Abstract – We use techniques from Finite Model Theory to construct a framework for hypothesis creation and ranking to aid biologists with hypothesis evaluation and experimental design. Most bioinformatics research is geared toward pattern recognition and biological database management. Our work has somewhat different aims. First, we seek to determine the structure of the space of biological hypotheses that can be composed about a given system. Second, we seek to combine a wide variety of experimental data and literature sources for use in “proofreading” such hypotheses.

This data fusion problem has been a major stumbling block in modeling biological pathways. Consequently, most modeling frameworks make use of only one or two types of data, typically promoter sequences and microarray data. We present a modeling framework that is contradiction based and that performs data fusion on the logical level for an arbitrary number of sources. This greatly facilitates the incorporation of new data sources as they become available.

Once a new hypothesis has been constructed, data from existing experimental databases can be fused to rank the hypothesis based on corroborating and contradictory experimental evidence. We demonstrate the logical underpinnings of this process, and show how inflationary and deflationary logical extensions alter the process.

Keywords: logical data fusion, biological hypothesis evaluation

1 Introduction

The scientific method is a contradictive process. Hypotheses are proposed, and experiments are designed in order to attempt to contradict elements of the given hypotheses. On the other hand, mathematics is a fundamentally constructive process in which axioms are assumed and deductive rules are used to construct new statements or theorems that are known a posteriori to be valid.

Bioinformatics has heretofore taken the constructive view by defining algorithms for the construction of gene and signaling networks and the population of databases [1]. This approach yields hypothetical connections which can then be tested by experiment. This is essentially the opposite of what the experimental community is looking for, as they are in greater need of algorithms to test their own ideas than they are in need of new ideas to test.

The bottleneck is not at the idea-generation stage. Nor is it at the data-generation stage, for the mass of accumulated biological data is growing faster than ever with the advent of high-throughput experimental methods. The trouble arises at the data integration stage, when all of the new ideas must be tested against all of the new experimental results.

The frustrating process of algorithm design and validated data mining can be very slow. This leads to an inherent imbalance in the amount of data that can be analyzed and the amount of analysis that can be done.

The constructive approach to bioinformatics is also hindered by the fact that not all experimental data are correct. As a result, researchers using stored data are forced to try to distinguish between what is correct and what is incorrect in a context free environment, which is a problematical situation.

In this work, we advocate a new and important alternate view to bioinformatics work. We propose a contradiction based framework for data and hypothesis analysis, a subset of which is implemented in our prototype hypothesis proofreading software, HyBrow (Hypothesis-Space Browser).

HyBrow performs logical data fusion of four distinct data types on experimental data from the galactose metabolism system (GAL system) in yeast. In this paper, we present the logical framework underlying HyBrow and exhibit inflationary and deflationary extensions to that framework. The inflationary extensions will make composing hypotheses easier by making more abstract constructs available to the user, while our deflationary extensions will streamline hypothesis evaluation.

Our contradiction-based approach works well in the biological data fusion environment because it brings
the various types of biological experiments to bear in an organized manner. Suppose, for example, that we are examining a hypothetical gene activation relationship wherein the protein produced by ‘gene A’ is responsible for causing ‘gene B’ to begin transcription. If we have microarray timecourse data that indicates that the concentration of gene B’s mRNA transcript decreases after the concentration of gene A’s transcript increases, curational data indicating that gene A is most closely related to a gene that encodes for a structural protein, and no sequence data that indicates that gene A’s protein product binds to the promoter region of gene B, we can combine these information sources to discredit the hypothesized activation and provide a logical argument (with references back to the raw data) for our decision.

Please note that the data fusion process we outline in this paper takes place at the logical level, and thus the development presented here is not statistical in nature and in fact would sit atop any statistical hypothesis testing methodology.

We begin by assuming that an initial hypothesis $\eta_1$ is given for a specific biological process in an organism for which there exists a certain amount of experimental data.

