Representation and efficient algorithms for the study of cell signaling pathways

A. Doncescu[1], **P. Siegel**[2], **T. Le**[1]
[1]LAAS-CNRS, University of Toulouse, Toulouse, France
[2]Aix Marseille Université, CNRS, LIF UMR 7279, 13288, Marseille, France

Abstract— *Artificial Intelligence and constraint programing find in the formal description of cell's interactions a field of inquiry well suited. Particularly, the questions concerning the representation of incomplete biological knowledge and explanation to some biological effects could be well formulated in A.I. frame. Basically, A.I. plays with small examples. In systems biology the algorithmic complexity is too important to be handled using traditional approaches. This article tries to answer to the question "What is the simplest representation for signalling pathways, reflecting reasonable complexity and tractable algorithms?"*

Keywords: Cell signalling pathways, knowledge representation, non-monotonic logics, default logic, abduction, tractable algorithms.

1. Introduction

For biologists, it is possible to represent a cell and its evolution by a graph that describes the interactions between proteins/genes. This representation can includes mathematical properties as connectivity; presence of positive and negative loops which is related to a main property of genetic regulatory networks. Biochemical reactions are very often a series of time steps instead of one elementary action. Therefore, one direction research in system biology is to capture or to describe the series of steps called pathways by metabolic engineering. All reactions that allow the transformation of one initial molecule to a final one constitute metabolic pathways. Each compound that participates in different metabolic pathways is grouped under the term metabolite.

The study of gene networks poses problems well identified and studied in Artificial Intelligence over the last thirty years. Indeed, the description of network is not complete: biological experiments provide a number of protein interactions but certainly not all of them. On the other hand the conditions and sometimes the difficulties of the experiments involves these data are not always accurate. Some data may be very wrong and must be corrected or revised in the future. Finally the information coming from different sources and experiences can be contradictory. It is the goal of different logics, and particularly non-monotonic logics, to handle this kinds of situation. Afterwards this interaction maps should be validated by biological experiments. Of course, these experiments are time consuming and expensive, but less than an exhaustive experiment.

We focus on four main problems: handling the conflicts which can occur in the gene representation, completing in-silico the gene network, knowledge discovery on these networks and the practical handling complexity of algorithms.

In this paper we present algorithms and formalism in the simplest manner to solve and process problems related to gene networks.

Our approach is based on default logic allowing to handle the incomplete information, and on abductive reasoning to complete the missing information from the gene network. But, even limited to propositional calculus, the theoretical complexity for default logic and abduction revolves around $\sum_2^p$ which is totally unacceptable for real problems. The last part is dedicated to a new language of representation, which seems to be the key to algorithm complexity handling.

2. Signaling Pathway

Nowadays, bioinformatics represents the key field to explain the functionality of life science. To analyze a biological system it is necessary to find out new mathematical models allowing to explain the evolution of the system in a dynamic context or to deal in a simple manner with the complex situations where the human experience overtakes mathematical reasoning. The study of systems biology trough First Order Logic aims at improving the understanding metabolite-proteins interactions by finding out new possible interactions which add consistency to the knowledge. The p53/Mdm2 complex represents the best-studied relationship between a tumor suppressor gene which functions as a transcription factor and an oncogene. Figure 1 given and studied by [1] and [14] is an elementary pathway of interactions around p53 and mdm2. This example explains in a very simple manner the interaction cancer-p53/Mdm2, represented on the form of causality graph, using Default Logic and Abduction.

Through different mechanisms, the ultraviolet UV drives the cell to cancer. This is represented by an arrow : $UV \rightarrow cancer$. On the other hand the UV activate the protein $p53$, $(UV \rightarrow P53)$. This protein activates a protein A $(P53 \rightarrow A)$) and A blocks cancer ($A \dashv cancer$). But, in some conditions, $Mdm2$ binds protein $p53$ and the obtained complex, activates B and B blocks A. We also have C blocks B. We will uses X thereafter.

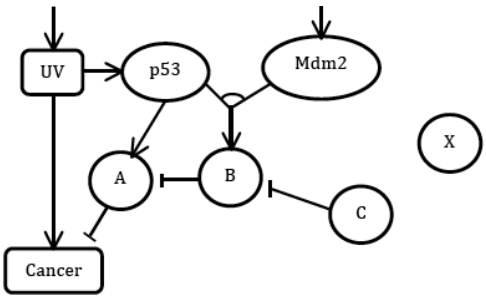


Fig. 1: The Simplified Model of Double Strand Break.

