Hypothesizing about Signaling Networks

Nam Tran [a] & Chitta Baral [b]

[a] *Department of Pathology, Yale School of Medicine, Yale University, New Haven, CT 06510, USA*

[b] *Computer Science and Engineering, School of Computing & Informatics, Arizona State University, Tempe, AZ 85281, USA*

Abstract

The current knowledge about signaling networks is largely incomplete. Thus biologists constantly need to revise or extend existing knowledge. The revision and/or extension are first formulated as theoretical hypotheses, then verified experimentally. Many computer-aided systems have been developed to assist biologists in undertaking this challenge. The majority of the systems help in finding "pattern" in data and leave the reasoning to biologists. A few systems have tried to automate the reasoning process of hypothesis formation. These systems generate hypotheses from a knowledge base and given observations. A main drawback of these knowledge-based systems is the knowledge representation formalism they use. These formalisms are mostly monotonic and are now known to be not quite suitable for knowledge representation, especially in dealing with the inherently incomplete knowledge about signaling networks. We proposed an action language based framework for hypothesis formation for signaling networks. We showed that the hypothesis formation problem can be translated into abduction problem. This translation facilitated the complexity analysis and an efficient implementation of our system. We illustrated the applicability of our system with an example of hypothesis formation in the signaling network of the p53 protein.

Key words: signaling networks, action languages, hypothesis formation, abduction
PACS:

Email addresses: nam.tran@yale.edu (Nam Tran), chitta@asu.edu (Chitta Baral).

1 Introduction

The living cell constantly receives and responds to signals from its environment. A signal is initiated when some extracellular molecules are sensed by their respective cell-surface receptor. Signaling molecules inside the cell then interact with one another to *transduce* the signal into cellular responses that regulate the introduction of different proteins, thus controlling various functions of the cell. Specific collections of interactions with a common theme in a network are often referred to as *signaling pathways* or *signaling networks*. Almost any disease can be described in terms of aberrations in signaling networks; for example cancer is caused by a breakdown in networks regulating cell growth. Modeling signaling networks is thus essential for understanding the cell function and can lead to effective therapeutic strategies that correct or alter abnormal cell behaviors.

Signal transduction and signaling networks have become a major research focus. The knowledge about signaling mechanisms is growing exponentially. It is impossible for a single biologist (or even a small group of biologists) to handle the large body of interactions in and the resulting complexity of signal transduction networks. This calls for knowledge representation and automated reasoning capability.

In recent years there have been intensive efforts in the modeling and reconstruction of signaling networks of the cell. Most of the modeling approaches are concerned with prediction of the cell behavior, using simulation of quantitative Voit (2000); Regev et al. (2001); Peleg et al. (2002); Davidson et al. (2002b); de Jong (2002); Hoffmann et al. (2002); Schoeberl et al. (2002); Priami (2003) or semi-quantitative models Meyers & Friedland (1984); Karp (1993b); Shimada et al. (1995); Heidtke & Schulze-Kremer (1998); Matsuno et al. (2000); Alur et al. (2001); Kam et al. (2001); Batt et al. (2003); de Jong et al. (2003). The major limitation of these approaches is that parameters and data for these quantitative models are hard to obtain. A minority of the approaches is geared toward qualitative modeling, i.e., representing and reasoning with knowledge of signaling networks Karp et al. (2000); Ogata et al. (1999); Karp et al. (2002a); Krieger et al. (2004); Eker et al. (2002); Talcott et al. (2004); Maimon & Browning (2000); Chiaverini & Danos (2003); Danos & Laneve (2003); Fukuda & Takagi (2001); Fukuda et al. (2003); Demir et al. (2004); Cook et al. (2001); Kohn (1999); Sambrano (2003); Kitano (2003); Funahashi et al. (2003). A main drawback of these qualitative approaches is the knowledge representation formalism they use. These formalisms are mostly monotonic and are now known to be not quite suitable for knowledge representation.

1.1 Knowledge Base of Signaling Networks

We propose a knowledge-based approach to modeling signaling networks. We represent cellular signaling networks in a knowledge base. We then reason and hypothesize about the networks by asking different kinds of queries formulated in a formal language. The knowledge base is augmented with various reasoning mechanisms that allow the answering of the queries. An important dimension of our approach is that it allows for reasoning mechanisms that gracefully handle incomplete or partial information. This is extremely important as existing regulatory networks often contain missing or suspected interaction links, or proven interactions whose outputs are uncertain (e.g., the yeast 2-hybrid interactions mentioned by Sambrano (2003)). Besides being able to handle such incomplete information, our approach also allows for easy updating of the knowledge base when new knowledge becomes available. This avoids significant overhauling of the old model or scrapping of the old model and making a new model from the scratch. This is important because we constantly need to revise or update our knowledge about signaling networks due to its inherent incompleteness.

A typical example of the kind of behavior we would like to model is the happening that follows when a particular ligand binds to a receptor in the membrane of a cell. The immediate effect is that the ligand binds with the receptor. Moreover, such a binding in the presence of certain other molecules inside the cell may *trigger* an action (or another binding) which in turn may trigger other actions. Sometimes the presence of particular molecules can *inhibit* certain actions that would have been otherwise triggered.

While modeling behaviors of the cell, we are interested in formalization and implementation of several reasoning abilities that include (i) *predicting* the impact of a particular action, (ii) *explaining* observations, (iii) *planning* to make certain components of the cell behave in a particular way. These kinds of reasoning have ultimate significance to cell biology and medical science. For example, a drug can have side effects such as preventing a particular hormone from being produced thus disrupting certain cellular and regulatory mechanisms. Reasoning about side effects of a drug corresponds to prediction. Another example is that one may observe abnormal cellular behaviors such as persistent proliferation in place of programmed cell death. Then one would want to find out the cause of such an abnormality. Such reasoning corresponds to explanation or medical *diagnosis*. Finally, one may want to figure out a way to correct abnormal behaviors of the cell. For example, one can introduce particular drug elements to the cell or cell membrane at particular time instances. Planing for these kinds of intervention corresponds to drug design and drug therapy.

1.2 Hypothesizing about Signaling Networks

Because of the complexity of living systems and the limitation of scientific methods available for the study of those systems, biological knowledge is inherently incomplete. The incompleteness of knowledge constantly manifests itself in unexplainable observations. To account for these novel observations, biologists need to revise or extend the existing knowledge. The revision and extension are first formulated as hypotheses. After being verified experimentally, a hypothesis is added to the existing knowledge and becomes part of the accepted biological theory.

Knowledge-based hypothesis formation has been a focus of Artificial Intelligence (AI) research in the past (Shrager & Langley, 1990; Darden, 1997). In regard to molecular biology and in particular signaling networks, the related works in hypothesis formation include HYPGENE (Karp, 1991, 1993a), HinCyc (Karp et al., 1996), TRANSGENE (Darden, 1998, 1997), GENEPATH (Zupan et al., 2003), and PathoLogic (Karp et al., 2002b). These works are built upon knowledge representation languages that are limited to *monotonic reasoning*. Furthermore, a large-scale knowledge base of signaling networks should allow for easy updates (referred to as *elaboration tolerance*) of the knowledge base when new knowledge becomes available, thus avoiding a significant overhaul of the old knowledge model. This issue of elaboration tolerance in knowledge representation has been addressed successfully by recent advances in AI research (Baral, 2003). Elaboration tolerance is essential for representation of signaling networks, since the knowledge about these networks is largely incomplete and constantly needs to be updated.

In this paper, we present a knowledge-based framework for hypothesis formation for signaling networks that is based on non-monotonic reasoning and elaboration tolerant representation. The organization of the paper is as follows. In the next section, we briefly introduce the action language $\mathcal{A}_T^0$ for representing signaling networks. In Section 3, we formally define the hypothesis formation problem in the language $\mathcal{A}_T^0$. In Section 4, we study the translation of the hypothesis formation problem into abduction, and the complexity of hypothesis formation as well as its implementation. In Section 5, we present an application of our system to the p53 signaling network. The proofs of theoretical results and additional technical background will be found in the Appendix.

2 Action Language for Signaling Networks

Representing and reasoning with knowledge about dynamic domains has been a major focus of research in Knowledge Representation and Reasoning - a

subfield of Artificial Intelligence. The most difficult and important research topics include: (1) compact representation of action descriptions; and (2) reasoning about effects of actions. In the last one and half decade there has been tremendous progress in representing and reasoning about actions and in applications of action languages to real world problems. If we consider the cell as a dynamic world, then issues in reasoning about actions are highly relevant to the reasoning about the cell discussed above (i.e., representing molecule interactions and doing prediction, planing and diagnosis). At the same time, the complexity of the cellular environments and mechanisms poses new and substantial challenges to established theories of reasoning about actions. We have decided to adopt the action language framework to build a knowledge base of signaling networks, since the expected benefits are twofold. On one hand, we would be able make use of recent advances in reasoning about actions. On the other hand, challenges in applying action theories to modeling the cell would stimulate significant new theoretical developments.

We extended the high-level action language $\mathcal{A}$ Gelfond & Lifschitz (1993) to an action language $\mathcal{A}_T^0$ for representing and reasoning about triggers in signaling networks. The language $\mathcal{A}_T^0$ has been applied to the representation of signaling pathways of the pRb, NFκB, ERK, and p53 protein.

2.1 Action language $\mathcal{A}$

An *action theory* in $\mathcal{A}$ Gelfond & Lifschitz (1993) is defined over two disjoint sets, a set of actions $\mathbf{A}$ and a set of fluents $\mathbf{F}$.

A *fluent literal* is a *fluent* (eg. f) or the negation of a fluent (eg. $\neg f$). A set of fluent literals is said to be *consistent* if it does not contain both f and $\neg f$ for some fluent f. An *interpretation* I of the fluents in $\mathcal{D}$ is a maximal consistent set of fluent literals of $\mathcal{D}$. A fluent f is said to be true (resp. false) in I if $f \in I$ (resp. $\neg f \in I$). The truth value of a fluent formula in I is defined recursively over the propositional connectives in the usual way. For example, $f \wedge g$ is true in I if f is true in I and g is true in I. We say that a formula φ holds in I (or I satisfies φ), denoted by $I \models \varphi$, if φ is true in I. A *state* is an interpretation of fluents.

A *domain description* is a set of statements of the form:

$$a \textbf{ causes } f \textbf{ if } \ f_1, \ldots, f_n \tag{1}$$

When $n = 0$, the above statement is simply written as a **causes** f.

Observations are statements about the initial state, which are of the form:

$$\textbf{initially } f$$

A set of observations O is said to be complete if for any fluent f, either **initially** $f \in O$ or **initially** $\neg f \in O$.

Queries in $\mathcal{A}$ are statements of the form:

$$Q = f \textbf{ after } a_1, \ldots, a_n \tag{2}$$

where f is a fluent literal, and $a_1, \ldots, a_n$ are actions. Intuitively, this statement queries whether f is true after the sequence of actions $a_1, \ldots, a_n$.

A *state transition* is a change of one state to another state due to effects of some actions. The effect of an action a in a state s is the set

$$E(a, s) = \{\ f \mid a \textbf{ causes } f \textbf{ if } \ f_1, \ldots f_n \in \mathcal{D} \text{ and } \{f_1, \ldots, f_n\} \subseteq s\ \}. \tag{3}$$

Let $\neg\neg g = g$ and $\neg E(a, s) = \{\neg g | g \in E(a, s)\}$. State transitions are computed by the transition function defined as follows.

Definition 2.1 *A transition function of a domain $\mathcal{D}$ is a function Φ from pairs of actions and states into states such that:*

- *if $E(a, s)$ is consistent, then*

$$\Phi(a, s) = (s \setminus \neg E(a, s)) \cup E(a, s)\ ;$$

- *otherwise $\Phi(a, s)$ is undefined.*

An *action theory* is a pair (D, O), where D is a domain description and O is a set of observations. A state s_0 is an initial state corresponding to an action theory (D, O) if for every fluent literal g, $g \in s_0$ iff **initially** $g \in O$. We then say that $\langle s_0, \Phi\rangle$ is a model of (D, O).

An action theory (D, O) entails a query Q of the form (2), if for all models $\langle s_0, \Phi\rangle$ of (D, O), f holds in the state $\Phi(a_n, \Phi(a_{n-1}, \ldots \Phi(a_1, s_0) \ldots))$. The entailment is denoted $(D, O) \models Q$.

2.2 Language $\mathcal{A}^0_T$ for triggered actions

The action language $\mathcal{A}^0_T$ extends $\mathcal{A}$ with statements representing triggered actions. A domain description $\mathcal{D}$ in $\mathcal{A}^0_T$ is a set of statements of the form (1) and of the following forms:

$$g_1, \ldots, g_m \textbf{ triggers } b \tag{4}$$
$$h_1, \ldots, h_l \textbf{ inhibits } c \tag{5}$$

where g_j, h_k are fluent literals and b, c are individual actions. (4) is called a *trigger* rule (or simply trigger), which says that action b is to occur if it is not inhibited and if all the literals $g_1, \ldots, g_m$ hold. (5) is an *inhibition* rule, which says that action c can not happen if all the literals $h_1, \ldots, h_l$ hold.

An action a is said to be *triggered* by a state s, if there exists a trigger rule (4) such that all the literals $g_1, \ldots, g_m$ are true in s. An action a is said to be *inhibited* by a state s, if there exists a inhibition rule (5) such that all the literals $h_1, \ldots, h_l$ are true in s.

Since a state can trigger multiple actions, *state transitions* in $\mathcal{A}_T^0$ are extended to pairs of sets of actions and states. The direct effect of a set A of actions in a state s is the set

$$E(A, s) = \bigcup_{a \in A} E(a, s).$$

where $E(a, s)$ is defined by (3). Let us denote $\neg E(A, s) = \{\neg f \mid f \in E(A, s)\}$. The state $\Phi(A, s)$ resulting from the occurrence of A in s is defined as follows.

- $\Phi(\emptyset, s) = s$;
- if $A \neq \emptyset$ and $E(A, s)$ is consistent, then

$$\Phi(A, s) = (s \setminus \neg E(A, s)) \cup E(A, s) \ ;$$

- otherwise $\Phi(A, s)$ is undefined.

A *transition sequence* τ is a sequence of the form $\tau = \langle s_0, A_0, s_1, A_1, \ldots \rangle$; where s_i's are states and A_j's are sets of actions in $\mathcal{D}$, such that $s_{i+1} = \Phi(A_i, s_i)$ for all i, and $A_j = \emptyset$ for all $j > k$ if $A_k = \emptyset$.