Using the data, we decide what experimental evidence exists that contradicts the hypothesis $\eta_1$, what data exist that support (but not prove) $\eta_1$, or if there is insufficient experimental data to either support or contradict $\eta_1$ [2]. This last case is the most interesting case when experimental techniques exist that could actually make such a determination, allowing new experiments to be defined that further refine our knowledge of $\eta_1$.

2 Languages, Logic and Models

We now present an overview of the elementary results from Logic and Model Theory [3] we will need for this paper. Logic is divided into the study of syntax and the study of semantics. Syntax describes the ways in which symbols can be manipulated as formulas and terms. Semantics is the application of meanings to formulas and the study of formulas as they refer to structures.

2.1 Bounded Quantifier Logic

To describe observed or conjectured interactions in a biological system, we will use a logical language. A language $\mathcal{L}$ is a triple $(\mathcal{F}, \mathcal{R}, \mathcal{C})$ where:

1. $\mathcal{F}$ is a set of function symbols $f$, each with positive integer arity $n_f$,
2. $\mathcal{R}$ is a set of relation symbols $R$, each with non-negative integer arity $n_R$, and
3. $\mathcal{C}$ is a set of constant symbols $c$.

For us, the constants in the logical language represent the elemental constituents of biological reactions such as proteins, amino acids, simple sugars and nucleic acids. Functions are used to describe the interactions between these basic components that produce new structures or new contexts for existing structures. The set of all interactions is called the set of $L$-terms; it is the smallest set $T$ such that

1. $c \in T$, where $c$ is a constant of $L$,
2. $x \in T$, where $x$ is a variable in $\mathcal{V}$ and
3. if $f$ is an $n$-ary function and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$.

We assume a finite but arbitrarily large reserve of variables $\mathcal{V}$. Relationships between various terms are described by formulas. We consider only a special type of formula, namely the smallest set $\Phi$ such that:

1. $R(t_1, \ldots, t_n) \in \Phi$, where $t_1, \ldots, t_n$ are $L$-terms and $R \in \mathcal{R}$ is an $n$-ary relation symbol,
2. if $\varphi \in \Phi$, then $\neg \varphi \in \Phi$, and
3. if $\varphi, \psi \in \Phi$, then $\varphi \ast \psi \in \Phi$, where $\ast \in \\{\land, \lor, \to, \leftrightarrow\}$.

A formula with no variables is called a sentence. The reader familiar with logic will recognize that we are somewhat restricting ourselves, considering only
bounded quantifier formulas. We do so first to simplify our results and second because the use of universal quantifiers is extremely uncommon in biology, since it rarely happens that an assertion is true for every biological actor.

To complete our logical description of a biological system, we must describe the system itself. An L-structure or model is a tuple \( M = (M, \mathcal{F}^M, \mathcal{R}^M, \mathcal{C}^M) \), such that

1. \( M \) is a set called the universe,
2. \( \mathcal{F}^M \) is a set of functions \( f^M : M^{n_f} \to M \), with \( n_f \) the arity of the function symbol \( f \in L \).
3. \( \mathcal{R}^M \) is a set of relations \( R^M : R^M \subseteq M^{n_R} \), where \( n_R \) is the arity of \( R \in L \).
4. and \( \mathcal{C}^M \) is a set of constants in \( M \).

By \( |M| \), we mean \( |M| \). When \( |M| \) is finite, we call \( M \) a finite model. (Our use of finite models supports our use of bounded quantifiers.)

Models provide a way of assigning truth values to sentences. Let \( M \) be a model of a language \( L \). We say that an \( L \)-sentence \( \varphi \) is true in \( M \) if when we replace each function, relation and constant in \( \varphi \) with its corresponding function, relation or constant in \( M \), then the sentence is true in \( M \). Otherwise, the sentence is false. (This is the Tarskian definition of truth [3].)

### 2.2 Theories, Models and Discoverability

We first define a system-specific language (\( L_1 \)) for describing biological properties and interactions. (Please refer to www.HyBrow.org for an example language for describing metabolic regulation in yeast.)