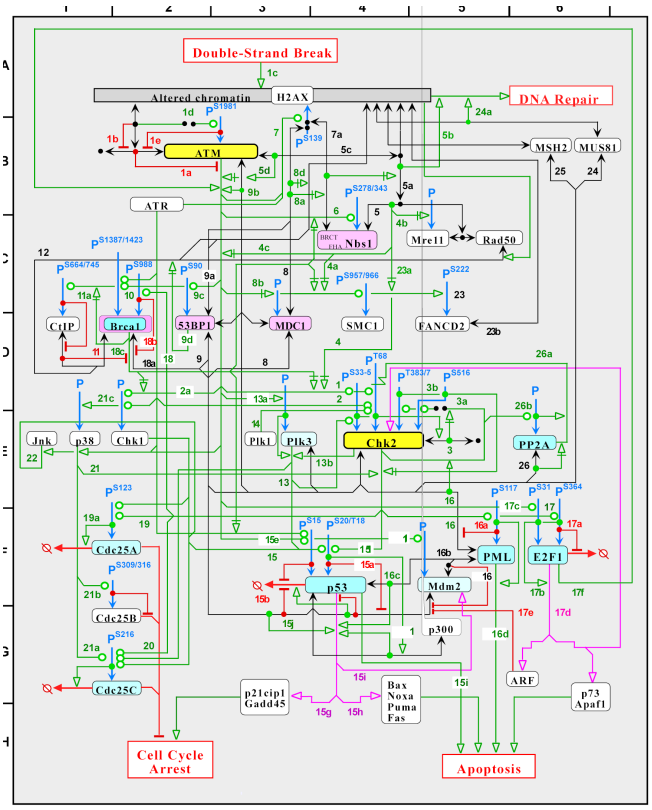


Fig. 2: DNA Double Strand Break Map.

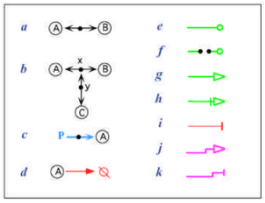
2.1 DNA Double Strand Break Map

A pathway may include thousands of genes, which will pose problems of complexity. We have worked from years on the DNA double strand break map introduced by Pommier and all [16] (Fig. 2). This map is very complicated and it is very difficult to complete it due to the great number of experiments necessary. Therefore, the only issue is to find out the main pathway when a gene is knockout.

3. Logic representation

3.1 Language representation

In our model genes and proteins are viewed as the same object (the genes produce proteins). To describe interactions between proteins it is possible to use a language of classical logic (propositional or first order logic). We can say, for example $stimulation(UV)$ to say that the cell is subjected to ultraviolet or $GlassScreen \rightarrow \neg stimulation(UV)$ to say that a glass screen protects against ultraviolet.



Symbols :
(a) Proteins A and B could link, the knot represents the binding A:B ;
(b) Multimolecular Complex binding ;
(c) Covalent Modification of A ;
(d) Degradation of A ;
(e) Enzymatic stimulation in transcription ;
(f) Autophosphorylation
(g) General Stimulation;
(h) Necessity ;
(i) Inhibition ;
(j) Activation of transcription ;
(k) Inhibition of transcription

In the context of cell pathways, a predicate can be an action on one or more protein. For example :

$product(P), trigger(P,R), binding(P,Q,R), block\text{-}bindind(P,Q)$,

$stimulation(P), phosphorylation(P)$

$dissociation, transcription\text{-}activating..$

The predicate can also represent properties on protein concentration :

$concentration(P, > 1000)$,

$concentrationIncrase(P)$

$concentrationDiscrase(P)..$

A formula is :

$product(p\text{-}s15\text{-}p53\text{-}mdm2) \wedge product(p\text{-}chk1)$
$\rightarrow phosphorylation(p\text{-}chk1, p\text{-}s15\text{-}p53\text{-}mdm2)$

and T represents time written as :

$stimulate(dsb, dna, T) \rightarrow product(altered\text{-}dna, T+1)$

Using a simple logic formalism can express much of what biologists are needed to represent. We are in a logical framework, so it is possible to represent a lot of information in a very natural way. The compromise in the case of using the entire first order language, is incompleteness and the combinatorial explosion of complexity algorithms. It is therefore essential to reduce the expressivity power of the language by often using of a propositional representation, if it is possible from biological view point.