A *trajectory* is a transition sequence $\tau = \langle s_0, A_0, s_1, A_1, \ldots \rangle$ where A_i is a the set of all actions that are triggered but not inhibited by the state s_i (for all $i \geq 0$). *Observations* are statements of the form "f **at** i" or of the form "a **occurs_at** j", where i and j are non-negative integers. The former statement means that the fluent literal f is observed to be true at time i. The latter means that the action a is observed to occur at time j. A trajectory $\tau = \langle s_0, A_0, s_1, A_1, \ldots \rangle$ satisfies "f **at** i" iff $f \in s_i$. Also, τ satisfies "a **occurs_at** j" iff $a \in A_j$.

Definition 2.2 (Trigger bounded domain) *A domain description $\mathcal{D}$ is called* trigger bounded*, if all trajectories in $\mathcal{D}$ with only triggered actions are finite.*

The upper bound on the lengths of the trajectories of a triggered bounded domain $\mathcal{D}$ is denoted *tbound*($\mathcal{D}$).

A *query* in $\mathcal{A}_T^0$ has the form

$$Q = f \textbf{ after } A_1 \textbf{ at } t_1, \ldots, A_n \textbf{ at } t_n \tag{6}$$

where f is a fluent, $A_1, \ldots, A_n$ are sets of actions and $t_1 < \ldots < t_n$ are time points. When $n = 0$, we simply write $Q = f$.

A *action theory* is a pair $(\mathcal{D}, \mathcal{O})$ where $\mathcal{D}$ is a domain description and $\mathcal{O}$ is a set of observations. A *model* of an action theory $(\mathcal{D}, \mathcal{O})$ is a trajectory of $\mathcal{D}$ that satisfies all the observations of $\mathcal{O}$.

Let $\mathcal{T} = (\mathcal{D}, \mathcal{O})$ be an action theory and Q be the query (6). Let $\mathcal{O}'$ be the set of observations $\mathcal{O}' = \mathcal{O} \cup \{A_1$ **occurs_at** $t_1, \ldots, A_n$ **occurs_at** $t_n\}$. Then $\mathcal{T}$ entails Q, written as $\mathcal{T} \models Q$, iff

(i) $(\mathcal{D}, \mathcal{O}')$ has at least one model; and
(ii) for all trajectory models $\tau = \langle s_0, A'_1, s_1, A'_2 \ldots A'_m, s_m \ldots \rangle$ of the theory $(\mathcal{D}, \mathcal{O}')$, there exists N such that f is true in all the states s_k, $k > N$.

3 Knowledge-based Hypothesis Formation

Let $\mathcal{L}$ be a knowledge representation language. Assume that $\mathcal{L}$ is composed of 3 sub-languages: (1) a domain description language $\mathcal{L}_D$; (2) an observation language $\mathcal{L}_O$; and (3) a query language $\mathcal{L}_Q$. A domain description, an observation and a query are respectively sets of statements in $\mathcal{L}_D$, in $\mathcal{L}_O$ and in $\mathcal{L}_Q$. A theory is a pair (D, O) where D is a domain description and O is an observation. Assume that the semantics of $\mathcal{L}$ defines the entailment of a query Q from a theory (D, O), which is written as $(D, O) \models Q$.

Definition 3.1 *Let (D, O) be a theory in a knowledge representation language $\mathcal{L}$. Let Q be a query such that $(D, O) \not\models Q$. Let $\mathcal{S}$ be a set of sentences in $\mathcal{L}_D$, which is called* hypothesis space. *Let $\preceq$ be a partial order on the set of all $\mathcal{L}$ theories, which is called a* preference relation. *A* candidate *hypothesis is subset H of $\mathcal{S}$ such that $(D \cup H, O) \models Q$. A* hypothesis *H is a maximally preferred candidate hypothesis; that is, there exists no other candidate $H' \preceq H$.* □

We consider the hypothesis formation where the knowledge representation language is $\mathcal{L} \equiv \mathcal{A}^0_T$ and the preference $\preceq$ is based on the subset relation $\subseteq$.

Definition 3.2 (Hypothesis formation in $\mathcal{A}^0_T$) *A hypothesis formation problem (HFP) in $\mathcal{A}^0_T$ is given by a tuple $\langle D, O, Q, \mathcal{S} \rangle$ such that:*

- *(D, O) is a $\mathcal{A}^0_T$ theory where O is initial state complete; and*
- *Q is a query that cannot be entailed from the theory: $(D, O) \not\models Q$; and*
- *$\mathcal{S}$ is a set of rules whose fluent and action symbols are from the alphabet of D.*

Given an HFP $\mathcal{P} = \langle D, O, Q, \mathcal{S} \rangle$, a candidate hypothesis *is a subset H of*

$\mathcal{S}$ *such that* $(D \cup H, O) \models Q$. *A* hypothesis *(or* solution*) for* $\mathcal{P}$ *is a candidate hypothesis* H *such that there exists no candidate hypothesis* $H' \subset H$. *The set of the solutions for an HFP* $\mathcal{P} = \langle D, O, Q, \mathcal{S} \rangle$ *is denoted* $Sol(\mathcal{P})$ *or* $Sol(D, O, Q, \mathcal{S})$. □

Note that we have an additional restriction that $\mathcal{O}$ is initial state complete. Without the restriction, we would have to deal with two reasoning problems at the same time: hypothesis formation and explanation.

Example 3.1 *Let* D *be the domain description consisting of the rules:*

$$a \textbf{ causes } g$$
$$b \textbf{ causes } g$$

Let $O = \{f \textbf{ at } 0\}$ *and* $Q = \{g \textbf{ at } 1\}$. *Then* Q *is not entailed by* (D, O). *Now let* $\{f \textbf{ triggers } a; \quad f \textbf{ triggers } b\}$ *be the hypothesis search space. There are 3 candidate hypotheses:* $H_1 = \{f \textbf{ triggers } a\}$; $H_2 = \{f \textbf{ triggers } b\}$; *and* $H_3 = \{f \textbf{ triggers } a; \quad f \textbf{ triggers } b\}$. *Among the candidates,* H_1 *and* H_2 *are hypotheses, and* H_3 *is not.* □

4 Hypothesis Formation as Abduction

Besides deduction, abduction is another important kind of reasoning, which has been first studied in depth by Peirce Peirce (1958, 1992). Given the observation of some facts, abduction aims at concluding the presence of other facts, from which, together with an underlying theory, the observed facts can be explained.

Hypothesis formation is a typical abductive reasoning process: From the observations and the biological knowledge, a hypothesis about a possible theory is abduced. Notice that this form of reasoning is not sound, and that in general several abductive hypotheses for observations may be possible. Various forms of abductions have been defined in logics or logic programming, such as Kakas et al. (1998); Poole (1988, 1989); Lin & You (2002) and many others. The various definitions use the notion of *abducibles*. Due to logical translations, abducibles can be assumed to be ground predicates or literals. Abduction has been used and studied in various AI applications Poole (1989); Poole et al. (1998), abductive logic programming Kakas et al. (1998, 2001); Lin & You (2002); Denecker & Kakas (2002); Doherty et al. (2004), probabilistic reasoning Poole (1993), diagnosis Reggia (1983); Reiter (1980), planning Eshghi (1998); Allen et al. (1991); Missiaen et al. (1995), default reasoning Poole et al. (1987); Poole (1988); Eshghi & Kowalski (1989); Kakas et al. (1998), and belief revision and update Boutilier & Becher (1995); Boutilier (1996).

We shall relate our work to and make use of the study of logic programming framework of abduction presented in Eiter et al. (1997), which is recapped in the following.

Given an AnsProlog program Π (see Appendix B.1), the set of the atoms of Π is denoted $atom(\Pi)$. An atom f or its negation $\neg f$ is called a literal. The set of the literals of the atoms of Π is denoted $lit(\Pi)$.

Definition 4.1 (Entailment in AnsProlog) *Let Π be an AnsProlog program. Let M be an answer set of Π. For any $f \in atom(\Pi)$, f is entailed by M iff $f \in M$ and $\neg f$ is entailed by M iff $f \notin M$. The entailment of a literal l by M is denoted that $M \models l$. There are two kinds of entailments by AnsProlog programs:*

Brave Reasoning*: A literal l is entailed by the AnsProlog program Π, denote $\Pi \models^b l$, iff l is entailed by at least one answer set of Π.* □

Cautious Reasoning*: A literal l is entailed by the AnsProlog Π, denote $\Pi \models^c l$, iff Π has at least one answer set and l is entailed by all the answer sets of Π.*□

Definition 4.2 (Abduction in logic programming) *A logic programming abduction problem LPAP is a tuple $\langle S, F, \Pi, \models \rangle$, where $S \subseteq atom(\Pi)$ is a finite set of abducibles called* hypothesis space*; $O \subseteq lit(\Pi)$ is a finite set of literals called observations; Π is an AnsProlog program and $\models$ is an entailment operator in $\{\models^b, \models^c\}$. A set $H \subseteq S$ is a solution for $\langle S, F, \Pi, \models \rangle$ iff $\Pi \cup S \models O$. The set of the solutions for an LPAP $\mathcal{P} = \langle S, F, \Pi, \models \rangle$ is denoted $Sol(\mathcal{P})$.* □

In computing solutions for an LPAP $\mathcal{P} = \langle S, F, \Pi, \models \rangle$, the following decision problems are important:

- *consistency*: does there exist a solution for $\mathcal{P}$?
- *relevance*: does a given abducible h belong to some solution of $\mathcal{P}$?
- *necessity*: does a given abducible h belong to all the solutions of $\mathcal{P}$?

Definition 4.3 *Let $\mathcal{P} = \langle S, F, \Pi, \models \rangle$ and $h \in S$. Then h is* relevant *to $\mathcal{P}$ iff $h \in H$ for some solution H of $\mathcal{P}$, and h is* necessary *for $\mathcal{P}$ iff $h \in H$ for all solution H of $\mathcal{P}$.* □

Let $\mathcal{P} = \langle D, O, Q, \mathcal{S} \rangle$ be a hypothesis formation problem. Let us assume that the domain D is trigger bounded (see Definition 2.2). Besides, let Q be the query

$$Q = f \textbf{ after } A_1 \textbf{ at } t_1, \ldots, A_n \textbf{ at } t_n \tag{7}$$

We shall show that $\mathcal{P}$ can be transformed to an abduction problem of the

form $trans(\mathcal{P}) = \langle ab(\mathcal{S}), \pi_f(Q), \pi(D) \cup \pi(O) \cup \pi_a(Q) \cup \pi(\mathcal{S}), \models^b \rangle$. The $\pi(D)$ and $\pi(O)$ is the following translation of D and O into AnsProlog programs.

4.1.1 The translation $\pi(D)$

The upper bound of time steps in $\pi(D)$ is $t_{max} = t_n + tbound(D)$. Here, t_n is the maximal time point of action occurrences in Q and $tbound(D)$ is the upper bound of the lengths of the trajectories of D.

Given a fluent literal g and some fluent f, let us denote $\pi(g,t) \equiv holds(f,t)$ if $g \equiv f$; and let $\pi(g,t) \equiv holds(neg(f),t)$ if $g \equiv \neg f$. Given an action a, let $\pi(a,t) \equiv holds(occurs(a),t)$. The program $\pi(D)$ includes *inertial rules*, *interpretation constraints* and the translations of all the propositions of D.

The set of inertial rules include the following rules, for each fluent f and for all time points t in $[0, t_{max})$:

$$\pi(f, t+1) \leftarrow \pi(f,t),\ \ not\ \pi(\neg f, t+1)$$
$$\pi(\neg f, t+1) \leftarrow \pi(\neg f, t),\ not\ \pi(f, t+1)$$

For each fluent f, there are interpretation constraints of the following form, for all time point t in $[0, t_{max}]$:

$$\bot \leftarrow holds(f,t), holds(neg(f),t)$$

Intuitively, the interpretation constraints guarantee that both fluent literal f and $\neg f$ cannot hold at the same time.

The propositions of D are translated as follows.

- A causal rule " a **causes** f **if** $f_1, \ldots, f_i$ " is translated into the set consisting of the following rules, $\forall t \in [0, t_{max})$:

$$\pi(f, t+1) \ \leftarrow\ \pi(a,t),\ \pi(f_1,t),\ \ldots\ ,\ \pi(f_i,t)$$

- A trigger rule " $g_1, \ldots, g_j$ **triggers** b " is translated into the set consisting of the following rules, $\forall t \in [0, t_{max})$:

$$\pi(b,t) \ \leftarrow\ \pi(g_1,t),\ \ldots\ ,\ \pi(g_j,t),\ not\ holds(ab(occurs(b)),t)$$

- An inhibitor rule " $h_1, \ldots, h_k$ **inhibits** c " is translated to the set consisting of the following rules, $\forall t \in [0, t_{max})$:

$$holds(ab(occurs(c)),t) \ \leftarrow\ \pi(h_1,t),\ \ldots\ ,\ \pi(h_k,t).$$

4.1.2 The translation $\pi(O)$

The program $\pi(\mathcal{O})$ consists of the translations of all the observations of $\mathcal{O}$. The observations in $\mathcal{O}$ are translated as follows.

- An observation of the form " **initially** f " is translated into the fact $holds(f, 0) \leftarrow$.
- If $t > 0$, an observation of the form " f **at** t " is translated into $\bot \leftarrow not\ holds(f, t)$.
- If a is a triggered action and t is a time point, then the observation " a **occurs_at** t " is translated into the constraint $\bot \leftarrow not\ holds(occurs(a), t)$.
- If a is a non-triggered action, and t is a time point then the observation " a **occurs_at** t " is translated into the fact $holds(occurs(a), t) \leftarrow$.

4.1.3 Transforming the query Q

The transformation of query Q includes $\pi_f(Q)$ and $\pi_a(Q)$. Given that Q is of the form in (7), $\pi_f(Q) = \{holds(f, t_{max})\}$ and $\pi_a(Q)$ is the set consisting of all the translations of the observations " A_i **occurs_at** t_i ", $i = 1, \ldots n$.

4.1.4 Transforming the hypothesis space $\mathcal{S}$

The transformation of $\mathcal{S}$ is two-fold, which includes the set $ab(\mathcal{S})$ of special atoms and the AnsProlog program $\pi(\mathcal{S})$. Let *label* be a 1-1 function from the element of $\mathcal{S}$ to a set of string labels. First, the set $ab(\mathcal{S})$ simply consists of all the atoms $picked(label(r))$ where $r \in \mathcal{S}$:

$$ab(\mathcal{S}) = \{picked(label(r)) \mid r \in \mathcal{S}\}$$

The AnsProlog program $\pi(\mathcal{S})$ consists of the translations of the rules of $\mathcal{S}$. The translation of a rule r of $\mathcal{S}$ is as follows.

