CICIP(z, d, a) \leftrightarrow false)$: Where there are no proteins capable of inhibiting the capacity of inhibition of a by d.

Using the previous inhibition axiom and these completion axioms we can deduce $A(d) \rightarrow I(a)$. Which means that the protein a is inhibited if the protein d is active.

4 Automated Reasoning

From what we defined in sections 2 and 3, the resulting translated axioms are of the following type $conditions \rightarrow results$, and can be chained together to create a series of reactions forming our pathway. Then questions of two different types can be answered using *abductive* or *deductive* reasoning.

(a) Questions answered by *abduction* looks for minimal assumptions (H) that must be added to the knowledge base (T) to derive that a certain fact (C) is true. A question can be of the following form: *what are the proteins and their respective states (active or inhibited) that should be present in order to derive that a certain protein is active or inhibited.*

(b) Questions answered from an abduced hypothesis H, that we call test basis and will be noted TB_H, are minimal consequences that can be derived by *deduction* over T and H, knowing that they are not direct consequences of T nor they can be deduced by H.

Both H and TB_H types of questions can be addressed in SOLAR (SOL for Advanced Reasoning) [13] a first-order clausal consequence finding system based on SOL (Skip Ordered Linear) tableau calculus [8, 20]. We will now present two examples corresponding to those two types of questions.

Example 3.

In the following we are going to show an example, based on figure 4, demonstrating abduction type queries where three coherent pathways have been found [11]. From section 2.3 we can define new predicates to suit the needs of the pathway, as the *Capacity of Binding* $CB(z, y, x)$ and the *Capacity to Inhibit the Capacity of Binding* $CICB(t, z, y, x)$. These new predicates can be used to model the binding between p53 and Bak using the predicate $CP(p53, bak, p53\_bak)$ where p53_bak is the complex formed by such binding.

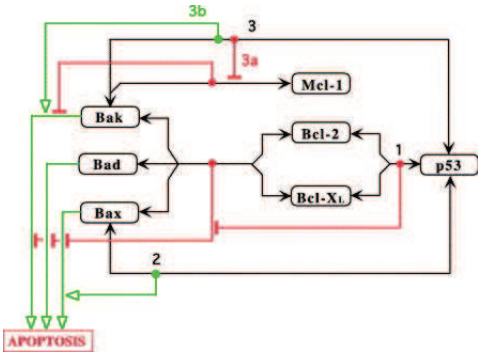

Fig. 4: Mitochondrial apoptosis induced by p53 independently of transcription

With these new predicates, new axioms can be defined that would enrich the descriptive capacities of the old ones, as seen in 2.3. Then the translation procedure applied to these axioms and to the completion axioms can be of the following form:

1. $A(p53) \wedge A(bak) \rightarrow A(bak\_p53)$. Where bak_p53 is the result of the binding between p53 and Bak.
2. $A(bak\_p53) \rightarrow I(bak\_mcl)$. Where bak_mcl is the result of binding between Bak and Mcl-1.
3. $A(bak\_p53) \wedge \neg A(b\_complex) \wedge \neg A(bak\_mcl) \rightarrow A(apoptosis)$. Where $b\_complex$ is result of the binding between Bcl-2, Bcl-XL, Bak, Bad, and Bax.
4. $A(bak) \wedge \neg A(b\_complex) \wedge \neg A(bak\_mcl) \rightarrow A(apoptosis)$
5. $A(p53) \wedge A(bcl) \rightarrow A(p53\_bb\_complex)$. Where bcl represents Bcl-2 and Bcl-XL. $p53\_bb\_complex$ is the result of binding between p53, Bcl-2 and Bcl-XL.

6. $A(p53\_bb\_complex) \rightarrow I(b\_complex)$
7. $A(bax) \wedge \neg A(b\_complex) \rightarrow A(apoptosis)$
8. $A(p53) \wedge A(bax) \wedge \neg A(b\_complex) \rightarrow A(apoptosis)$
9. $A(bad) \wedge \neg A(b\_complex) \rightarrow A(apoptosis)$

If we want to know what are the proteins and their respective states that should be present in order to derive that the cell reached apoptosis, the answer is given by applying abduction over the previous set of compiled clauses. In the set of consequences returned by SOLAR we can find the following:

- $A(p53) \wedge A(bcl) \wedge A(bak)$: is a plausible answer, because p53 can bind to Bcl giving the $p53\_bb\_complex$, which can in return inhibit the $b\_complex$ that is responsible of inhibiting the capacity of Bak to activate the cell's apoptosis.
- Another interpretation of the previous answer is that p53 can also bind to Bak giving the bak_p53 protein, which can in return inhibit the bak_mcl responsible of inhibiting the capacity of Bak to activate the cell's apoptosis. bak_p53 can also stimulate Bak to reach apoptosis. Without forgetting that $p53\_bb\_complex$ inhibit $b\_complex$.

Example 4.

Let's consider the case where proteins b and c have the capacity to activate the protein a, b can also inhibit d, and e can inhibit b. This proposition can be expressed by the following completion axioms T:

$$A(b) \rightarrow A(a). \;\; A(c) \rightarrow A(a). \;\; A(b) \rightarrow I(d). \;\; A(e) \rightarrow I(b).$$

In order to derive $A(a)$, one the following hypotheses H should be considered: $A(b)$ or $A(c)$. For $H = A(b)$ we can deduce the following $TB_{A(b)}$ consistency conditions: $\neg A(e)$ and $\neg A(d)$, that describe that for $A(b)$ to be consistent with the main proposition, which is $A(a)$, as a condition the protein e should not be active, and as a result the protein d is inhibited (not active). These new conditions can help us reason about consistency because if we know, by means of scientific experiments or as a result of some observations, that either d or e is active, this means that b is not active, which leaves us with c as the only protein that is activates a.