Accurate quantification of protein concentration is very important in the case of determining a binding constant or measure enzyme kinetics. But even if something more qualitative is considered, having a good idea of how much protein you have will enable you to compare results from one experiment to the next and from one protein to the next. Therefore, the protein concentration is rarely precise and to represent a change in concentration predicates such as $increased$ or $decrease$ are introduced in our model.

3.2 From conflict to inconsistency and causality

The Representation of Signaling Pathways uses classical first-order logic formulas. This representation is not adequate due to the conflict between the different rules

and therefore inconsistency. The principal problem comes from simultaneously activations and blocking. For example, the Signaling Pathway represented in the Figure 1 we can represent that the UV gives cancer ($UV \rightarrow cancer$) by the clause $trigger(UV, cancer)$; but in the same time we can consider that UV blocks cancer ($UV \dashv cancer$) using the predicate $block(UV, cancer)$ [1] [6]. In this case A and UV active cancer simultaneously which is not possible and therefore the inconsistency of the model.

In fact, this is a very simple form of causality. A lot of research has been done to represent causality. It is possible to use symbolic or numeric formalisms. Even if Bayesian approaches or probabilistic logics [19] seem to be very attractive approach, it is impossible here to go around all these works. We will simply try to describe and use a form of causality (the simplest possible) sufficient for the application to Biological Systems.

The inferences of classical logic $A \rightarrow B$ or $A \vdash B$ are fully described formally, with all the "good" logic properties (tautology, not contradiction, transitivity, contraposition..). But the causality cannot be seen as a classical logic relation.

In a first approach, the first properties that we want to give can be expressed naturally:

(1) If A triggers B and A is true, then B is true.
(2) If A blocks B and A is true, then B is false.

Depending on the context, true can mean the known, certain, believed, proved... The first idea is to express these laws in classical logic by axioms:

$trigger(A, B) \wedge A \rightarrow B$
$block(A, B) \wedge A \rightarrow \neg B$

They can also be weakly expressed by inference rules:

$trigger(A, B), A \vdash B$
$block(A, B), A \vdash \neg B$

But these two formulations are problematic when there is conflict. If for example we have a set of four formulas $F = \{A, B, trigger(A, C), blocks(B, C)\}$, we will in the two approaches above infer from F, B and $\neg B$. This is inconsistent. To solve such conflicts, we can try to use methods inspired by constraint programming, such as the use of negation by failure in Prolog or Solar. It is also possible to use a non-monotonic logic.

The first method, negation by failure, poses many theoretical and technical problems if you go further as the simple cases. These problems are often solved by adding properties to the formal system, properties that pose other problems. Therefore, we will use a classical non-monotonic formalism, the default logic [18].

4. Default logic and Causality

To resolve the conflicts seen above, the intuitive idea is to weaken the formulation of rules :

(1) If A triggers B, if A is true and it is possible/non-contradictory that B, then B is true.

(2) If A blocks B, if A is true and it is possible/non-contradictory that B is false then B is false.

The question which we want to answer is *What is formally "possible" and "not contradictory"* ?

This question began to arise in artificial intelligence forty years ago. In this type of reasoning, one has to reason with incomplete information, uncertain and subject to revision and sometimes false. On the other hand we must often choose between several possible conclusions contradictory. The non-monotonic logics have been studied to solve this type of problems.

4.1 Non-monotonic logics

The "classical" logics, as first order logic or modals logics are monotonic : if it adds information or a formula E' to a formula E, everything which was deduced from E will be deducted from $E \cup E'$. In other words, anything which is deduced from informations will always be true if we had new information. This monotonicity property will generate incompleteness or revisable informations. Indeed, in this case it will be common to invalidate previously established conclusions when new information is added or changed.

A non-monotonic logic allows to eliminate the monotony property of the classical logic: the conclusions could be revised with the addition of new knowledge.

4.2 Default logic

Of course, the problem of non-monotonicity generates many logics, but the most used is Default Logic [18]. Default logic could be seen as an improvement and a generalization of the negation by failure in $Prolog$. It is also a generalization of ASP formalisms which appeared later [13]. Default logic formalizes the default reasoning: conclusions can be made, in the absence of opposite proof.