- If r is a causal rule of the form " a **causes** f **if** $f_1, \ldots, f_i$ " , then the translation of r includes the following rules, $\forall t \in [0, t_{max})$:

$$\pi(f, t+1) \leftarrow \pi(a, t),\ \pi(f_1, t),\ \ldots\ ,\ \pi(f_i, t),\ picked(label(r))$$

- If r is a trigger rule of the form " $g_1, \ldots, g_j$ **triggers** b ", then the translation of r includes the following rules, $\forall t \in [0, t_{max})$:

$$\pi(b, t) \leftarrow \pi(g_1, t), \ldots, \pi(g_j, t),\ not\ holds(ab(occurs(b)), t), picked(label(r))$$

- If r is an inhibitor rule " $h_1, \ldots, h_k$ **inhibits** c ", then the translation of r includes the following rules, $\forall t \in [0, t_{max})$:

$$holds(ab(occurs(c)), t) \leftarrow \pi(h_1, t),\ \ldots\ ,\ \pi(h_k, t), picked(label(r))$$

Proposition 4.1 *Let $\mathcal{P} = \langle D, O, Q, \mathcal{S}\rangle$ be a hypothesis formation problem. Let $trans(\mathcal{P}) = \langle ab(\mathcal{S}), \pi_f(Q), \pi(D) \cup \pi(O) \cup \pi_a(Q) \cup \pi(\mathcal{S}), \models^b\rangle$ be the transformation of $\mathcal{P}$ into abduction. Then $H \subseteq \mathcal{S}$ is a solution of $\mathcal{P}$ if and only if the set $\{picked(label(r)) \mid r \in H\}$ is a solution of the LPAP $trans(\mathcal{P})$.*

4.2 Complexity Analysis

We have showed in the previous section that a hypothesis formation problem $\mathcal{P} = \langle D, O, Q, \mathcal{S}\rangle$ can be transformed in to the abduction problem $trans(\mathcal{P}) = \langle \pi(S), \pi_f(Q), \pi(D) \cup \pi(O) \cup \pi_a(Q), \models^b\rangle$. In light of Proposition 4.1, computing hypothesis formation is not harder than computing abduction. Thus upper bounds for the complexity of hypothesis formations are easily obtained from the complexity for abduction. Particularly, we shall make use of the following result by (Eiter et al., 1997).

Theorem 4.1 *(Eiter et al., 1997) Let $\mathcal{P} = \langle S, O, \Pi, \models^b\rangle$ be an LPAP problem. Let $H \subseteq S$ and $h \in S$.*

- *Deciding if $Sol(\mathcal{P}) \neq \emptyset$ is NP-complete.*
- *Deciding if h is relevant to $\mathcal{P}$ is Σ_2^P-complete.*
- *Deciding if h is necessary for $\mathcal{P}$ is coNP-complete.*

Similar to the case of abduction, the notion of relevance and necessity are defined for hypothesis formation.

Definition 4.4 *Let $\mathcal{P} = \langle D, O, Q, \mathcal{S}\rangle$ be a hypothesis formation problem. A rule $r \in \mathcal{S}$ is relevant to $\mathcal{P}$ if $r \in \mathcal{S}$ for some solution H of $\mathcal{P}$. A rule $r \in \mathcal{S}$ is necessary for $\mathcal{P}$ if r belongs to all the solutions of $\mathcal{P}$.*

It can be showed that an $\mathcal{A}_T^0$ action theory (D, O) where O is initial state complete has at most one model. Consequently, augmented with a hypothesis, the AnsProlog program in the transformation $trans(\mathcal{P})$ of a hypothesis formation problem $\mathcal{P}$ has at most one answer set. Then it would be expected that the complexity of hypothesis formation would be lower than that of abduction with a general AnsPrlog program. However, it is not the case. We have the following result for the complexity of hypothesis formation.

Proposition 4.2 *Let $\mathcal{P} = \langle D, O, Q, \mathcal{S}\rangle$ is a hypothesis formation problem. Let $H \subseteq \mathcal{S}$ and $r \in \mathcal{S}$.*

- *Deciding if $Sol(\mathcal{P}) \neq \emptyset$ is NP-complete.*
- *Deciding if $r \in \mathcal{S}$ is relevant to $\mathcal{P}$ is Σ_2^P-complete.*
- *Deciding if $r \in \mathcal{S}$ is necessary to $\mathcal{P}$ is coNP-complete.*

Based on the complexity analysis, a straightforward implementation of hypothesis formation is to translate to abduction in AnsProlog. Given a problem $\mathcal{P} = \langle D, O, Q, \mathcal{S} \rangle$, the translations of the components of $\mathcal{P}$ into AnsProlog are readily available. What needed is to augment the AnsProlog engine with the ability to compute the minimality of hypotheses (i.e., abductive solutions). As it has turned out, this can be done in CR-Prolog - an extension of AnsProlog (see Appendix B.2).

Let $\mathcal{P} = \langle D, O, Q, \mathcal{S} \rangle$. Recall that $\mathcal{P}$ can be transformed to the LPAP $trans(\mathcal{P}) = \langle ab(\mathcal{S}), \pi_f(Q), \pi(D) \cup \pi(O) \cup \pi_a(Q) \cup \pi(\mathcal{S}), \models^b \rangle$. Let $\Pi_{CR}(\mathcal{P})$ denote the encoding of $\mathcal{P}$ in CR-Prolog. Then:

$$\Pi_{CR}(\mathcal{P}) = \{\leftarrow \ not\ f \mid f \in \pi_f(Q)\} \cup \pi(D) \cup \pi(O) \cup \pi_a(Q) \cup \pi_{CR}(\mathcal{S})$$

where $\pi_{CR}(\mathcal{S})$ is the translation of $\mathcal{S}$ into CR-Prolog.

The translation $\pi_{CR}(\mathcal{S})$ of $\mathcal{S}$ includes the translation $\pi_{CR}(r)$ of all the rules r of $\mathcal{S}$. Let *label* be a 1-1 function from the element of $\mathcal{S}$ to a set of string labels. Let r be any rule of $\mathcal{S}$. The translation $\pi_{CR}(r)$ of r into CR-Prolog is as follows.

- If r is a causal rule of the form " a **causes** f **if** $f_1, \ldots, f_i$ " , then $\pi_{CR}(r)$ includes the following rules, $\forall t \in [0, t_{max})$:

$$label(r): \quad \pi(f, t+1) \stackrel{+}{\leftarrow} \pi(a, t),\ \pi(f_1, t),\ \ldots\ ,\ \pi(f_i, t)$$

- If r is a trigger rule of the form " $g_1, \ldots, g_j$ **triggers** b ", then $\pi_{CR}(r)$ includes the following rules, $\forall t \in [0, t_{max})$:

$$label(r): \quad \pi(b, t) \stackrel{+}{\leftarrow} \pi(g_1, t), \ldots, \pi(g_j, t),\ not\ holds(ab(occurs(b)), t)$$

- If r is an inhibitor rule " $h_1, \ldots, h_k$ **inhibits** c ", then the translation $\pi_{CR}(r)$ includes the following rules, $\forall t \in [0, t_{max})$:

$$label(r): \quad holds(ab(occurs(c)), t) \stackrel{+}{\leftarrow} \pi(h_1, t),\ \ldots\ ,\ \pi(h_k, t)$$

5 Biological Application

From the user perspective, the process of hypothesis formation is declarative. The process involves three steps: (i) construct the knowledge base K; ii) formulate the novel experiment to be explained; i.e., the initial condition I and

novel observation O; (iii) construct the hypothesis space $\mathcal{S}$. Usually, step *(i)* is not necessary since the knowledge base K already exists. Step *(ii)* amounts to the translation of experimental observations into causal, trigger and inhibition rules. Step *(iii)* amounts to the formulation of domain background knowledge that is relevant to hypothesis formation. In this work, *(iii)* has been done manually based on biological research literature, but it will be automated in the future development.

We now present the p53 signal network as a case study to illustrate the process of hypothesis formation. First, we describe the biology of the p53 network during cancer in human cells. We present the biological description in parallel with its knowledge-based representation.

5.1 p53 signal network

The p53 protein plays a central role as a tumor suppressor and is subjected to tight control through a complex mechanism involving several proteins. The key aspects of the p53 network are as follows.

5.1.1 Tumor suppression by p53

The p53 protein has three main functional domains; the N terminal transactivator domain, the central DNA-binding domain and a C terminal domain that recognizes DNA damage. The binding of the transactivator domain to the promoters of target genes activates pathways to lead to a reversible arrest of the cell cycle, prevention of genomic instability or apoptosis and thus protects the cell from cancer (Michael & Oren, 2002). The level and activity of p53 in the cell is influenced by its interactions with other proteins. The ability to suppress tumors is retained when the interacting partners of p53 do not inhibit the functionality of the transactivator domain.

$$high(p53) \textbf{ inhibits } grow(tumor)$$
$$high([p53 : mdm2]),\ not\ bound(dom(p53, N)) \textbf{ inhibits } grow(tumor)$$

Here, $[A : B]$ denotes the complex of protein A and B.

5.1.2 Interaction between Mdm2 and p53

Mdm2 binds to the transactivator domain of p53, thus inhibiting the p53 induced tumor suppression. The binding of Mdm2 to p53 also causes changes

in the protein concentration levels.

$$bind(p53, mdm2) \textbf{ causes } bound(dom(p53, N))$$
$$high(p53), high(mdm2) \textbf{ triggers } bind(p53, mdm2)$$
$$bind(p53, mdm2) \textbf{ causes } high([p53 : mdm2]),$$
$$bind(p53, mdm2) \textbf{ causes } \neg high(p53), \neg high(mdm2)$$

5.1.3 Upregulation of p53

The elevated levels of p53 may be a result of upregulation of p53 gene expression, increased transcript stability, enhanced translation of p53 mRNA (Hamid & Kakar, 2004), or post-translational modifications of the p53 protein which favor a prolonged half life and increased activity (Bode & Dong, 2004). The upregulation of p53 expression can be represented as follows.

$$upregulate(mRNA(p53)) \textbf{ causes } high(mRNA(p53))$$
$$high(mRNA(p53)) \textbf{ triggers } translate(p53)$$
$$translate(p53) \textbf{ causes } high(p53)$$

5.1.4 Stress

UV, ionizing radiation, and chemical carcinogens cause stress. Stress can induce the upregulation of p53.

$$high(UV) \textbf{ triggers } upregulate(mRNA(p53))$$

Apart from tumor suppressor genes like p53, stress can also influence the expression of oncogenes (e.g., *cmyc*). The regulation of expression of tumor related genes involves stress sensing mechanisms and multiple signal transduction events, and appears to be a complex phenomenon. An abstract representation of the process is:

$$high(UV) \textbf{ triggers } sense(UV\_signal)$$
$$is\_sensed(UV) \textbf{ triggers } transduce(UV\_signal)$$
$$is\_transduced(UV\_signal) \textbf{ triggers } alter(expr(cmyc))$$
$$is\_altered(expr(cmyc)) \textbf{ triggers } grow(tumor)$$

Given the above knowledge base of the p53 network, a hypothesis formation problem arises as follows.

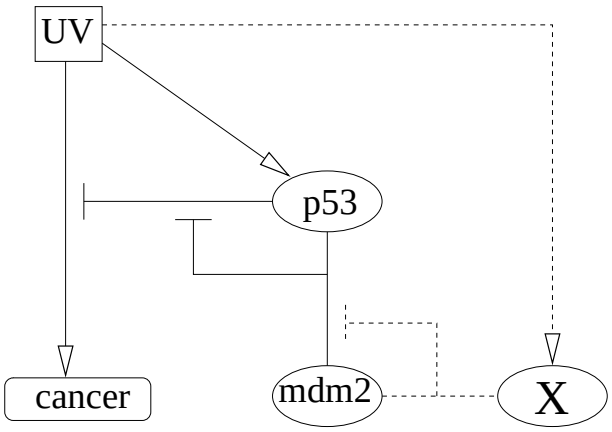

Fig. 1. A hypothesis in p53 interaction network. The $\rightarrow$ represents trigger. The $\dashv$ represents inhibition. The solid and dash lines represent known and hypothetical interactions, respectively.

5.2 Biological hypothesis formation

The original biological problem is as follows. *X is a tumor suppressor gene: mutants of X are highly susceptible to cancer and behave similarly to the mutants of p53. Our objective is to hypothesize about the various possible influences of X on the p53 pathway.*

Let us assume that in certain experiments, exposure of the cell to high level UV does not lead to cancer, given that the initial concentrations of p53 and Mdm2 are high. Besides, a high level of gene expression of the X protein is also observed in these cases. Thus we have a hypothesis formation problem $\mathcal{P} = \langle D, O, Q, \mathcal{S} \rangle$ such that:

- the original domain description is $D \equiv D_{p53}$, where D_{p53} is the domain description of the p53 network;
- the initial observation O is about the high concentrations of the proteins:

$$O = \{high(UV) \textbf{ at } 0,\ high(mdm2) \textbf{ at } 0,\ high(X) \textbf{ at } 0\}\ ;$$

- the query is whether cancer will occur:

$$Q = tumorous$$

It is straightforward to verify that $(D, O) \not\models Q$. Our next step is to construct the hypothesis space $\mathcal{S}$. Based on the literature (Michael & Oren, 2002; Hamid & Kakar, 2004; Bode & Dong, 2004), we formulate the rules to be included in $\mathcal{S}$ as follows.