We say that a model \( M \) satisfies a sentence \( \varphi \) if \( \varphi \) is true in \( M \). In this case we write \( M \models \varphi \). Similarly, a sentence \( \varphi \) is satisfiable if there is a model \( M \) such that \( M \models \varphi \).

Suppose that \( \Phi \) is a set of sentences. If, for every model \( M \) where \( M \models \Phi \), then \( M \models \varphi \), we write \( \Phi \vdash \varphi \). Given a set of sentences \( \Phi \), the set of sentences deduced by \( \Phi \) is denoted \( \text{Th}(\Phi) \).

In order to construct models of biochemical events, we suppose a language \( L_1^W \subseteq L_1 \) for biochemical composition. We call the language \( L_1^W \), the working language for a biologist. Biologists formulate hypotheses in this language. We further require that every \( L_1 \) model satisfy a certain set of constraints, which are enforced by rules (statements taken to be true without proof). These statements express fundamental “common sense” truths known in the biological domain, and will be referred to as \( \mathcal{R}_1 \). Note that the language used by biologists to formulate hypotheses may be smaller than the language used to formulate common sense rules. In practice, this allows fusion software to isolate the biologist from an overwhelmingly large theoretical framework (and simplifies graphical user interface programming). A model of a biological reaction is any finite model of \( L_1^W \) satisfying \( \mathcal{R}_1 \). (Note that the \( L_1^W \) model may in fact be an \( L_1 \) model.) We will call these finite models events.

### Example 2.1

Consider the following simple hypothesis about gene regulation in the GAL system in yeast. In English:

In the wild type organism, galactose activates Gal3p in the cytosol. Gal3p binds to the promoter of gal1 (in the nucleus), and then induces production of gal1’s transcript.

This hypothesis can be represented using the following three events from the HyBrow event language:

1. \( \text{ev1} = \text{galactose activate Gal3p in wt in cyt} \)
2. \( \text{ev2} = \text{Gal3p Binds to promoter gal1 in wt in nuc} \)
3. \( \text{ev3} = \text{Gal3p induce gal1 in presence of galactose in wt in nuc} \)

Here, “activates,” “binds to promoter,” and “induces” are the (first-order) relations, and “galactose,” “Gal3p,” and “gal1” are the (first-order) constants.

### 2.3 Interval Logic for Biological Reaction

Reactions in a biological system happen in time and space. Following biological conventions, we describe spatial dynamics using cellular compartments and cellular transport mechanisms. Following the example of the Monadic Second Order Theory (MSO) for describing automata, we will build temporal models of reaction systems out of individual reaction models. In this case, the analogy comes from the fact that language models in MSO are built up out of word models [4].

Let \( \mathcal{H}_1, \ldots, \mathcal{H}_k \) be \( k \) events. To contradict statements about causality in models of these reactions, we appeal to a simple temporal logic to relate the models.

We use five binary temporal relations, which are sufficient to describe all necessary temporal relations. Let \( \mathcal{H}_1 \) and \( \mathcal{H}_2 \) be two events (finite first order models of \( L_1^W \)).

1. \( A(\mathcal{H}_1, \mathcal{H}_2) \): The start of the reaction \( \mathcal{H}_1 \) precedes the start of the reaction \( \mathcal{H}_2 \).
2. \( B(\mathcal{H}_1, \mathcal{H}_2) \): The end of the reaction \( \mathcal{H}_1 \) precedes the start of the reaction \( \mathcal{H}_2 \).
3. \( C(\mathcal{H}_1, \mathcal{H}_2) \): The end of the reaction \( \mathcal{H}_1 \) precedes the end of the reaction \( \mathcal{H}_2 \).
4. \( D(\mathcal{H}_1, \mathcal{H}_2) \): The completion of the reaction \( \mathcal{H}_1 \) follows the start of the reaction \( \mathcal{H}_2 \).
5. \( E(\mathcal{H}_1, \mathcal{H}_2) \): The completion and start times of the two reactions are unrelated.