A default logic is defined by $\Delta = (D, W), W$ is a set of facts (formulae from the propositional logic or the first order logic) and with D, a set of defaults (inference rules with specific content, which handle the uncertainty). A default is an expression of the form:

$$d = \frac{A(X) : B(X)}{C(X)}, \tag{1}$$

where $A(X), B(X)$ and $C(X)$ are formulae and X is a set of variables. $A(X)$ is the prerequisite, $B(X)$ is the justification and $C(X)$ is the consequent. Intuitively, the default d means: if $A(X)$ is true, if it is possible that $B(X)$ is true, then $C(X)$ is true. If $B(X) = C(X)$, the default is normal. In this paper a default is semi-normal its justifications entail its conclusion. The normal default means "Normally, the A are B". In this paper a default is semi-normal if its justification entail its conclusion.

The goal of default logic is to find extensions of a default theory $\Delta = \{W, D\}$. Simplifying, an extension E is a consistent set of formulas obtained by adding, under

condition, to W a maximal set of consequents of D. For us, an extension can for example, represent a subgraph without conflict, of the gene network. Formaly E is an extension of Δ if and only if $E = \cup_{i=0}^{\infty} E_{i+1}$ with:

1) $E_0 = W$

2) $E_{i+1} = Th(E_i) \cup \{C / (\frac{A : B}{C}) \in D, A \in E_i, \neg B \notin E\}$

where $Th(E_i)$ is the set of theorem obtained in a monotonic way from E_i.

It is important to notice that E appears in the definition of E_{i+1}. We need to verify that $\neg B \notin E$. So, we need to know E to find E_i. In a general case an extension is a fixed point which give a lot of problems. But, if we work with normal or seminormal defaults, the definition of an extension is changed: we need only to verify that $\neg B \notin E_i$ and the algorithms are much simpler and effective. For our case study, we just use normal or seminormal defaults.

4.3 Default logic for Signaling Pathways

4.3.1 Representation using Default Logic

We showed that the problem of activation and inhibition is traduced in logic as:

(1) If A triggers B, if A is true and it is possible that B, then B is true.

(2) If A blocks B, if A is true and it is possible that B is false then B is false.

In default logic, these rules can be represented by a set $D = \{d1, d2\}$ of two normal defaults :

$$d1 = \frac{trigger(A, B) \wedge A : B}{B}$$

$$d1 = \frac{trigger(A, B) \wedge A : \neg B}{\neg B}$$

$trigger(A, B)$ et A are both $TRUE$, so it is written as a set $W = \{trigger(A, B), A\}$ containing two classical logic formula. Therefore, the information is represented by the default theory $\Delta = \{W, D\}$

4.3.2 Extensions and choice of extension

If the extensions Δ are calculated for the previous biological situation, 2 extensions are obtained. Each extension is represented by a subgraph without $conflict$. The extensions $E1$ and $E2$ are:

$E1 = \{trigger(A, B), A, B\}$ if $d1$ is used.
$E2 = \{trigger(A, B), A, \neg B\}$ if $d2$ is used.

$E1$ contains B and $E2$ contains $\neg B$. From biological interpretation $E1$ mean B is active and $E2$ is blocked.

The conflict is solved, but it is not possible to rank the extensions: B is active or blocked? In fact this will really depend on the context. For biologists, some times the positive interactions are preferred to negatives. Another possibility is to use probabilistic or statistical methods or to weight each extension based on the evaluation of the knowledge. From an algorithmic viewpoint the ranking of extension could also be evaluated during the calculation of the extensions and even the off-line ranking could be preferred.

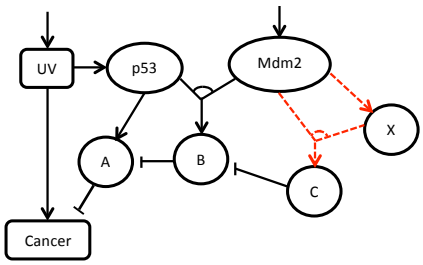


Fig. 3: mdm2 binds X.

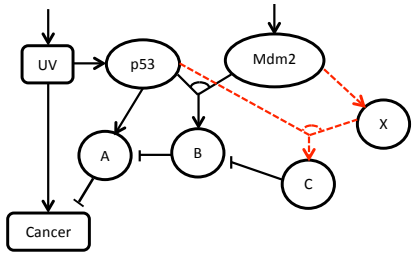


Fig. 4: p53 binds X.