5.3 Hypothesis Space $\mathcal{S}$

5.3.1 Functional similarities between X and p53

According to the literature, X is a tumor suppressor, so it may play the same role as p53 in stressed cells. The following rules are included in the $\mathcal{S}$, which describes that X may have interactions similar to those of p53:

$$high(X), high(mdm2) \textbf{ triggers } bind(X, mdm2)$$
$$high(X), high(mdm2) \textbf{ inhibits } bind(X, mdm2)$$

5.3.2 Stress-induced high level of X

Data from the literature show that the level of protein X is found to be higher in cells subjected to stress. Consequently, it is possible that stress induces the upregulation of X gene expression, resulting in an elevated level of X. Thus $\mathcal{S}$ includes this background knowledge in the form of the rule:

$$high(UV) \textbf{ triggers } upregulate(mRNA(X))$$

5.3.3 Correlation between X- and p53-induced upregulations

There are observations from the literature that high levels of X are concomitant with elevated levels of p53. Thus, it is possible that a high level of X induces the upregulation of p53, or vice versus. This background knowledge are captured by the following rules in the search space:

$$high(X) \textbf{ triggers } upregulate(mRNA(p53))$$
$$high(p53) \textbf{ triggers } upregulate(mRNA(X))$$

5.3.4 Interactions of X with the known proteins

There are possible interactions $bind(p53, X)$ and $bind(mdm2, X)$. The possible related properties are about the protein levels and the domains of p53. Hence, $\mathcal{S}$ includes rules that associate the possible actions with the possible effects, such as:

$$bind(p53, X) \textbf{ causes } bound(dom(p53, N))$$
$$high(p53), high(X) \textbf{ triggers } bind(p53, mdm2)$$
$$high(p53), high(X) \textbf{ inhibits } bind(p53, mdm2)$$

5.4 Result

We have constructed a set $\mathcal{S}$ consisting of 12 elements. The logic program output 5 hypotheses (which are subsets of $\mathcal{S}$). Among these, the most intuitive hypotheses are:

- *X is a negative regulator of Mdm2:* Stress induces high expression of X. Then X binds to Mdm2, which competes against thus inhibiting the Mdm2-p53 interaction. Hence, the p53 induced tumor suppression is preserved (Figure 1). The rules representing the hypothesis include:

$$high(UV) \textbf{ triggers } upregulate(mRNA(X))$$
$$high(X), high(mdm2) \textbf{ triggers } bind(X, mdm2)$$

- *X directly influences p53 protein stability:* X binds to p53 protein (possibly at a domain different from the transactivator domain), so p53 is stabilized (i.e., formation of Mdm2-p53 complex is prevented) and still functional as tumor suppressor. The rules representing the hypothesis include:

$$high(UV) \textbf{ triggers } upregulate(mRNA(X))$$
$$high(X), high(p53) \textbf{ inhibits } bind(p53, mdm2)$$
$$high(X), high(p53) \textbf{ triggers } bind(p53, X)$$

The other hypotheses have not been discussed here, since we were unable to provide meaningful biological interpretations. For example, a hypothesis suggests that X inhibits the formation of Mdm2-p53 complex by interacting with both the proteins:

$$high(UV) \textbf{ triggers } upregulate(mRNA(X))$$
$$high(X), high(mdm2) \textbf{ inhibits } bind(p53, mdm2)$$
$$high(X), high(p53) \textbf{ triggers } bind(p53, X)$$

We expect that such hypotheses can be eliminated by incorporating more background knowledge.

The non-monotonicity of the framework manifests itself in the results. The knowledge base in Section 5.1 predicts that cancer will finally occur due to high level of UV (stress). After being extended with the hypothesis described in Figure 1., the new knowledge base predicts that cancer may not occur, given the presence of UV.

6 Related Works

6.1 HYPGENE

HYPGENE (Karp, 1991, 1993a) treated the general problem of hypothesis formation as a *planning problem*. The actions are operators that modify an existing knowledge base and/or assumed initial conditions of an experiment. The goal of the planning problem is to resolve the mismatch between theoretical predictions computed by the knowledge base and experimental observations, with respect to the same initial conditions. The knowledge base was implemented using a frame-based knowledge representation language. HYPGENE was proposed to be domain-independent and has been tested on a problem of E. coli gene regulation.

HYPGENE and our approach tackle the same hypothesis formation problem that arises when an existing theory does not predict an experimental observation. Their major differences include:

- The frame-based representation language is limited to monotonic reasoning. Thus HYPGENE would have difficulty in dealing with incompleteness of biological knowledge.
- A hypothesis involves the modification of an existing knowledge base and/or assumed initial conditions of an experiment. HYPGENE was restricted to the modification of the initial conditions. This restricted problem amounts to a form of reasoning called *explanation* and has been studied in (Baral et al., 2004).

6.2 TRANSGENE

TRANSGENE (Darden, 1998, 1997) considered hypothesis formation as diagnosis and redesign of theories. According to this model, when a theory cannot predict an experimental observation, the theory must contain some faulty components that can be found and fixed. TRANSGENE used a "functional representation" language for knowledge representation (Sembugamoorthy & Chandrasekaran, 1986). This representation language was chosen to overcome the limitations of rule based and frame based system. Nevertheless, the language could not allow for non-monotonic reasoning. To sum up, TRANSGENE showed that limitations of knowledge representation language can seriously hinder hypothesis formation. On the other hand, it illustrates that hypothesis formation is intuitive and straightforward in knowledge based framework.

6.3 GenePath

GenePath (Zupan et al., 2003) automated the inference of genetic networks from experimental data. A knowledge base is a genetic network that represents positive and negative influences of a gene on another. Experiments are perturbations to the network, performed by means of gene mutations. A fixed set of inference rules was formalized and implemented in GenePath using Prolog. These rules encode heuristic reasoning that is routinely applied by geneticists, namely epistasis analysis. Prior background knowledge is encoded in an initial network. Starting with the initial network, GenePath applies the rules to construct a plausible network as a hypothesis that explains experimental data. GenePath can also propose new experiments for further verification and refinement of hypotheses. Although the knowledge representation and reasoning are simple in GenePath, it has illustrated the value of domain background knowledge in hypothesis formation, and that logic programming provides a straightforward and intuitive representation of human reasoning.

6.4 Robot Scientist

Robot Scientist (King & et. al., 2004) used machine learning techniques (active learning, decision tree, inductive logic programming) to predict gene function in metabolic networks. The knowledge representation language is a monotonic logical formalism implemented in Prolog. The system demonstrated state-of-the-art AI methods, especially machine learning and robotics. However, it is unclear how the system can incorporate elaboration representation and non-monotonic reasoning into hypothesis formation.

6.5 Approximate Database

Doherty et.al. (Doherty et al., 2004) presented a first-order logic representation of biochemical reactions in metabolic pathways. The logical representation is implemented in approximate database, which supported reasoning about pathways in form of asking database queries. Hypotheses about missing components of pathways are generated by abductive reasoning based on weakest sufficient and strongest necessary conditions (Lin & You, 2002). The system has been illustrated with an aromatic amino acid pathway in yeast. Like the other related works, the major drawback of this work is the way it represents and reasons with pathway knowledge.

6.6 BIOCHAM

Calzone et.al. (Laurence Calzone et al., 2005) presented a system for learning biochemical interactions in the formal system BIOCHAM (Fages et al., 2004). A knowledge base in the BIOCHAM is a set of rules representing biochemical reactions. Experimental observations are expressed by temporal logic formulas. When a knowledge base does not satisfy observed temporal logic formulas, the learning system can be invoked interactively to refine the knowledge base. The search for refinement is an exhaustive enumeration and verification of rules whose patterns are given a priori. The system is implemented with symbolic model checking and has been evaluated with different small examples of cell cycles. Although the languages for knowledge representation and reasoning are monotonic in BIOCHAM, the model checking approach can lead to effective modeling and construction of large scale biochemical interaction networks.

7 Conclusion

We have presented a knowledge based formulation of the hypothesis formation problem. We then studied the hypothesis formation for signaling networks in the context of reasoning about actions. The advantages of our approach include:

- Hypothesis formation is defined as a form of reasoning and is implemented using AnsProlog, which is an elaboration tolerant and non-monotonic representation and reasoning language;
- Hypothesis formation in our framework is highly declarative.
- The user-level building of knowledge, the design of knowledge model (e.g., the language $\mathcal{A}_T^0$), and the computational implementation (i.e., AnsProlog engine) are highly independent from one another. This modularity will facilitate research and development of large-scale knowledge bases.

The case study of the p53 network is a proof of concept of our approach. Substantial work remains for hypothesis formation for larger networks such as (Kohn, 1999; Davidson et al., 2002a; Oda et al., 2004).

References

Allen, J., Kautz, H., Pelavin, R., & Tenenberg, J. (1991). *Reasoning about plans*. Morgan Kaufmann, San Mateo,CA.

Alur, R., Belta, C., Ivanicic, F., Kumar, V., Mintz, M., Pappas, G. J., Rubin, H., & Schug, J. (2001). Hybrid modeling and simulation of biomolecular networks. *Hybrid Systems: Computation and Control. LNCS*, **2034**, 19–32.

Balduccini, M. (2005). *Answer Set Based Design of Highly Autonomous, Rational Agents*. PhD thesis. Texas Tech University.

Balduccini, M. & Gelfond, M. (2003). Logic programs with consistency-restoring rules.

Baral, C. (2003). *Knowledge representation, reasoning and declarative problem solving*. Cambridge University Press.

Baral, C., Chancellor, K., Tran, N., Tran, N., & Berens, M. (2004). A knowledge based approach for representing and reasoning about signaling networks. *Bioinformatics 20 (Suppl 1)*, pages i15–i22.

Batt, G., de Jong, H., Geiselmann, J., & Page, M. (2003). Analysis of genetic regulatory networks: A model-checking approach. In *International Workshop on Qualitative Reasoning (QR'03)*.

Bode, A. M. & Dong, Z. (2004). Post-translational modification of p53 in tumorigenesis. *Nat. Rev. Cancer.*, **4**(10), 793–805.

Boutilier, C. (1996). Abduction to plausible causes: An even based model of belief update. *Artif. Intell.*, **83**, 143–166.

Boutilier, C. & Becher, V. (1995). Abduction as belief revision. *Artif. Intell.*, **77**, 43–94.

Chiaverini, M. & Danos, V. (2003). A core modeling language for the working molecular biologist. In *Computational Methods in Systems Biology (CMSB 2003)*, page 166.

Cook, D., Farley, J., & Tapscott, S. (2001). A basis for a visual language for describing, archiving and analyzing functional models of complex biological systems. *Genome Biology*, **2**(4).

Danos, V. & Laneve, C. (2003). Graphs for core molecular biology. In *Computational Methods in Systems Biology (CMSB 2003)*, pages 34–46.

Darden, L. (1997). Recent work in computational scientific discovery. In *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society*, pages 161–166.

Darden, L. (1998). Anomaly-driven theory redesign: computational philosophy of science experiments. *Digital Phoenix: how computers are changing philosophy*, pages 62–78.

Davidson, E. H., Rast, J. P., Oliveri, P., Ransick, A., Calestani, C., Yuh, C.-H., Minokawa, T., Amore, G., Hinman, V., Arenas-Mena, C., Otim, O., Brown, C. T., Livi, C. B., Lee, P. Y., Revilla, R., Rust, A. G., Pan, Z. j., Schilstra, M. J., Clarke, P. J. C., Arnone, M. I., Rowen, L., Cameron, R. A., McClay, D. R., Hood, L., & Bolouri, H. (2002a). A genomic regulatory network for development. *Science*, **295**(5560), 1669–1678.

Davidson, E. H., Rast, J. P., Oliveri, P., Ransick, A., Calestani, C., Yuh, C.-H., Minokawa, T., Amore, G., Hinman, V., Arenas-Mena, C., Otim, O., Brown, C. T., Livi, C. B., Lee, P. Y., Revilla, R., Rust, A. G., jun Pan, Z., Schilstra, M. J., Clarke, P. J. C., Arnone, M. I., Rowen, L., Cameron, R. A., McClay, D. R., Hood, L., & Bolouri, H. (2002b). A genomic regulatory network for development. *Science*, **295**, 1669–1678.

de Jong, H. (2002). Modeling and simulation of genetic regulatory systems: A literature review. *Journal of Computational Biology*, **9**(1), 67–103.

de Jong, H., Geiselmann, J., Hernandez, C., & Page, M. (2003). Genetic Network Analyzer: qualitative simulation of genetic regulatory networks. *Bioinformatics*, **19**, 336–344.

Demir, E., Babur, O., Dogrusoz, U., Gursoy, A., Ayaz, A., Gulesir, G., Nisanci, G., & Cetin-Atalay, R. (2004). An ontology for collaborative construction and analysis of cellular pathways. *Bioinformatics*, **20**(3), 349–356.

Denecker, M. & Kakas, A. C. (2002). Abduction in logic programming. In *Computational Logic: Logic Programming and Beyond*, pages 402–436.

Doherty, P., Kertes, S., Magnusson, M., & Szalas, A. (2004). Towards a logical analysis of biochemical pathways. In *Proc. JELIA*.

Eiter, T., Gottlob, G., & Leone, N. (1997). Abduction from logic programs: semantics and complexity. *Theoretical Computer Science*, **189**(1–2), 129–177.

Eker, S., Knapp, M., Laderoute, K., Lincoln, P., JosMeseguer, & Sonmez, K. (2002). Pathway logic: symbolic analysis of biological signaling. In *Pacific Symposium on Biocomputing 2002 (PSB 2002)*, page 400412.

Eshghi, K. (1998). Abductive planning with event calculus. In *Proc. ICLP*, pages 562–579.

Eshghi, K. & Kowalski, R. (1989). Abduction compated with negation as failure. In *Proc. ICLP*, pages 234–255.

Fages, F., Soliman, S., & Chabrier-Rivier, N. (2004). Modelling and querying interaction networks in the biochemical abstract machine BIOCHAM. *Jour. Biol. Phys. Chem.*, **4**(2), 64–73.

Fukuda, K. & Takagi, T. (2001). Knowledge representation of signal transduction pathways. *Bioinformatics*, **17**(9), 829–37.

Fukuda, K., Yamagata, Y., & Takagi, T. (2003). Frex: a query interface for biological processes with a hierarchical and recursive structures. *In Silico Biology*, **4, 0007**.

Funahashi, A., Tanimura, N., Morohashi, M., & Kitano, H. (2003). Celldesigner: a process diagram editor for gene-regulatory and biochemical networks. *BIOSILICO*, **1**(5), 159–162.

Gelfond, M. & Lifschitz, V. (1988). The stable model semantics for logic programming. In *Proc. ICLP*, pages 1070–1080.

Gelfond, M. & Lifschitz, V. (1993). Representing action and change by logic programs. *Journal of Logic Programming*, **17**(2/3&4), 301–321.

Hamid, T. & Kakar, S. (2004). PTTG/securin activates expression of p53 and modulates its function. *Mol. Cancer.*, **3**(1), 18.

Heidtke, K. & Schulze-Kremer, S. (1998). Design and implementation of a qualitative simulation model of lambda phage infection. *Bioinformatics*, **14**, 81–91.

Hoffmann, A., Levchenko, A., Scott, M. L., & Baltimore, D. (2002). The ikappabnfkappab signaling module: Temporal control and selective gene activation. *Science*, **298**, 1241–1245.

Kakas, A., Kowalski, R., & Toni, F. (1998). The role of abduction in logic programming. *Handbook of logic in Artificial Intelligence and Logic Programming*, pages 235–324.