Using \( A, B, C, D, \) and \( E \) we can describe any temporal relationship between two reactions. (We will illustrate one case to demonstrate the simplicity of the fact.) We show that we can describe the case when reaction \( \mathcal{H}_2 \) occurs completely during the time reaction \( \mathcal{H}_1 \) occurs. Figure 3 shows this case. Then we write \( A(\mathcal{H}_1, \mathcal{H}_2) \land D(\mathcal{H}_2, \mathcal{H}_1) \). To prove this in general, suppose that reaction \( \mathcal{H}_1 \) begins and ends at times \( x_1 \) and \( x_2 \) respectively and that reaction \( \mathcal{H}_2 \) begins and ends at times \( x_3 \) and \( x_4 \) respectively. Then \( A(\mathcal{H}_1, \mathcal{H}_2) \) states \( x_1 < x_3 \); \( B(\mathcal{H}_1, \mathcal{H}_2) \) states \( x_1 < x_4 \); \( C(\mathcal{H}_1, \mathcal{H}_2) \) states \( x_2 < x_3 \); \( D(\mathcal{H}_1, \mathcal{H}_2) \) states \( x_2 < x_4 \). Finally,
3 Using the Logic

In this section, we lay out concrete steps for using the machinery defined in the previous section. These are the steps performed by our Hypothesis-Space Browser (HyBrow) software package. The software allows a biologist to input logical sentences in a fragment of $L^w_1$ using either a structured-text representation or a common graphical language familiar to biologists. The following steps are performed to evaluate $L^w_1$ and $L^w_2$ formulas.

1. Define a class of finite events by specifying first order hypotheses $\eta_1, \ldots, \eta_k$ in $L^w_1$. Each hypothesis $\eta_j$ generates a set of event models $H^*_1, \ldots, H^*_m$, such that $H^*_i \models \eta_j \cup R_1$.

2. Using the events defined in Step 1, specify a second order hypothesis $\tau$ in $L^w_2$. This hypothesis generates a set of event set models $T_1, \ldots, T_m$ such that $T_k \models \eta_2 \cup R_2$.

3. Given data $D$, if there exists $\varphi \in Th(D)$ such that for some $i$, $H^*_j \models \neg \varphi$ for all $j < n_i$, then $\eta_k$ is contradicted. Alternatively, if $T_l \models \neg \varphi$ for all $l < m$, then $\eta_2$ is contradicted.

The data $D$ can be obtained from local experiments or can be obtained by searching publicly available databases such as BIND [5], the Saccharomyces Genome Database (SGD) [6], and the Yeast Proteome Database (YPD) [7], (in the case of yeast biology, for which the HyBrow prototype was deployed) or other publicly accessible databases. The logic itself becomes the vehicle for data fusion, since multiple data sources each contain data points, which in turn produce finite data models that are used to derive contradictions to working hypotheses.

In the HyBrow prototype, four different data types are fused in this way. HyBrow refers to protein binding data, sequence data, microarray data, and literature data to make its judgments about GAL system hypotheses. These data types are stored in a local MySQL database and queried at need. (See www.HyBrow.org for details.)

We have demonstrated [2] the ability of the HyBrow prototype to evaluate biologist-generated hypotheses about gene regulation in yeast. We will next describe inflationary and deflationary logic extensions, to be incorporated into future generations of the HyBrow software, that will enable us to improve upon the performance of the prototype system.

4 Inflationary and Deflationary Logic and Expressiveness

The initial logical model of HyBrow presented in [2, 8] assumes an underlying set of systems models of biological reactions. We assume that each model satisfies the rule sets $R_1$ and $R_2$ and furthermore must not contradict existing data sets. These models are many-sorted and contain all currently available knowledge of biological interactions. Unlike data models, we
Figure 4: This is what the evaluation summary of a small set of related hypotheses looks like in HyBrow. This figure shows a “slice” of the hypothesis space sorted by the level of contraction or support each hypothesis was judged to possess. Shown are five seed hypotheses, together with their HyBrow-generated variants. The best hypothesis of each series is indicated by a “b” label.

assume that these models may contain as yet undiscovered knowledge.