5. Completing the Signaling Pathways by Default Abduction

In section 2 figure 1, we have introduced a simple example which sums up the question *"How to block cancer by preventing B?"*. One analytical solution is to activate a protein C, which blocks B and to consider that a protein X is candidate for binding with $p53$ or $mdm2$. This simple reasoning reflects the biologists approach. Therefore, in this case it is possible to set-up some experiments which prove this hypothesis. It is well known that experiments are time consuming and expensive. It is obviously that in the case of big data it is necessary to automize the process taking into account all available information. In this case we can consider how it is possible to find out, in-silico, a molecule (a future drug) which has the chance to act effectively by blocking some pathways. In terms of A.I. the drug discovery is a problem of abduction.

Classical logic uses the deduction $F \vdash R$ that says that a result R is inferred from an information (formula) F. Abduction generalizes deduction. The information F is incomplete and abduction amounts to adding to F a set of hypotheses H such that $F \wedge H \vdash R$ and $F \wedge H$ is consistent.

Abduction is very important in AI. The trouble comes with implementation of the algorithms. Abduction algorithms are far too high computational complexity. Even limited to

propositional calculus, the theoretical complexity revolves around $\sum_2^p$ which is totally unacceptable when we go beyond small the examples. Many theoretical studies have been done on the complexity of the abduction and research sub-language of propositional calculus where complexity is reduced. These sub-languages most often cover the Horn clauses and renaming. But even here the complexity is too great for even, more or less, NP-complete. Conversely, existing polynomial classes provide only a low power of expression on issues to be addressed. On the other hand, for many real applications, experience shows that it is not necessary to use the full expressive power of logic.

In the case of Signaling Pathways, abduction is used mainly to search missing interactions. The predicate $trigger(\alpha, X)$ is used inside the default with variables $\frac{:trigger(\alpha,X)}{trigger(\alpha,X}$. Here α is a variables which could be instansiated by any protein. This can be generalized to all predicates. By calculating the extensions which contains the predicate $block(cancer)$ we obtained 8 extensions but only 2 contains $p53$ and $mdm2$ (Figures 3 and 4)

6. Logic Representation of Pathway to Reduce Computational Complexity

We present the outline of a language dedicated to discovery of biological interactions answering these requests. This formalism uses the default logic and also has a dynamic approach by considering time as a succession of events. The syntax is described in the next section.

6.1 Clauses

Here $product(P53)$ means that the protein p53 increases in concentration and $\neg product(P53)$ means that it is not possible to determine if the p53 concentration is increasing. The dynamic of the system can be, specified by $concentration(p53, 100, T)$ which means the concentration of $p53$ at the time T is equal to 100 a.u. And $\neg concentration(P53, sup(200), T+2)$ says that at the time $T + 2$, the concentration of $p53$ is not greater than 200 a.u.

The simplest formulas are the clauses. Formally, a clause is a disjunction (or a list) of literals $l_1 \vee .. \vee l_n$. A Horn clause is a clause with a maximum of one positive literal. The clauses a and $\neg b \vee \neg c \vee d$ and $\neg b \vee \neg c$ are Horn clauses. And $a \vee b$ is not one. For the rest we use Horn clauses which are interesting for two reasons.

First using Horn clauses is a natural way to represent knowledge. The formula $a \wedge b \wedge c \wedge d \rightarrow d$ is equivalent to the Horn clause $\neg a \vee \neg b \vee \neg c \vee d$. In the same time the formula $\neg(a \wedge b)$ (a and be cannot be True in the same time) is equivalent to the negative Horn clause $\neg a \vee \neg b$.

The second advantage of Horn clauses, fundamental here, is that their use drastically reduces computational complexity. Indeed, any logical formula can be rewritten as a set of clauses, so complexity problems may arise in terms of clauses. For propositional calculus the fundamental problem is whether a set of clauses is consistent or not. This is the problem SAT which is NP-complete. Otherwise all known algorithms are exponential in the worst case. On the other hand, if all clauses are Horn causes, algorithms can be linear proportional to the size of the data. For genes pathways, the use of Horn clauses provides practically usable algorithms .

Obviously Horn clauses can not represent all formulas. In particular $a \vee b$ is not a Horn clause. But in practice, this type of positive disjunctive information is quite rare. We have not really found it for the gene networks that we studied. If there are, most of the time you can use renaming techniques to solve the problem. Finally, if nothing works and it is impossible to use only Horn clauses, there are techniques to limit the combinatorial explosion. For example use strong backdoors, managing mutual exclusion and cardinality, recognition of symmetries. Here we are in the topic of practice solving NP-complete problems.