Kakas, Antonis, C., Van Nuffelen, B., & Denecker, M. (2001). A-system : Problem solving through abduction. In *Proc. IJCAI*, volume 1, pages 591–596.

Kam, N., Cohen, I., & Harel, D. (2001). The immune system as a reactive system: modeling T-cell activation with statecharts.

Karp, P. (1991). Artificial intelligence methods for theory representation and hypothesis formation. *Comput. Appl. Biosci.*, **7**(3), 301–308.

Karp, P. D. (1993a). Design methods for scientific hypothesis formation and their application to molecular biology. *Machine Learning*, **12**, 89–116.

Karp, P. D. (1993b). A qualitative biochemistry and its application to the regulation of the tryptophan operon. *Artificial Intelligence and Molecular Biology*, pages 289–324.

Karp, P. D., Ouzounis, C., & Paley, S. (1996). HinCyc: A Knowledge Base of the Complete Genome and Metabolic Pathways of H. influenzae. In *Proc. ISMB*.

Karp, P. D., Riley, M., Saier, M., Paulsen, I. T., Paley, S. M., & Pellegrini-Toole, A. (2000). The ecocyc and metacyc databases. *Nucleic Acids Research*, **28**(1), 56–59.

Karp, P. D., Riley, M., Saier, M., Paulsen, I. T., Collado-Vides, J., Paley, S. M., Pellegrini-Toole, A., Bonavides, C., & Gama-Castro, S. (2002a). The ecocyc database. *Nucleic Acids Research*, **30**(1), 56–58.

Karp, P. D., Paley, S., & Romero, P. (2002b). The pathway tools software. *Bioinformatics*, **18**(Suppl. 1), S225–S232.

King, R. & et. al. (2004). Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature*, **427**(6971), 247–52.

Kitano, H. (2003). A graphical notation for biological networks. *BioSilico*, **1**(5), 169–176.

Kohn, K. W. (1999). Molecular interaction map of the mammalian cell cycle control and DNA repair systems. *Molecular Biology of the Cell*, **10**, 2703–2734.

Krieger, C. J., Zhang, P., Mueller, L. A., Wang, A., Paley, S., Arnaud, M., Pick, J., Rhee, S. Y., & Karp, P. D. (2004). Metacyc: A multiorganism database of metabolic pathways and enzymes. *Nucleic Acids Research*, **32**, D438–42.

Laurence Calzone, Nathalie Chabrier-Rivier, Francois Fages, & Sylvain Soliman (2005). A machine learning approach to biochemical reaction rules discovery. In *Proc. FOSBE*.

Lifschitz, V. & Turner, H. (1994). Splitting a logic program. In P. V. Hentenryck, editor, *Proceedings of the Eleventh International Conference on Logic Programming*, pages 23–38.

Lifschitz, V. & Turner, H. (1999). Representing transition systems by logic programs. In *Proceedings of the 5th International Conference on Logic Programming and Nonmonotonic Reasoning*, pages 92–106.

Lin, F. & You, J.-H. (2002). Abduction in logic programming: a new definition and an abductive procedure based on rewriting. *Artif. Intell.*, **140**(1-2), 175–205.

Maimon, R. & Browning, S. (2000). Diagrammatic notation and computational structure of gene networks. In *Proceedings of the Second International Conference on Systems Biology*.

Marek, V. & Truszczynski, M. (1999). Stable models and an alternative logic programming paradigm. *The Logic Programming Paradigm: a 25-year Perspective*, pages 375–398.

Matsuno, H., Doi, A., Nagasaki, M., & Miyano, S. (2000). Hybrid petri net representation of gene regulatory network. In *Pacific Symposium on Biocomputing*

2000 (PSB 2000), pages 341–52.

Meyers, S. & Friedland, P. (1984). Knowledge-based simulation of genetic regulation in bacteriophage lambda. *Nucleic Acids Research*, **12**(1), 1–9.

Michael, D. & Oren, M. (2002). The p53 and Mdm2 families in cancer. *Curr. Opin. Genet. Dev.*, **12**(1), 53–59.

Missiaen, M., Bruynooghe, L., & Denecker, M. (1995). CHICA: A planning system based on event calculus. *J. Logic Comput.*, **5**(5), 579–602.

Niemela, I. (1999). Logic programming with stable model semantics as a constraint programming paradigm. *Annals of Mathematics and Artificial Intelligence*, **25**(3,4), 241–273.

Oda, K., Kimura, T., Matsuoka, Y., Funahashi, A., Muramatsu, M., & Kitano, H. (2004). Molecular Interaction Map of a Macrophage. *AfCS Research Report.*

Ogata, H., Goto, S., Sato, K., Fujibuchi, W., Bono, H., & Kanehisa, M. (1999). Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Research*, **27**(1), 29–34.

Peirce, C. (1931-1958). *Collected papers of Charles Sanders Peirce, Vol. 1-8.* Havard University Press, Cambridge, MA.

Peirce, C. (1992). *Reasoning and the Logic of Things.* Havard University Press, Cambridge, MA.

Peleg, M., Yeh, I., & Altman, R. B. (2002). Modelling biological processes using workflow and petri net models. *Bioinformatics*, **18**(6), 825–837.

Poole, D. (1988). A logical framework for default reasoning. *Artif. Intell.*, **36**(1), 27–48.

Poole, D. (1989). Explanation and prediction: an architecture for default and abductive reasoning. *Computational Intelligence*, **5**(1), 97–110.

Poole, D. (1993). Probabilistic Horn abduction and Bayesian networks. *Artif. Intell.*, **64**(1), 81–129.

Poole, D., Goebel, R., & Aleliunas, R. (1987). Theorist: A logical reasoning system for default and diagnosis. *The Knowledge Frontier: Essays in the Representation of Knowledge*, pages 331–352.

Poole, D., Mackworth, A., & Goebel, R. (1998). *Computational Intelligence.* Oxford University Press, Oxford.

Priami, C., editor (2003). *Computational Methods in Systems Biology. LNCS*, volume 2602. Springer Verlag.

Regev, A., Silverman, W., & Shapiro, E. (2001). Representation and simulation of biochemical processes using π-calculus process algebra. In *Proc. PSB*, pages 459–470.

Reggia, R. (1983). Diagnostic expert system based on a set covering model. *Internat. J. Man Machine Studies*, **19**(5), 437–460.

Reiter, R. (1980). A theory of diagnosis from first principles. *Artif. Intell.*, **13**(1–2), 81–132.

Sambrano, G. R. (2003). Developing a navigation and visualization system for signaling pathways. *AfCS Reports. http://www.signaling-gateway.org/reports/v1/DA0009.pdf.*

Schoeberl, B., Eichler-Jonsson, C., Gilles, E., & Muller, G. (2002). Computational modeling of the dynamics of the map kinase cascade activated by surface and internalized egf receptors. *Nature Bitechnology*, **20**(4), 370–375.

Sembugamoorthy, V. & Chandrasekaran, B. (1986). Functional Representation of Devices and Compilation of Diagnostic Problem-Solving Systems. *Experience, Memory and Reasoning*, pages 47–73.

Shimada, T., Hagiya, M., Arita, M., Nishizaki, S., & Tan, C. (1995). Knowledge-based simulation of regulatory action in lambda phage. *Int. J. Artif. Intell. Tools*, **4**(4), 511 V524.

Shrager, J. & Langley, P. (1990). *Computational Models of Scientific Discovery and Theory Formation*. Morgan Kaufmann.

Talcott, C., Eker, S., Knapp, M., Lincoln, P., & Laderoute, K. (2004). Pathway logic modeling of protein functional domains in signal transduction. In *Proc. PSB*, pages 568–580.

Voit, E. O. (2000). *Computational Analysis of Biochemical Systems*. Cambridge University Press.

Zupan, B., Bratko, I., Demsar, J., Juvan, P., Curk, T., Borstnik, U., Beck, J. R., Halter, J., Kuspa, A., & Shaulsky, G. (2003). GenePath: a system for inference of genetic networks and proposal of genetic experiments. *Artif. Intell. Med.*, **29**(1-2), 107–30.

A Proofs

Lemma A.1 *Let $\mathcal{D}$ be a domain description and $\mathcal{O}$ be an initial state complete set of observations in $\mathcal{D}$. Then there exists a unique initial state that is consistent with $\mathcal{O}$.*

Proof. We prove the proposition by contradiction. Let s and s' be different initial states that is consistent with $\mathcal{O}$. Then either $s \setminus s' \neq \emptyset$ or $s \setminus s' \neq \emptyset$. Assume that $s \setminus s' \neq \emptyset$. Let f be a fluent literal in $s \setminus s'$. Since $f \in s$ and s is consistent with $\mathcal{O}$, the observation " **initially** f " is in $\mathcal{O}$. Since $f \in s'$ and s' is consistent with $\mathcal{O}$, the observation " **initially** $\neg f$ " is in $\mathcal{O}$. But both " **initially** f " and " **initially** $\neg f$ " cannot belong to $\mathcal{O}$. □

Lemma A.2 *Let $(\mathcal{D}, \mathcal{O})$ be an action theory where $\mathcal{O}$ is initial state complete. If $\sigma = \langle s_0, A_0, \ldots s_n, A_n \rangle$ is a trajectory model of $(\mathcal{D}, \mathcal{O})$ then*

$$A_t = (trig(s_t) \setminus inhi(s_t)) \cup \{a \mid (a \ \textbf{occurs\_at} \ t) \in \mathcal{O}\} \qquad \text{(A.1)}$$

for all $t \geq 0$.

Proof. Let a_1 belong to the set on the right-hand side of (A.1). If $a_1 \in trig(s_t) \setminus inhi(s_t)$, then $a_1 \in trig(s_t)$ and $a_1 \notin inhi(s_t)$. By the definition of trajectories, $a_1 \in A_t$. If $a_1 \in \{a \mid (a \ \textbf{occurs\_at} \ t) \in \mathcal{O}\}$, then $(a_1 \ \textbf{occurs\_at} \ t) \in \mathcal{O}$. Since σ is an interpretation of $(\mathcal{D}, \mathcal{O})$, $a_1 \in A_t$. Thus we have shown that:

$$(trig(s_t) \setminus inhi(s_t)) \cup \{a \mid (a \ \textbf{occurs\_at} \ t) \in \mathcal{O}\} \subseteq A_t \ .$$

Now let $a_2 \in A_t$. If a_2 is a triggered action, then $a_2 \in trig(s_t)$ and $a_2 \notin inhi(s_t)$, by the definition of trajectories. If a_2 is an exogenous action, then by the definition of trajectory models, $\mathcal{O}$ contains the observation " a_2 **occurs_at** t ". Thus we have shown that

$$A_t \subseteq (trig(s_t) \setminus inhi(s_t)) \cup \{a \mid (a \ \textbf{occurs\_at} \ t) \in \mathcal{O}\} \ .$$

Note that (A.1) also tells us how to construct the unique trajectory model of $(\mathcal{D}, \mathcal{O})$ if such trajectory model exists. □

We define a splitting of the translation $\pi(\mathcal{D}, \mathcal{O})$ of an action theory $(\mathcal{D}, \mathcal{O})$. The splitting shall be used in proving properties of the AnsProlog program $\pi(\mathcal{D}, \mathcal{O})$.

Definition 1 *Let $(\mathcal{D}, \mathcal{O})$ be an action theory. We define the partition of $\pi(\mathcal{D}, \mathcal{O})$ into sub-programs π_i^S, π_j^A, $0 \leq i \leq t_{max}$, $0 \leq j < t_{max}$ as follows.*

- *The sub-program π_0^S consists of the following rules:*
  - *For all observation " **initially** f " in $\mathcal{O}$:*

$$\pi(f, 0) \leftarrow \tag{A.2}$$

  - *For all fluent f:*

$$\bot \leftarrow \pi(f, 0), \pi(\neg f, 0) \ .$$

- *The sub-program π_t^A, $0 \leq t < t_{max}$, consists of the following rules:*
  - *For all trigger rules " $g_1, \ldots, g_m$ **triggers** b " of $\mathcal{D}$:*

$$\pi(b, t) \ \leftarrow \ \pi(g_1, t), \ \ldots \ , \ \pi(g_m, t), \ not \ holds(ab(occurs(b)), t) \ . \tag{A.3}$$

  - *For all inhibition rules " $h_1, \ldots, h_l$ **inhibits** c " of $\mathcal{D}$:*

$$holds(ab(occurs(c)), t) \ \leftarrow \ \pi(h_1, t), \ \ldots \ , \ \pi(h_l, t) \ . \tag{A.4}$$

  - *For all observations " a **occurs_at** t " in $\mathcal{O}$ where a is a triggered action:*

$$\bot \leftarrow \ not \ \pi(a, t) \ . \tag{A.5}$$

  - *For all observations " a **occurs_at** t " in $\mathcal{O}$ where a is a non-triggered action :*

$$\pi(a, t) \leftarrow \ . \tag{A.6}$$

- *The sub-program π_{t+1}^S, $0 \leq t < t_{max}$, consists of the following rules:*
  - *For all fluents f in $\mathcal{D}$:*

$$\pi(f, t+1) \leftarrow \pi(f, t), \ \ not \ \pi(\neg f, t+1) \ . \tag{A.7}$$
$$\pi(\neg f, t+1) \leftarrow \pi(\neg f, t), \ not \ \pi(f, t+1) \ . \tag{A.8}$$
$$\bot \leftarrow \pi(f, t+1), \pi(\neg f, t+1) \ . \tag{A.9}$$

· *For all causal rules " a* **causes** *f* **if** $f_1, \ldots, f_n$ *" of* $\mathcal{D}$*:*

$$\pi(f, t+1) \leftarrow \pi(a, t),\ \pi(f_1, t),\ \ldots\ ,\ \pi(f_n, t)\ . \tag{A.10}$$

· *For all observations of the form " f* **at** $(t+1)$ *" in* $\mathcal{O}$*:*

$$\bot \leftarrow \ not\ \pi(f, t+1)\ . \tag{A.11}$$

Lemma A.3 *Let* $(\mathcal{D}, \mathcal{O})$ *be an action theory. Let the sets* π_i^S*'s and* π_j^A*'s form the partition of the translation* $\pi(\mathcal{D}, \mathcal{O})$ *given by Definition 1. Then* $\pi(\mathcal{D}, \mathcal{O})$ *is splitted by the sequences of sets* $L_0, L_1, \ldots, L_{2*t_{max}}$ *defined as:*

$$\begin{aligned} L_0 &= lit(\pi_0^S) \\ L_1 &= L_0 \cup lit(\pi_0^A) \\ &\cdots\cdots \\ L_{2t} &= L_{2t-1} \cup lit(\pi_t^S) \\ L_{2t+1} &= L_{2t} \cup lit(\pi_t^A) \\ &\cdots\cdots \\ L_{2*t_{max}} &= L_{2*t_{max}-1} \cup lit(\pi_{t_{max}}^S) \end{aligned}$$