Our approach is distinct from other approaches in that we do not attempt to build these models up from amassed data, but instead attempt to whittle them down by contradiction. This results in mixed inflationary and deflationary behavior as new data affects the system by contradicting models. The following results are proved in [8] and describe the impact of the hypothesis validation method upon system models.

Let $\mathcal{L}$ be a logic and let $\mathcal{L}$ be a language describing a biological system. Biological data can only offer evidence to support or disprove hypotheses to a certain level of confidence. Hence, if $\mathcal{M}$ is a model of $\mathcal{L}$ in the logic $\mathcal{L}$, then $\mathcal{M} = \text{Th}(\mathcal{M}) \subset \text{Th}(\mathcal{R}_1 \cup \mathcal{R}_2)$ is the set of all hypotheses true in the model $\mathcal{M}$. Simply put, $\mathcal{M}$ is a “view of the world” and $\mathcal{M}$ is the set of all things true in the world. If $\psi$ is a sentence in the language $\mathcal{L}$ that is disproved by an experiment, then this results give rise to a new theory $T^* \subseteq T$, where

$$\psi \in T^* \iff \psi \in T \cap \text{Th}(\mathcal{R}_1 \cup \mathcal{R}_2) \land \psi \not\vdash \varphi.$$  (1)

That is to say that $\psi$ does not imply the incorrect statement $\varphi$ and $\psi$ is not implied by the truth of $\varphi$.

Let $F : \mathcal{P}(T) \rightarrow \mathcal{P}(T)$. We shall say that $F$ is deflationary if for all $S \subseteq T$, $F(S) \subseteq S$. A fixed point of $F$ is a set $T_0$ such that $F(T_0) = T_0$. We shall say that $F$ is trivial if only $\emptyset$ is a fixed point of $F$. We can create a sequence:

$$T_0, T_1, \ldots, T_{\infty},$$

where $T_0 = \text{Th}(\mathcal{R}_1 \cup \mathcal{R}_2)$, $T_1 = F(T_0)$ etc. If we restrict our attention to only finite models of theories, then we can relate the deflationary operator $F$ to an operator on the models. Suppose that $\mathcal{M}$ is a finite model of a theory $T$. Then $\text{Th}(\mathcal{M})$ is finite by necessity. Hence, the application of $F$ to $\text{Th}(\mathcal{M})$ will lead to a new theory $T^* \subseteq \text{Th}(\mathcal{M})$.

**Theorem 4.1.** If $F$ is a deflationary operator and $\mathcal{M}$ is a finite model of a theory $T$, then there is a finite model $\mathcal{M}^*$ such that $\mathcal{M}^* \models T^*$ and $|\mathcal{M}^*| = |\mathcal{M}|$ and $\mathcal{M} = \mathcal{M}^*$.

**Proof.** Without loss of generality, suppose that $T^*$ contains one less sentence, $\varphi$, than $T$. If $\varphi$ is atomic, then it has the form $R(a_1, \ldots, a_k)$ and hence we can construct a model $\mathcal{M}^*$ from $\mathcal{M}$ by removing elements $a_1, \ldots, a_k$ from $R^\mathcal{M}$ to form $\mathcal{M}^*$. Suppose that $\varphi = \neg R(a_1, \ldots, a_k)$. Then we simply add $a_1, \ldots, a_k$ to $R^\mathcal{M}$ to form $\mathcal{M}^*$. Now suppose that $\varphi$ is conjunctive. Then it is clear how to construct $\mathcal{M}^*$. The result then follows by formula induction. $\square$

**Remark 4.2.** We see from the proof that if $\varphi$ is a positive atomic formula, then the relations of $\mathcal{M}$ are smaller than the relations of $\mathcal{M}^*$. Conversely, if $\varphi$ is not positive, then the relations of $\mathcal{M}^*$ are larger than those of $\mathcal{M}$. Conjunctions of positive and negative formula will inflate or deflate relations accordingly.