A logician may be offend by the foregoing. But for our experiments on cell signaling pathways, it was not necessary to leave the framework of Horn clauses. In fact the use of Horn clauses and default logic had been studied for dynamical application [20]. In that case, default logic is used to simulate decisions of a commander on a submarine in wartime. This is a problem of incomplete information and the decisions have to be done in real-time. Using Horn clauses, defaults logic, mutual exclusion and simple management of the temporal aspect, it was possible to simulate several hours of fighting in a few seconds. In that simulation program the abduction was not used but it is one of the first proof of default logic Horn clauses used to solve real problems.

6.2 Language syntax

A rule is a triplet $(< type >, < corps >, < weight >)$.

• <type> can take 2 values : $hard$ or def. If the value is $hard$ the rule is an hard-rule and represents an Horn clause, which is sure and non-revisable. If the value is def the rule represents a normal default.

• <weight> weights the rule. These weights will make it possible to choose between the different extensions proposed by the algorithm.

• <corps> is a couple (L, R). The left element L is a set of literals $(l_1, ..l_n)$ perhaps empty. This set is identified to $l_1 \wedge .. \wedge l_n$. The right element R is either a single literal or empty. If the rule is hard, the couple (L, R) represents the formula $L \rightarrow R$. If the rule is a default, the couple represents a normal default $\frac{L:R}{R}$ An increased attention is done to these two cases.

Hard Rules

A hard rule (L, R) represents the formula $L \rightarrow R$ where L is a conjunction of literals and R a literal. How we decided to restrict our algorithm to Horn clauses all literals of L are positive. The literal R can be positive or negative. Here we have two special cases. :

1) L is empty. Therefore the rule represents a positive or negative unary clause. The unary clauses are elementary sources of information. They did not contain variables, they are ground clauses. This allows the decidability of the algorithm. However the other clauses can contain variables, leave the pure propositional calculus.

2) R is empty. For this empty-consequence, the rule $L \rightarrow \emptyset$ is equivalent to $\neg L$ equivalent to a negative clause. For example, we can use such a clause to represent a mutual exclusion "It is impossible to trigger and to block a protein at the same time".

Default Rules

If the rule (L, R) is a default, then it represents a normal default, the prerequisite is L, and R is the justification and also in the same time the consequent. If the prerequisite is empty, the default is without justification. By definition of the defaults it is impossible to have an empty consequence. Contrary of the hard rules the prerequiste R can contains negative literals.

6.3 Cell Signaling Pathway Representation

We have worked on the bibliographic data of the response to DSBs translated on a map of molecular interactions Figure 2 [16]. A draw back of this map is that it is very difficult to add a new interaction or protein without full reassessment. In particular the management of conflicts is very difficult. So we worked on the translation of this map into our language. Initial results have translated this map and tested some algorithms [10], [11].

Today, the map is translated by 206 rules in a very natural way, without having to "tweak" the predicates or the rules. The rules are expressed in the syntax above. These rules can be hard rules or defaults. With our syntax it is very simple to change the nature of the rules to test different configurations. We can calculate the extensions in a very short time. We never needed to use non Horn clauses. This reinforces our opinion that it is possible to use a nonmonotonic logic and also abduction and also time, on real applications.

6.4 Rule Examples

In the context of cell pathways, a predicate can be an action on one or more protein. In section 3.1 we saw examples of predicates.

We give here some examples of rules written for our example :

$hard : stimulate(dsb, dna)$

that is an elementary fact (a ground unary clause) who says that dsb stimulates ADN.

$def : stimulate(dsb, dna) \rightarrow product(altered\text{-}dna)$

that is default rule "Generally when dsb stimulates DNA, altered DNA is produced.

$hard : product(p\text{-}atm\text{-}atm\text{-}bound)$
$\rightarrow \neg product(atm\text{-}atm)$

that is a negative clause.

Using a simple logic formalism can express much of what biologists are needed to represent.

7. Implantation of Algorithms

The algorithm is written in SWI Prolog. A rule :

$(< type >, < corps >, < weight >)$

is represented by a unary Prolog clause:

$rule(< type >, < corps >, < weight >).$

Therefore, the rules and the algorithm are in the same Prolog program, which is very practical. Another advantage to use Prolog is that the unification, the backtracking and the lists are well optimized. Of course Prolog is interpreted, so it is slower than compiled languages (but not that much). In the other hand Prolog programs are short and simple, which saves a lot of time to test programs and heuristics.