Moreover, the corresponding "bottom" b_{L_k} *layers are* $b_{L_0} = \pi_0^S$ *and for all* $0 \leq t < t_{max}$*:*

$$\begin{aligned} b_{L_{2t+1}} &= b_{L_{2t}} \cup \pi_t^A \\ b_{L_{2t+2}} &= b_{L_{2t+1}} \cup \pi_{t+1}^S \end{aligned}$$

Proof. It is true according to the definition of splitting set sequence. □

Lemma A.4 *Let* $(\mathcal{D}, \mathcal{O})$ *be an action theory where* $\mathcal{O}$ *is initial state complete. If* M *is an answer set of* $\pi(\mathcal{D}, \mathcal{O})$*, then the inverse translation* $\pi^{-1}(M)$ *is a trajectory model of* $(\mathcal{D}, \mathcal{O})$*.*

Proof. Let us consider the splitting set sequence of $(\mathcal{D}, \mathcal{O})$ in Lemma A.3. By the splitting set theorem, there exist sets M_{2i}'s and M_{2j+1} such that:

$$M = M_0^S \cup \bigcup_{t=1}^{t_{max}} [M_t^A \cup M_t^S]$$

and such that M_0^S is an answer set of π_0^S and for all $0 < t \leq t_{max}$:

- M_{t-1}^A is an answer set of the partial evaluation

$$E_{2t-1} = e_{L_{2t-2}}(b_{L_{2t-1}} \setminus b_{L_{2t-2}}, M_{2t-2}) = e_{L_{2t-2}}(\pi_{t-1}^A, M_{2t-2})$$

- M_t^S is an answer set of the partial evaluation

$$E_{2t} = e_{L_{2t-1}}(b_{L_{2t}} \setminus b_{L_{2t-1}}, M_{2t-1}) = e_{L_{2t-1}}(\pi_t^S, M_{2t-1})$$

where $M_0 = M_0^S$ and for all $0 < t \leq t_{max}$:

$$M_{2t-1} = M_{t-1}^A \cup M_{2t-2}$$
$$M_{2t} = M_t^S \cup M_{2t-1}$$

Let $\sigma = \pi^{-1}(M) = s_0\, A_1\, s_1\, A_2\, \ldots\, A_n\, s_n \ldots$. It is straightforward to verify that $s_t = \pi^{-1}(M_t^S)$ and $A_t = \pi^{-1}(M_t^A)$ for all t.

We now show that σ is a trajectory model, using Lemma A.2.

- Since M_0^S is an answer set of π_0^S:

  $$M_0^S = \{f \mid \{\pi(f,t) \leftarrow\} \subseteq \pi_0^S\} = \{f \mid (\textbf{initially}\ f) \in \mathcal{O}\}$$

  Because $\mathcal{O}$ is initial state complete, it follows that $s_0 = \pi^{-1}(M_0)$ is an initial state.
- Let $A_t = \pi^{-1}(M_t^A)$, we shall prove (A.1). We have that M_t^A is an answer set of the partial evaluation:

  $$E_{2t+1} = e_{L_{2t}}(\pi_t^A, M_{2t})$$

  By definition, E_{2t+1} consists of the following rules:
  · For all trigger rules " $g_1, \ldots, g_m$ **triggers** b " of $\mathcal{D}$ such that $\{\pi(g_1,t), \ldots, \pi(g_m,t)\} \subseteq M_{2t}$, or equivalently, such that $\{g_1, \ldots, g_m\} \subseteq s_t$:

  $$\pi(b,t) \leftarrow \quad not\ holds(ab(occurs(b)),t)\ .$$

  · For all inhibition rules " $h_1, \ldots, h_l$ **inhibits** c " of $\mathcal{D}$ such that $\{\pi(h_1,t), \ldots, \pi(h_l,t)\} \subseteq M_{2t}$, or equivalently, such that $\{h_1, \ldots, h_l\} \subseteq s_t\}$:

  $$holds(ab(occurs(c)),t) \leftarrow \quad .$$

  · For all observations " a **occurs_at** t " in $\mathcal{O}$ where a is a triggered action:

  $$\bot \leftarrow\ not\ \pi(a,t)\ .$$

  · For all observations " a **occurs_at** t " in $\mathcal{O}$ where a is a non-triggered action :

  $$\pi(a,t) \leftarrow \quad .$$

  By computing the reduct of E_{2t+1} with respect to M_t^A, it follows that M_t^A is the set of the following atoms:
  ⋆ all the atoms $\pi(a,t)$ such that a is a triggered action, a is triggered by s_t and a is not inhibited by s_t; and
  ⋆ all the atoms $\pi(a,t)$ such that a is a non-triggered action, " a **occurs_at** t " is in $\mathcal{O}$; and
  ⋆ all the atoms $holds(ab(occurs(a)),t)$ where a is a triggered action and a is inhibited by s_t.

Consequently, (A.1) holds. Also, we have that $a \in \pi^{-1}(M_t^A)$ for all ' a **occurs_at** t " in $\mathcal{O}$.

- Let assume that $s_t = \pi^{-1}(M_t^S)$ is a state. We shall prove that $s_{t+1} = \Phi_{\mathcal{D}}(A_t, s_t)$. By definition, we have that $s_{t+1} = \pi^{-1}(M_{t+1}^S)$. Recall that the set M_{t+1}^S is an answer set of the partial evaluation:

$$E_{2t+2} = e_{L_{2t+1}}(\pi_{t+1}^S, M_{2t})$$

Based on the definition of partial evaluation, it is straightforward to verified that E_{2t+2} consists of the following rules:

· For all fluents literal g such that $\pi(g,t) \in M_{2t}$:

$$\pi(g, t+1) \leftarrow \quad not\ \pi(\neg g, t+1) \tag{A.12}$$

· For all fluents f of $\mathcal{D}$:

$$\bot \leftarrow \pi(f, t+1), \pi(\neg f, t+1)\ . \tag{A.13}$$

· For all causal rules " a **causes** f **if** $f_1, \ldots, f_n$ " of $\mathcal{D}$ such that $\{\pi(a,t), \pi(f_1,t), \ldots, \pi(f_n,t)\} \subseteq M_{2t}$, or equivalently, such that $a \in A_t$ and $\{f_1, \ldots, f_n\} \subseteq s_t$:

$$\pi(f, t+1) \ \leftarrow \ . \tag{A.14}$$

· For all observations of the form " f **at** $t+1$ " in $\mathcal{O}$:

$$\bot \leftarrow \ not\ \pi(f, t+1)\ . \tag{A.15}$$

Let E'_{2t+2} be the sub-program of E_{2t+2} that contains the rules (A.12) and (A.14). It can be verified that the reduct of E'_{2t+2} with respect to M_{t+1}^S are the set of rules " $\alpha \leftarrow$ ", where α is an atom such that:

⋆ $\alpha = \pi(f,t)$ where $f \in E_{\mathcal{D}}(A_t, s_t)$; or

⋆ $\alpha = \pi(g, t+1)$, where $\pi(g,t) \in M_t^S$ and $\pi(\neg g, t+1) \notin E_{\mathcal{D}}(A_t, s_t)$; that is, where $g \in s_t \setminus \neg E_{\mathcal{D}}(A_t, s_t)$.

It follows that $s_{t+1} = \pi^{-1}(M_{t+1}^S) = \Phi_{\mathcal{D}}(A_t, s_t)$. Moreover, due to the integrity constraints of E_{2t+2}, $s_{t+1} = \pi^{-1}(M_{t+1}^S)$ is a state and $f \in s_{t+1}$ for all observation " f **at** $(t+1)$ " in $\mathcal{O}$.

□

Lemma A.5 *Let $(\mathcal{D}, \mathcal{O})$ be an action theory where $\mathcal{O}$ is initial state complete. If σ is a model of $(\mathcal{D}, \mathcal{O})$, then $\pi(\sigma)$ is an answer set of $\pi(\mathcal{D}, \mathcal{O})$.*

Proof. (*sketch*) We consider the same splitting structure of the program $\pi(\mathcal{D}, \mathcal{O})$ as in the proof of Lemma A.4. Let $\sigma = s_0\, A_1\, s_1\, A_2\, \ldots\, A_n\, s_n \ldots$. We define:

$$\pi(s_t) = \{\pi(f,i) \mid f \in s_t\}$$
$$\pi(A_t) = \{\pi(a), t) \mid a \in A_t\} \cup \{holds(ab(occurs(a)), t) \mid a \in \mathbf{A}_{trig} \setminus A_t\}$$

Given that σ is trajectory model, using Lemma A.2, we can show that $\pi(s_t)$ is an answer set of the partial evaluation E_{2t} and $\pi(A_t)$ is an answer set of the partial evaluation E_{2t+1}. It then follows from the splitting set theorem that $\pi(\sigma)$ is an answer set of $\pi(\mathcal{D}, \mathcal{O})$. □

Lemma A.6 *Let $(\mathcal{D}, \mathcal{O})$ be an action theory where $\mathcal{O}$ is initial state complete. If $(\mathcal{D}, \mathcal{O})$ is consistent then it has a unique trajectory model.*

Proof. Let $\sigma = \langle s_0, A_0, \ldots s_n, A_n \rangle$ be a trajectory model of $(\mathcal{D}, \mathcal{O})$. Since $A_{t+1} = \Phi_{\mathcal{D}}(A_t, s_t)$, the set A_{t+1} is uniquely determined by s_t and A_t. Because of Lemma A.2, it follows that σ is uniquely determined by s_0. By Proposition A.1, such a state s_0 is unique, so σ is the unique model of $(\mathcal{D}.\mathcal{O})$. □

Lemma A.7 *Let $(\mathcal{D}, \mathcal{O})$ be a consistent action theory, where $\mathcal{O}$ is initial state complete. Then the program $\pi(\mathcal{D}, \mathcal{O})$ has a unique answer set.*

Proof. Since $(\mathcal{D}, \mathcal{O})$ is consistent, it has a model σ. Then by Lemma A.5, $\pi(\sigma)$ is an answer set of $\pi(\mathcal{D}, \mathcal{O})$. If there exists another answer set M of $\pi(\mathcal{D}, \mathcal{O})$, then $\pi^{-1}(M)$ is a model of $(\mathcal{D}, \mathcal{O})$, by Lemma A.4. Since $M \neq \pi(\sigma)$, $\pi^{-1}(M) \neq \sigma$. It follows that $(\mathcal{D}, \mathcal{O})$ has more than one model, which contradicts to Proposition A.6. □

The following Lemma A.8 and Lemma A.9 are corollaries of Lemma A.6 and A.7, and Lemma A.4 and A.5.

Lemma A.8 *Let $(\mathcal{D}, \mathcal{O})$ be a consistent action theory where $\mathcal{O}$ is a initial state complete. If $\sigma = s_0\, A_1\, s_1\, A_2\, \ldots\, A_n\, s_n$ is the trajectory model of $(\mathcal{D}, \mathcal{O})$ and M is the answer set of $\pi(\mathcal{D}, \mathcal{O})$, then $M = \pi(\sigma)$.*

Lemma A.9 *Let $(\mathcal{D}, \mathcal{O})$ be a consistent action theory where $\mathcal{O}$ is a initial state complete. Then*

$$(\mathcal{D}, \mathcal{O}) \models f \textbf{ at } t \iff \pi(f, t) \in M$$
$$(\mathcal{D}, \mathcal{O}) \models a \textbf{ occurs\_at } t \iff \pi(a, t) \in M$$

where M is the unique answer set of $\pi(\mathcal{D}, \mathcal{O})$.

Lemma A.10 *Let $pred(\mathcal{D}, \mathcal{O}, Q)$ be a prediction problem, where the domain $\mathcal{D}$ is trigger bounded and Q is the query f **after** A_1 **at** t_1, $\ldots$ A_n **at** t_n , where $t_1 < t_2 \ldots < t_n$. Let $t_{max} = t_n + tbound(\mathcal{D})$; and let π_{pred} be the AnsProlog translation*

$$\pi(\mathcal{D}, \mathcal{O}) \cup \pi(\{\ A_1 \textbf{ occurs\_at } t_1, \ldots A_n \textbf{ occurs\_at } t_n\ \}).$$

Then the theory $(\mathcal{D}, \mathcal{O})$ predicts Q if and only if the program π_{pred} has at least one answer set; and for all the answer set S of the program π_{pred}, there exists $N \geq t_n$ such that $holds(f, k) \in S$ for every time point $k \in (N, t_{max}]$.

Proof. Let $\mathcal{O}' = \mathcal{O} \cup \{ A_1 \ \textbf{occurs\_at} \ t_1, \ldots A_n \ \textbf{occurs\_at} \ t_n \}$. Then $\pi_{pred} = \pi(\mathcal{D}, \mathcal{O}')$. Note that $\mathcal{O}'$ is initial state complete, since $\mathcal{O} \supseteq \mathcal{O}'$ and $\mathcal{O}$ is initial state complete.

($\Rightarrow$) Let us assume that $(\mathcal{D}, \mathcal{O}) \models Q$. We shall show that π_{pred} has a unique answer set which contains $\pi(f, t_{max})$.
Since $(\mathcal{D}, \mathcal{O}) \models Q$, the theory $(\mathcal{D}, \mathcal{O}')$ is consistent. We have noted that $\mathcal{O}'$ is also initial state complete, thus the theory $(\mathcal{D}, \mathcal{O}')$ has a unique trajectory model $\sigma = s_0 \, B_1 \, s_1 \, \ldots \, s_i \, B_i \, \ldots$. Besides, $t_{max} = t_n + tbound(\mathcal{D})$. It follows from the definition of $tbound(\mathcal{D})$ that $B_k = \emptyset$ and $s_k = s_{t_{max}}$, for all $k \geq t_{max}$. It is also followed from $(\mathcal{D}, \mathcal{O}) \models Q$ that there exists $N \geq t_n$ such that f is true in all the states s_k with $k \geq N$. Particularly, for some j greater than both N and t_{max}, we have that $s_j = s_{t_{max}}$ and $f \in s_j$. Then $f \in s_{t_{max}}$ and $(\mathcal{D}, \mathcal{O}') \models f$ **at** t_{max}.
By Lemma A.7, the program $\pi(\mathcal{D}, \mathcal{O}')$ has a unique answer set M. By Lemma A.8, $M = \pi(\sigma)$. We have shown that $(\mathcal{D}, \mathcal{O}') \models f$ **at** t_{max}. Taking into account Lemma A.9, we have that $\pi(f, t_{max}) \in M$.