5 Conclusion

A new language has been defined in this paper capable of modeling both positive and negative causal effects between proteins in a metabolic pathway. We showed

how this basic language can be extended to include more specific actions that describes different relations between proteins. These extensions are important in this context, because there is always the possibility that new types of actions are discovered through biological experiments. We later showed how the axioms defined in such languages can be compiled against background knowledge, in order to form a new quantifier free axioms that could be used in either deduction or abduction reasoning. Although the first order axioms can be also well used to answer queries by deduction or abduction methods, the main advantage of translated axioms is their low computation time needed in order to derive consequences.

Future works can focus on extending the language used to define domain formulas, introducing the notion of time and quantities in the model. Trying to get as precise as possible in describing such pathways can help biologists discover contradictory informations and guide them during experiments knowing how huge the cells metabolic networks have become. One of the extensions that can also be introduced is the notion of *Aboutness* [4] that can limit and focus search results to what seems relevant to a single or a group of entities (proteins).

References

1. Hajnal Andréka, István Németi, and Johan van Benthem. Modal languages and bounded fragments of predicate logic. *Journal of Philosophical Logic*, 27(3):217–274, 1998.
2. Robert Demolombe. Syntactical characterization of a subset of domain-independent formulas. *J. ACM*, 39(1):71–94, 1992.
3. Robert Demolombe and Luis Fariñas del Cerro. An Inference Rule for Hypothesis Generation. In *Proc. of International Joint Conference on Artificial Intelligence*, Sydney, 1991.
4. Robert Demolombe and Luis Fariñas del Cerro. Information about a given entity: From semantics towards automated deduction. *J. Log. Comput.*, 20(6):1231–1250, 2010.
5. Robert Demolombe, Luis Fariñas del Cerro, and Naji Obeid. A logical model for metabolic networks with inhibition. *BIOCOMP'13*, 2013.
6. Martin Erwig and Eric Walkingshaw. Causal reasoning with neuron diagrams. In *Proceedings of the 2010 IEEE Symposium on Visual Languages and Human-Centric Computing*, VLHCC '10, pages 101–108, Washington, DC, USA, 2010. IEEE Computer Society.
7. V Glorian, G Maillot, S Poles, J S Iacovoni, G Favre, and S Vagner. Hur-dependent loading of mirna risc to the mrna encoding the ras-related small gtpase rhob controls its translation during uv-induced apoptosis. *Cell Death Differ*, 18(11):1692–70, 2011.
8. Katsumi Inoue. Linear resolution for consequence finding. *Artificial Intelligence*, 56(23):301 – 353, 1992.

9. Katsumi Inoue, Andrei Doncescu, and Hidetomo Nabeshima. Hypothesizing about causal networks with positive and negative effects by meta-level abduction. In *Proceedings of the 20th international conference on Inductive logic programming*, ILP'10, pages 114–129, Berlin, Heidelberg, 2011. Springer-Verlag.
10. Katsumi Inoue, Andrei Doncescu, and Hidetomo Nabeshima. Completing causal networks by meta-level abduction. *Machine Learning*, 91(2):239–277, 2013.
11. Kurt W Kohn and Yves Pommier. Molecular interaction map of the p53 and mdm2 logic elements, which control the off-on swith of p53 response to dna damage. *Biochem Biophys Res Commun*, 331(3):816–27, 2005.
12. Stephen Muggleton and Christopher H. Bryant. Theory completion using inverse entailment. In *Proceedings of the 10th International Conference on Inductive Logic Programming*, ILP '00, pages 130–146, London, UK, UK, 2000. Springer-Verlag.
13. Hidetomo Nabeshima, Koji Iwanuma, Katsumi Inoue, and Oliver Ray. Solar: An automated deduction system for consequence finding. *AI Commun.*, 23(2-3):183–203, April 2010.
14. Y. Pommier, O. Sordet, V.A. Rao, H. Zhang, and K.W. Kohn. Targeting chk2 kinase: molecular interaction maps and therapeutic rationale. *Curr Pharm Des*, 11(22):2855–72, 2005.
15. Oliver Ray. Automated abduction in scientific discovery. In *Model-Based Reasoning in Science and Medicine*, pages 103–116. Springer, June 2007.
16. Oliver Ray, Ken Whelan, and Ross King. 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, pages 661–666, Washington, DC, USA, 2010. IEEE Computer Society.
17. Philip Reiser, Ross King, Douglas Kell, Stephen Muggleton, Christopher Bryant, and Stephen Oliver. Developing a logical model of yeast metabolism. *Electronic Transactions in Artificial Intelligence*, 5:233–244, 2001.
18. Raymond Reiter. Readings in nonmonotonic reasoning. chapter On closed world data bases, pages 300–310. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1987.
19. Joseph Shoenfield. *Mathematical logic*. Addison-Wesley series in logic. Addison-Wesley Pub. Co., 1967.
20. Pierre Siegel. *Representation et utilisation de la connaissance en calcul propositionnel*. Thèse d'État, Université d'Aix-Marseille II, Luminy, France, 1987.
21. Alireza Tamaddoni-Nezhad, Antonis C. Kakas, Stephen Muggleton, and Florencio Pazos. Modelling inhibition in metabolic pathways through abduction and induction. In Rui Camacho, Ross D. King, and Ashwin Srinivasan, editors, *Inductive Logic Programming, 14th International Conference, ILP 2004, Porto, Portugal, September 6-8, 2004, Proceedings*, volume 3194 of *Lecture Notes in Computer Science*, pages 305–322. Springer, 2004.
22. Jeffrey Ullman. *Principles of database systems*. Computer software engineering series. Computer Science Press, 1980.