This algorithm calculates the extensions. As the clauses are Horn clauses and as the defaults are normal, the research tree is optimized. Particularly it is easy to calculate extensions without duplication (we do not calculate several times the same extension). For algorithms, we can also use a weak form of negation as failure.

For initial tests, given by the map of the entire network of Pommier [16], we can calculate all extensions in a short time. For example with most of the rules by default, there are two extensions. The calculation takes 500000 LIPS and 0.4 seconds of CPU time on MacBook. The temporal aspect of gene networks has been tested for small examples, but the scaling has not yet been done. For the abduction, it is almost the same.

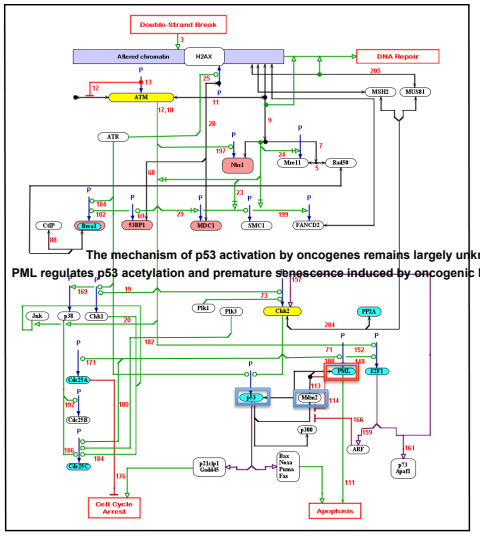


Fig. 5: DNA double strand break generated automatically by using the most relevant extension.

8. Results

Basically, many researchers are trying to complete the Signaling Map. In our approach the map is simplified which is very useful for biological experiments. Introducing time in defaults (the prerequisite considered at time t and the conclusion at time $t+1$), we obtained a simplified map of Pommier. The most interesting result is the identification of the molecule "X" from figure 1 as be PML which regulates p53 acetylation and premature senescence induced by oncogenic Ras.

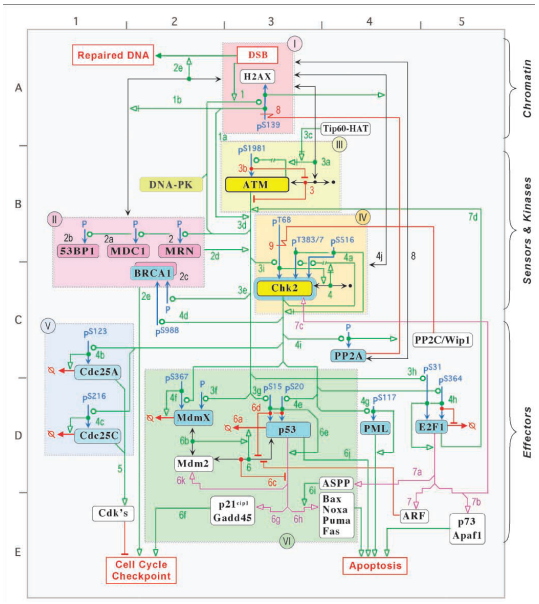


Fig. 6: The Molecular Interaction Map and Rationale for Chk2 build-up "manually" by Pommier [17] using biological cause-effect graph.

At first we tested the notion of production fields and a consequence finding algorithm for producing clauses [4], [15]. We tested also the SOLAR language that uses production field and this algorithm. The results are mixed in the case of "big" examples. We also looked at the ASP formalism [13]. Again the impression is mixed. Indeed ASP deal mainly with normal defaults without prerequisites. Getting all the power representation of defaults with prerequisite is possible by rewriting techniques. But you lose a a lot of clarity and also efficiency.

By generating automatically the DNA double strand breaks map (Figure 5) we noticed that the protein p73 is not directly involved in activation of apoptosis as in the case of Pommier map (Figure 6). This result obtained from incomplete knowledge constitutes a theory formation framework for "Knowledge Discovery" using Default Logic.

9. Conclusion.

The A.I. challenge is to explain new phenomena using automatic causal discovery. In this paper we introduced a formalism able to infer signaling pathway by using defaults approach and abductive reasoning.Therefore the originality of the work is given by the capacity of the proposed logical model to find out the consistency of Pommier's map by efficient algorithms.