($\Leftarrow$) Let us assume that $\pi(\mathcal{D}, \mathcal{O}') \models_{\mathcal{A}} \pi(f, t_{max})$. We shall prove that $(\mathcal{D}, \mathcal{O}) \models Q$.
Because $\pi(\mathcal{D}, \mathcal{O}') \models_{\mathcal{A}} \pi(f, t_{max})$, the program $\pi(\mathcal{D}, \mathcal{O}')$ has at least an answer set M. By the lemmas , the theory $(\mathcal{D}, \mathcal{O}')$ is consistent, $(\mathcal{D}, \mathcal{O}')$ has a unique trajectory model σ, M is the unique answer set of $\pi(\mathcal{D}, \mathcal{O}')$ and $M = \pi(\sigma)$.
Assume that $\sigma = s_0 \, B_1 \, s_1 \, \ldots \, s_i \, B_i \, \ldots$. Since $t_{max} = t_n + tbound(\mathcal{D})$, it follows from the definition of $tbound(\mathcal{D})$ that $B_k = \emptyset$ and $s_k = s_{t_{max}}$, for all $k \geq t_{max}$. Besides, it follows from $\pi(\mathcal{D}, \mathcal{O}') \models_{\mathcal{A}} \pi(f, t_{max})$ that $f \in s_{t_{max}}$. Then $f \in s_k$, for all $k \geq t_{max}$, so $(\mathcal{D}, \mathcal{O}) \models f$ by he definition of entailment. □

Proposition 4.1 *Let $\mathcal{P} = \langle D, O, Q, \mathcal{S} \rangle$ be a hypothesis formation problem. Let $trans(\mathcal{P}) = \langle ab(\mathcal{S}), \pi_f(Q), \pi(D) \cup \pi(O) \cup \pi_a(Q) \cup \pi(\mathcal{S}), \models^b \rangle$ be the transformation of $\mathcal{P}$ into abduction. Then $H \subseteq \mathcal{S}$ is a solution of $\mathcal{P}$ if and only if the set $\{picked(label(r)) \mid r \in H\}$ is a solution of the LPAP $trans(\mathcal{P})$.*

Proof. Let $H \subseteq \mathcal{S}$. Let $picked(H)$ be the program:

$$prog(H) = \{picked(label(r)) \leftarrow | \ r \in H\}$$

and let $picked(H)$ be the set of atoms:

$$picked(H) = \{picked(label(r)) \mid r \in H\}$$

Let $\Pi_H = prog(H) \cup \pi(D) \cup \pi(O) \cup \pi_a(Q) \cup \pi(\mathcal{S})$. Consider the splitting of Π_H by the set $ab(\mathcal{S})$ (see Appendix C). The bottom of Π_H relative to $ab(\mathcal{S})$ is $prog(H)$, whose unique answer set is $picked(H)$. It is straightforward to verify that the partial evaluation

$$e_{ab(\mathcal{S})}(\Pi_H \setminus prog(H), picked(H))$$

is the translation $\pi(D \cup H, O) \cup \pi_a(Q)$. Note that O is initial state complete, so the brave and cautious entailment by $\pi(D \cup H, O) \cup \pi_a(Q)$ coincide, thus Lemma A.10 is applicable. By Lemma A.10, $(D \cup H, O) \models Q$ if and only if $\pi(D \cup H, O) \cup \pi_a(Q) \models^b \pi_f(Q)$. Consequently, $(D \cup H, O) \models Q$ if and only if $\Pi_H \models^b \pi_f(Q)$; that is, if and only if $picked(H)$ is a solution of the LPAP $trans(\mathcal{P})$. □

Proposition 4.2 *Let $\mathcal{P} = \langle D, O, Q, \mathcal{S} \rangle$ is a hypothesis formation problem. Let $H \subseteq \mathcal{S}$ and $r \in \mathcal{S}$.*

- *Deciding if $Sol(\mathcal{P}) \neq \emptyset$ is NP-complete.*
- *Deciding if $r \in \mathcal{S}$ is relevant to $\mathcal{P}$ is Σ_2^P-complete.*
- *Deciding if $r \in \mathcal{S}$ is necessary to $\mathcal{P}$ is coNP-complete.*

Proof. The upper bounds of the complexities follow easily from Proposition 4.1. We shall now prove the lower bounds of the complexities.

Deciding if $Sol(\mathcal{P}) \neq \emptyset$ is NP-hard
Let Φ be a 3CNF formula of the variables $x_1, \ldots, x_m$. We present a transformation of Φ into an HFP $\mathcal{P}_\Phi = \langle D_\Phi, O_\Phi, Q_\Phi, \mathcal{S}_\Phi \rangle$ such that Φ is satisfiable if and only if $Sol(\mathcal{P}_\Phi) \neq \emptyset$.

Intuitively, the possible truth assignments of the variables $x_1, \ldots, x_m$ will correspond to candidate hypotheses for $\mathcal{P}_\Phi$. Thus we have an action $assign$, then define the hypothesis space $\mathcal{S}$ to be the set:

$$\mathcal{S} = \{assign \textbf{ causes } x_i \mid 1 \leq i \leq m\}$$

The domain description D_Φ is used for the evaluation of Φ given a set of truth assignment. It consists of the following rules:

- For each clause $C = y_1^c \vee y_2^c \vee y_3^c$ of Φ:

$$\begin{aligned} eval_{disj} &\textbf{ causes } holds(C) \textbf{ if } y_1^c \\ eval_{disj} &\textbf{ causes } holds(C) \textbf{ if } y_2^c \\ eval_{disj} &\textbf{ causes } holds(C) \textbf{ if } y_3^c \\ eval_{3CNF} &\textbf{ causes } \neg holds(\Phi) \textbf{ if } \neg holds(C) \end{aligned}$$

- Rules to trigger the actions $assign$, $eval_{disj}$ and $eval_{3CNF}$:

$$\begin{aligned} f_0 &\textbf{ triggers } assign \\ assign &\textbf{ causes } \neg f_0, f_1 \\ f_1 &\textbf{ triggers } eval_{disj} \\ eval_{disj} &\textbf{ causes } \neg f_1, f_2 \\ f_2 &\textbf{ triggers } eval_{3CNF} \\ eval_{3CNF} &\textbf{ causes } \neg f_2 \end{aligned}$$

The set O_Φ defines the initial state

$$O_\Phi = \{\neg x_i \mid 1 \leq i \leq m\} \cup \{\neg holds(C) \mid C \in \Phi\} \cup \{holds(\Phi)\} \cup \{f_0, \neg f_1, \neg f_2\}$$

Note that the variables are set to be *false* initially. A candidate hypothesis determines what variables will be set to *true*.

The query Q_Φ encodes that we want to find an assignment making Φ true:

$$Q_\Phi = holds(\Phi)$$

Note that for all subset $H \subseteq \mathcal{S}$, the action theory $(D \cup H, O)$ is consistent. The trajectory model of $(D \cup H, O)$ is of the form

$$\sigma = \langle s_0, assign, s_1, eval_{disj}, s_2, eval_{3CNF}, s_3 \rangle$$

where s_0 is the unique initial state corresponding to O. Furthermore, σ has the following properties:

- s_1 gives us a unique truth assignment for x_i, where $x_i = true$ if and only if " $assign$ **causes** x_i " is in H;
- For any clause $C \in \Phi$, C holds with respect to the truth assignment given by s_1 if and only if $holds(C) \in s_2$;
- The formula Φ is true with respect to the truth assignment given by s_1 if and only if $holds(\Phi) \in s_3$.

Thus Φ is satisfiable if and only if there exists $H \subset \mathcal{S}$ such that $(D \cup H, O) \models Q$.

Deciding if r is relevant to $\mathcal{P}$ is Σ_2^P-hard
Let Φ be a QBF formula:

$$\Phi = \exists x_1, \ldots, x_m \; \forall y_1, \ldots, y_n : \; \varphi(x_1, \ldots, x_m, y_1, \ldots, y_n)$$

where $\varphi(x_1, \ldots, x_m, y_1, \ldots, y_n)$ is a 3DNF formula of the variables $x_1, \ldots, x_m, y_1, \ldots, y_n$. We present a transformation of Φ into an HFP $\mathcal{P}_\Phi = \langle D_\Phi, O_\Phi, Q_\Phi, \mathcal{S}_\Phi \rangle$ such that Φ is valid if and only if a certain rule in $\mathcal{S}$ is relevant to $\mathcal{P}_\Phi$.

The transformation is built upon an intuition similar to that of the above case. There is an action *assign* to set the truth values of variables. We introduce an additional "dummy" variable ω. The hypothesis space $\mathcal{S}_\Phi$ is the set:

$$\begin{aligned}\mathcal{S} = &\{assign \textbf{ causes } x_i \mid i = 1, \ldots, m\} \cup \\ &\{assign \textbf{ causes } \hat{x}_i \mid i = 1, \ldots, m\} \cup \\ &\{assign \textbf{ causes } y_j \mid j = 1, \ldots, n\} \cup \\ &\{assign \textbf{ causes } \omega\}\end{aligned}$$

Let $r(z)$ denote the rule of the form " $assign$ **causes** z " of $\mathcal{S}$. Let $I(z)$ denote a truth assignment of the variables $x_1, \ldots, x_m, y_1, \ldots y_n$. Given a truth assignment I, let $H(I)$ be the subset of $\mathcal{S}$ such that:

$$\begin{aligned} H(I) =& \{assign \textbf{ causes } x_i \mid I(x_i) = true\} \cup \\ & \{assign \textbf{ causes } \hat{x}_i \mid I(x_i) = false\} \cup \\ & \{assign \textbf{ causes } y_j \mid I(y_j) = true\} \end{aligned}$$

We shall be interested in candidate hypotheses of the form $H(I)$ or $H(I) \cup \{r(\omega)\}$.

The domain description D_Φ contains actions $assign$, $eval_{conj}$, $eval_{DNF}$, $eval_\Phi$. Intuitively, $eval_{conj}$ evaluates the conjunctions of φ, $eval_{DNF}$ evaluates φ and $eval_\Phi$ evaluates Φ. The domain description consists of the following rules:

- For all conjunction $C = z_1^c \wedge z_2^c \wedge z_3^c$ of φ:

$$\begin{aligned} eval_{conj} \textbf{ causes } holds(C) \textbf{ if } & \ z_1^c, z_2^c, z_3^c \\ eval_{DNF} \textbf{ causes } holds(\varphi) \textbf{ if } & \ holds(C) \end{aligned}$$

- Rules to determine if Φ is valid:

$$\begin{aligned} eval_\Phi \textbf{ causes } \neg valid \textbf{ if } & \ \neg holds(\varphi), \neg\omega \\ eval_\Phi \textbf{ causes } \neg valid \textbf{ if } & \ y_1, \ldots, y_n, \omega \end{aligned}$$

- Rules to enforce the consistency of a candidate hypothesis, for all $1 \leq i \leq m$:

$$eval_\Phi \textbf{ causes } \neg consistent \textbf{ if } \ x_i, \hat{x}_i$$

- Rules to trigger the actions $assign$, $eval_{conj}$, $eval_{DNF}$, and $eval_\Phi$:

$$\begin{aligned} f_0 \textbf{ triggers } & assign \\ assign \textbf{ causes } & \neg f_0, f_1 \\ f_1 \textbf{ triggers } & eval_{conj} \\ eval_{conj} \textbf{ causes } & \neg f_1, f_2 \\ f_2 \textbf{ triggers } & eval_{DNF} \\ eval_{DNF} \textbf{ causes } & \neg f_2, f_3 \\ f_3 \textbf{ triggers } & eval_\Phi \\ eval_\Phi \textbf{ causes } & \neg f_3 \end{aligned}$$

The set O_Φ defines the initial state

$$\begin{aligned} O_\Phi =& \{\textbf{initially } \neg x_i, \hat{x}_i \mid 1 \leq i \leq m\} \cup \{\textbf{initially } \neg y_j \mid j = 1, \ldots, n\} \\ & \cup \{\textbf{initially } \neg holds(C) \mid C \in \Phi\} \\ & \cup \{\textbf{initially } \neg holds(\varphi), \neg\omega, consistent, valid, f_0, \neg f_1, \neg f_2, \neg f_3\} \end{aligned}$$

The query Q is that $Q = \{consistent, \neg valid\}$. We shall show that Φ is valid if and only if $r(\omega)$ is relevant to $\mathcal{P}_\Phi$.

Let us define a relation $\overset{x}{\sim}$ between the assignments such that $I_1 \overset{x}{\sim} I_2$ iff $I_1(x_i) = I_2(x_i)$, for all $1 \leq i \leq m$. Moreover, we call an assignment I *y-all-true* iff $I(y_j) = true$, for all $1 \leq j \leq n$. We can prove the following properties about $\mathcal{P}_\Phi$:

(1) The sets $H(I) \cup \{r(\omega)\}$, where I's are a *y-all-true* assignments, are candidate hypotheses.
(2) The sets $H(I)$, where I's are truth assignments that make φ *false*, are candidate hypotheses.
(3) All hypotheses of $\mathcal{P}_\Phi$ are of the form $H(I) \cup \{r(\omega)\}$ or of the form $H(I)$ described in 1. and 2. above.

Let I be a *y-all-true* assignment. If I' is an assignment such that $I' \overset{x}{\sim} I$ then $H(I') \subseteq H(I)$. It follows that $H(I) \cup \{r(\omega)\}$ is a hypothesis if and only if $H(I')$ is not a candidate hypothesis for all $I' \overset{x}{\sim} I$. Consequently, $H(I) \cup r(\omega)$ is a hypothesis if and only if I' makes φ *true* for all $I' \overset{x}{\sim} I$.

Let us assume that $r(\omega)$ is relevant to $\mathcal{P}_\Phi$. Then $r(\omega)$ belongs to some hypothesis. This hypothesis must be of the form $H(I) \cup \{r(\omega)\}$ with I being a *y-all-true* assignment. It follows that I' make φ *true* for all $I' \overset{x}{\sim} I$. Consequently, let $\Phi_{I(x)}$ be the formula

$$\Phi_{I(x)} = \forall y_1, \ldots, y_n : \varphi(I(x_1), \ldots, I(x_m), y_1, \ldots, y_n)$$

then $\Phi_{I(x)}$ is valid. Thus Φ is valid.