References

[1] N. Tran, C. Baral, *Hypothesizing and reasoning about signaling networks.* Journal of Applied Logic, 7, 253-274,2007.

[2] CH. Bassing, FW Alt. *H2AX may function as an anchor to hold broken chromosomal DNA ends in close proximity.* Cell Cycle 2004; 3:149-53.

[3] J. Bartkova, Z. Horejsi,et al., *DNA damage response as a candidature anticancer barrier in early human tumorigenesis.* Nature 2005; 434:864-70.

[4] J.M. Boi, E. Innocenti, A. Rauzy, P. Siegel, *Production Fields : A New approach to Deduction Problems and two Algorithms for Propositional Calculus.* Revue d'Intelligence Artificielle, 25(3) : 235-255, 1992.

[5] G. Bossu, P. Siegel. *Saturation, Nonmonotonic Reasoning and the Closed World Assumption.* Artificial Intelligence, 25(1) :13-63, 1985.

[6] A. Doncescu, Y. Yamamoto, K. Inoue, *Biological systems analysis using Inductive Logic Programming.* Proc. of the 21st International Conference on Advanced Information Networking and Applications (AINA 2007), pages 690-695, IEEE Computer Society, 2007.

[7] A. Doncescu, K. Inoue, Y. Yamamoto, *Knowledge-based discovery in systems biology using CF-induction. New Trends in Applied Artificial Intelligence.* Proc. 20th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA / AIE 2007), Lecture Notes in Artificial Intelligence, volume 4570, pages 395-404, Springer, 2007.

[8] A. Doncescu, J. Weisman, G. Richard, G. Roux, *Characterization of bio-chemical signals by inductive programming.* Knowledge Based Systems, 15 (1), 129-137, 2002.

[9] A. Doncescu, T. Le, P. Siegel, *Default Logic for Diagnostic of Discrete Time Systems.* Proc. BWCCA-2013 - 8th International Conference on Broadband and Wireless Computing, Communication and Applications p. 488-493, Compiegne, France, Oct 2012

[10] A. Doncescu, P. Siegel, *The Logic of Hypothesis Generation in Kinetic Modeling of System Biology,* Proc. 23rd IEEE International Conference on Tools with Artificial Intelligence, p. 927-929, Boca Raton, Florida, USA, Nov. 2012

[11] A. Doncescu, T. Le, P. Siegel, *Utilization of Default Logic for Analyzing a Metabolic System in Discrete Time.* Proc.13th International Conference on Computational Science and Its Applications, ICCSA 2013, p. 130-136, Ho Chi Min, Vietnam, June 2013.

[12] D. Kayser, F. Levy, *Modeling symbolic causal reasoning,* Intellecta 2004, 1, 38, pp 291-232, 2004

[13] E. Giunchiglia, J. Lee, V. Lifschitz, N. McCain, H. Turner, *Nonmonotonic causal theories* Artificial Intelligence, No. 1-2 vol.153 pp.49-104, 2004.

[14] K. Inoue,A. Doncescu, H. Nabeshima. *Completing causal networks by meta-level abduction.* Machine Learning, 91 (2) :239-277, 2013.

[15] H. Nabeshima, K. Iwanuma, K. Inoue, O. Ray. *SOLAR: An automated deduction system for Finding consequence.* AI Commun, 23 (2-3): 183-203 (2010)

[16] Pommier Y. and all. *Targeting Chk2 Kinase : Molecular Interaction Map and Therapetic Rationale.* Current pharmacy design, 11(22):2855-72, 2005.

[17] Pommier Y. and all. *Chk2 Molecular Interaction Map and Rationale for Chk2 Inhibitors* Clin Cancer Res. 2006 May 1;12(9):2657-61..

[18] R. Reiter *A Logic for Default Reasoning.* Art. Int. 13(1-2): 81-132 (1980).

[19] T Sato, Y Kameya. *PRISM: a language for symbolic-statistical modeling.* International Joint Conference on Artificial Intelligence 15, 1330-1339.

[20] I. Toulgouat, P. Siegel, Y. Lacroix, J. Botto *Operator decision modeling : application to a scenario involving two submarines* Proc. COMPIT10, Gubbio, April 2010 *and* Proc. 13th NMR10, Toronto, Canada May 2010