Now let us assume that Φ is valid. There exists an assignment $x_i := \epsilon_i$, $1 \leq i \leq m$, such that the formula

$$\Phi_{epsilon} = \forall y_1, \ldots, y_n : \varphi(\epsilon_1, \ldots, \epsilon_m, y_1, \ldots, y_n)$$

is valid. Let I be a *y-all-true* assignment such that $I(x_i) = \epsilon_i$, $\leq i \leq m$. Let I' be any assignment such that $I' \overset{x}{\sim} I$. Since $\Phi_{epsilon}$ is valid, I' makes φ *true*. Then $H(I) \cup \{r(\omega)\}$ is a hypothesis, so $r(\omega)$ is relevant to $\mathcal{P}_\Phi$.

Deciding if r is necessary to $\mathcal{P}$ is coNP-complete
Let r be a rule in $\mathcal{S}$. Define $\mathcal{P}_{\overline{r}} = \langle D, O, Q, \mathcal{S} \backslash \{r\} \rangle$. Consequently, r is necessary to $\mathcal{P}$ if and only if:

$$\begin{aligned} & r \in H, \text{ for all } H \in Sol(\mathcal{P}) \\ \Longleftrightarrow\ & r \in H, \text{ for all } H \text{ candidate hypothesis for } \mathcal{P} \\ \Longleftrightarrow\ & Sol(\mathcal{P}_{\overline{r}}) = \emptyset \end{aligned}$$

Since to decide the consistency of an HFP is NP-complete, to decide the relevancy of r to an HFP $\mathcal{P}$ is coNP-complete. □

B Logic Programming with Answer Sets (AnsProlog)

In declarative programming the intent of a program ('what') is described in a particular declarative programming language and the interpreter figures out the 'how' part. For example, a declarative program for sorting a set of numbers just describes what sorting is (and does not say how to sort a set of numbers), and the interpreter when given a set of numbers, figures out the 'how' part and does the sorting. Although this approach could be inefficient (in terms of run-time) for certain kind of programs, it has acceptable performance for query answering and reasoning programs. Its strong point is that it is much quicker and easier (and is often more robust) to write a declarative program than a procedural one. Declarative programming is a natural choice in database querying and for these reasons we will use it for the reasoning associated with our system. For example, SQL is a declarative language as in SQL we specify what we want, and not how the system should look and search the files to find the answer to our queries.

Among the various declarative and/or knowledge representation languages (most of which are declarative) that have been proposed one language that has probably the largest body of support structure (both theoretical results, efficient interpreters and developed applications) is the language of logic programming with the *answer set* (or *stable model*) semantics Gelfond & Lifschitz (1988) – simply referred to as AnsProlog in Baral (2003).

B.1 AnsProlog

The answer set semantics of logic programs – initially referred to as the stable model semantics – was proposed by Gelfond and Lifschitz in 1988 Gelfond & Lifschitz (1988). Unlike earlier characterizations of logic programs where the goal was to find a unique appropriate 'model' of a logic program, the answer set semantics allows the possibility that a logic program may have multiple appropriate models, or no appropriate models at all. Initially, some considered the existence of multiple or no stable models to be a drawback of stable model semantics, while others considered it to be a reflection of the poor quality of the program in question. Nevertheless, it is this feature of the answer set semantics Marek & Truszczynski (1999); Niemela (1999); Lifschitz & Turner (1999) that is key to the use of AnsProlog for problem solving. We now present the syntax and semantics of AnsProlog, which we will simply refer to as a logic

program.

A logic program Π is a set of rules of the form

$$a_0 \leftarrow a_1, \ldots, a_m, \ not\ a_{m+1}, \ldots, \ not\ a_n \tag{B.1}$$

or

$$\bot \leftarrow a_1, \ldots, a_m, \ not\ a_{m+1}, \ldots, \ not\ a_n \tag{B.2}$$

where $0 \leq m \leq n$, each a_i is an atom of a first-order language $\mathcal{LP}$, $\bot$ is a special symbol denoting the truth value *false*, and *not* is a connective called *negation-as-failure*. A negation as failure literal (or naf-literal) is of the form *not* a where a is an atom. For a rule of the form (B.1)-(B.2), the left and right hand side of the rule are called the *head* and the *body*, respectively. A rule of the form (B.2) is also called a constraint.

Given a logic program Π, we will assume that each rule in Π is replaced by the set of its ground instances so that all atoms in Π are ground. Consider a set of ground atoms X. The body of a rule of the form (B.1) or (B.2) is satisfied by X if $\{a_{m+1}, \ldots, a_n\} \cap X = \emptyset$ and $\{a_1, \ldots, a_m\} \subseteq X$. A rule of the form (B.1) is satisfied by X if either its body is not satisfied by X or $a_0 \in X$. A rule of the form (B.2) is satisfied by X if its body is not satisfied by X. An atom a is supported by X if a is the head of some rule of the form (B.1) whose body is satisfied by X.

For a set of ground atoms S and a program Π, the reduct of Π with respect to S, denoted by Π^S, is the program obtained from the set of all ground instances of Π by deleting

(1) each rule that has a naf-literal *not* a in its body with $a \in S$, and
(2) all naf-literals in the bodies of the remaining clauses.

S is an *answer set (or a stable model)* of Π if it satisfies the following conditions.

(1) If Π does not contain any naf-literal (i.e., $m = n$ in every rule of Π) then S is the smallest set of atoms that satisfies all the rules in Π.
(2) If the program Π does contain some naf-literal ($m < n$ in some rule of Π), then S is an answer set of Π if S is the answer set of Π^S. (Note that Π^S does not contain naf-literals, its answer set is defined in the first item.)

A program Π is said to be *consistent* if it has an answer set. Otherwise, it is inconsistent.

CR-Prolog Balduccini & Gelfond (2003); Balduccini (2005) is an extension of AnsProlog resulting from the introduction of consistency-restoring rules and preferences over them. CR-Prolog is an extension of AnsProlog with a new kind of rules called *consistency-restoring rules* (or *cr-rules*) and a *preference* relation $\ll$ defined over the set of cr-rules.

A cr-rule has the form:

$$r: \quad l_0 \stackrel{+}{\leftarrow} l_1, \ldots, l_n, \ not\ l_{m+1}, \ldots, \ not\ l_{m+n} \ .$$

Here, r is a label representing the name of such a rule. The regular form of the rule r, denoted $reg(r)$ is the standard AnsProlog rule:

$$l_0 \leftarrow l_1, \ldots, l_n, \ not\ l_{m+1}, \ldots, \ not\ l_{m+n} \ .$$

The regular forms of the cr-rules of a set X is also denoted $reg(X)$; that is:

$$reg(X) = \{reg(r) \mid r \in X\} \ .$$

Given a CR-Prolog program Π, a cr-rule r_1 is preferred to a cr-rule r_2, if π contains the expression $r_1 \ll r_2$. For any CR-Prolog program Π, let $ap(\Pi)$ denote the set of the standard AnsProlog rules of Π, $cr(\Pi)$ denote the set of the cr-rules of Π and $pref(\Pi)$ denote the set of the preferences of Π.

Definition B.1 *Let Π be a CR-Prolog program. Let M be a set of literals in Π and R be a subset of $cr(\Pi)$. The pair (M, R) is called a* view *of Π iff:*

- *M is an answer set of the AnsProlog program $ap(\Pi) \cup reg(R)$;*
- *there exists no rules $r_1 \neq r_2$, $\{r_1, r_2\} \subseteq R$ such that $r_1 \ll r_2$; and*
- *there exists no $R' \subset R$ such that M is an answer set of $ap(\Pi) \cup reg(R')$.* □

Example B.1 *Let Π be the CR-Prolog program:*

$$\Pi = \begin{cases} r_1: \quad p \stackrel{+}{\leftarrow} not\ r \\ r_1: \quad q \stackrel{+}{\leftarrow} not\ r \\ \leftarrow \ not\ p, \ not\ q \\ r_1 \ll r_2 \end{cases}$$

Then $(\{p\}, \{r_1\})$ are $(\{q\}, \{r_2\})$ are views of Π. The pair $(\{p, q\}, \{r_1, r_2\})$ is not a view of Π because of the preference $r_1 \ll r_2$.

Definition B.2 *Let (M_1, R_1) and (M_2, R_2) be view of a CR-Prolog program Π. The view (M_1, R_1)* dominates *the view (M_2, R_2) iff there exists $r_1 \in R_1$ and $r_2 \in R_2$ such that $r_1 \ll r_2$.* □

Definition B.3 *A view* (M, R) *is a* candidate answer set *of a CR-Prolog program* Π *if there exists no other view that dominates* (M, R). □

Definition B.4 *A set* M *of literal is an answer set of CR-Prolog program* Π *iff:*

- *there exists* $R \subset cr(\Pi)$ *such that* (M, R) *is a candidate answer set of* Π*; and*
- *there exists no candidate answer set* (M', R') *of* Π *such that* $R' \subset R$. □

Example B.2 *Let us continue with Example B.1. The view* $v_1 = (\{p\}, \{r_1\})$ *is a candidate answer sets of* Π. *The view* $v_2 = (\{q\}, \{r_2\})$ *is not a candidate answer set of* Π, *since* v_1 *dominates* v_2.

C Splitting Set Theorems

In this appendix, we review the basics of the Splitting Theorem Lifschitz & Turner (1994).

Let r be a rule

$$a_0 \leftarrow a_1, \ldots, a_m,\ not\ a_{m+1}, \ldots, a_n.$$

By $head(r)$, $body(r)$, and $lit(r)$ we denote a_0, $\{a_1, \ldots, a_n\}$, and $\{a_0, a_1, \ldots, a_n\}$, respectively. $pos(r)$ and $neg(r)$ denote the set $\{a_1, \ldots, a_m\}$ and $\{a_{m+1}, \ldots, a_n\}$, respectively.

For a program Π over the language $\mathcal{LP}$, a set of atoms of $\mathcal{LP}$, A, is a splitting set of Π if for every rule $r \in \Pi$, if $head(r) \in A$ then $lit(r) \subseteq A$.

Let A be a splitting set of Π. The *bottom of* Π *relative to* A, denoted by $b_A(\Pi)$, is the program consisting of all rules $r \in \Pi$ such that $lit(r) \subseteq A$.

Given a splitting set A for Π, and a set X of atoms from $lit(b_A(\Pi))$, the *partial evaluation of* Π *by* X *with respect to* A, denoted by $e_A(\Pi \setminus b_A(\Pi), X)$, is the program obtained from Π as follows. For each rule $r \in \Pi \setminus b_A(\Pi)$ such that

(1) $pos(r) \cap A \subseteq X$;
(2) $neg(r) \cap A$ is disjoint from X;

we create a rule r' in $e_A(\Pi, X)$ such that

(1) $head(r') = head(r)$, and
(2) $pos(r') = pos(r) \setminus A$,
(3) $neg(r') = neg(r) \setminus A$.

Let A be a splitting set of Π. A *solution to* Π *with respect to* A is a pair $\langle X, Y \rangle$

of sets of atoms satisfying the following two properties:

(1) X is an answer set of $b_A(\Pi)$; and
(2) Y is an answer set of $e_A(\Pi \setminus b_A(\Pi), X)$;

The splitting set theorem is as follows.

Theorem C.1 (Splitting Set Theorem, Lifschitz & Turner (1994)) *Let A be a splitting set for a program Π. A set S of atoms is a consistent answer set of Π iff $S = X \cup Y$ for some solution $\langle X, Y\rangle$ to Π with respect to A.* □

A *sequence* is a family whose index set is an initial segment of ordinals $\{\alpha \mid \alpha < \mu\}$. A sequence $\langle A_\alpha\rangle_{\alpha<\mu}$ of sets is *monotone* if $A_\alpha \subseteq A_\beta$ whenever $\alpha < \beta$, and *continuous* if, for each limit ordinal $\alpha < \mu$, $A_\alpha = \bigcup_{\gamma<\alpha} A_\gamma$.

A *splitting sequence* for a program Π is a nonempty, monotone, and continuous sequence $\langle A_\alpha\rangle_{\alpha<\mu}$ of splitting sets of Π such that $lit(\Pi) = \bigcup_{\alpha<\mu} A_\alpha$.

Let $\langle A_\alpha\rangle_{\alpha<\mu}$ be a splitting sequence of the program Π. A *solution to Π with respect to A* is a sequence $\langle E_\alpha\rangle_{\alpha<\mu}$ of set of atoms satisfying the following conditions.

(1) E_0 is an answer set of the program $b_{A_0}(\Pi)$;
(2) for any α such that $\alpha + 1 < \mu$, $E_{\alpha+1}$ is an answer set for $e_{A_\alpha}(b_{A_{\alpha+1}}(\Pi) \setminus b_{A_\alpha}(\Pi), \bigcup_{\gamma\leq\alpha} E_\gamma)$; and
(3) For any limit ordinal $\alpha < \mu$, $E_\alpha = \emptyset$.

The splitting set theorem is generalized for splitting sequence next.

Theorem C.2 (Splitting Sequence Theorem, Lifschitz & Turner (1994)) *Let $A = \langle A_\alpha\rangle_{\alpha<\mu}$ be a splitting sequence of the program Π. A set of atoms E is an answer set of Π iff $E = \bigcup_{\alpha<\mu} E_\alpha$ for some solution $\langle E_\alpha\rangle_{\alpha<\mu}$ to Π with respect to A.* □

To apply Theorems C.1-C.2 to programs with constraints of the form (B.2), we need to modify the notation of the bottom of a program relative to a set of atoms as follows.

Let $\Pi = \Pi_1 \cup \Pi_2$ be a program with constrains where Π_1 is a set of rules of the form (B.1) and Π_2 is a set of rules of the form (B.2). For a splitting set A of Π, we define $b_A(\Pi) = b_A(\Pi_1) \cup c_A(\Pi_2)$ where $c_A(\Pi_2) = \{r \mid r \in \Pi_2,\ lit(r) \subseteq A\}$.

We can prove that Theorems C.1-C.2 hold for programs with constraints. For example, if A is a splitting of the program Π, then S is an answer set of Π iff $A = X \cup Y$ where X is an answer set of $b_A(\Pi)$ and Y is an answer set of $e_A(\Pi \setminus b_A(\Pi), X)$. The proof of the modified theorems is based on two observations: (i) a set A of atoms from $lit(\Pi)$ is a splitting set of Π iff it is a

splitting set of Π_1 (because $\bot \notin lit(\Pi)$); and (ii) a set of atoms S is an answer set of Π iff S is answer set of Π_1 and S satisfies the rules of Π_2.