Using this approach, we can define inflationary and deflationary operators $F_1^\mathcal{M}$ and $F_2^\mathcal{M}$ on $\mathcal{P}(\mathcal{M})$. Let $T$, $\mathcal{M}$ be as above. Let us assume that $F$ removes exactly $H = \{ \psi : \psi \vdash \varphi \}$ for some special $\varphi$. Since we are considering only bounded quantifier predicate formulae, we can assume that $H$ is composed of atomic formula and their negations since removing a formula of the form $\alpha \land \beta$ is identical to removing both $\alpha$ and $\beta$. Trivially, $H$ is countable and can be separated into the disjoint union $H^+$, the positive formulae and $H^-$ the negative formulae. Let $F_1^\mathcal{M}$ map subsets of $\mathcal{M}$ to subsets of $\mathcal{M}$ corresponding to the removal of formula in $H^-$. Likewise, let $F_2^\mathcal{M}$ be defined for $H^+$. In this way, alterations to $\mathcal{M}$ correspond to mixed inflationary and deflationary actions.

**Theorem 4.3.** Suppose that $L$ is a language and $L^* = L \cup \{c_1, \ldots, c_n\}$. Suppose $\mathcal{M}$ is an $L$-structure and $\varphi$ is an $L$-sentence with unknown place holders $u_1, \ldots, u_n$. If $\mathcal{M} \models \varphi$, then $\mathcal{M}$ is also an $L^*$ structure and $\mathcal{M} \models \varphi^*$, when we replace $u_1, \ldots, u_n$ with $c_1, \ldots, c_n$.

**Proof.** Simply let the interpretation of $c_1, \ldots, c_n$ be the original interpretation of $u_1, \ldots, u_n$. $\square$

Thus we can extend $\mathcal{L}$ by adding constants as we determine new biological elements.

**Definition 4.4.** Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be logics. Then, $\mathcal{L}_1 \subseteq \mathcal{L}_2$ (read: $\mathcal{L}_1$ is at most as expressive as $\mathcal{L}_2$) if for every $\tau$ and every sentence $\varphi \in \mathcal{L}_1[\tau]$, there is a sentence $\phi \in \mathcal{L}_2[\tau]$ such that $\text{Mod}(\varphi) = \text{Mod}(\phi)^1$.

**Theorem 4.5.** If $F_1$ and $F_2$ are two deflationary operators on $T$, then $F_1 \circ F_2(T) = F_2 \circ F_1(T)$.

**Proof.** Without loss of generality, suppose $F_1$ removes $\psi_1$ and $F_2$ removes $\psi_2$. Apply Equation 1 to see that the two operations are interchangeable. $\square$

---

1 $\text{Mod}(\phi)$ is the set of all models satisfying $\phi$. 
If we assume the consistency of the initial theory \( T \) and \( \mathcal{M} \) is a model of \( T \), then \( F^\infty(T) \) yields a theory and a corresponding unique model \( \mathcal{M}^* \), that \( \mathcal{M}^* \models F^\infty(T) \) and the relations of \( T \) are obtained by the inflation and deflation of relations in \( \mathcal{M} \). This logic is like an inflationary logic, but incorporates both inflation and deflation of the relations.

Unfortunately, the expressive power of this logic depends entirely on the deflationary operator \( F \) and the resulting theory \( F^\infty(T) \), hence it cannot be determined without a complete knowledge of \( \mathcal{M}^* \). Let IDFP be the inflationary-deflationary fixed point logic generated from the quantifier free logic QF we have defined. Thus we deduce:

**Theorem 4.6.** If every relation of \( \mathcal{M}^* \) is larger than or equal to the relations of \( \mathcal{M} \), then \( \text{QF(IDFP)} \geq \text{QF} \). Conversely, if every relation of \( \mathcal{M}^* \) is smaller than or equal to the relations of \( \mathcal{M} \), then \( \text{QF(IDFP)} \leq \text{QF} \). Otherwise, the two logics are incomparable.

### 4.1 Inflationary Operators for Functional Abstraction

The system of inflation and deflation defined in the previous section is not classic in the sense of Finite Model Theory. However, it has led us to investigate the uses of traditional inflation for hypothesis input. Finite Model Theory often uses the concept of fixed point inflation to extend languages. We have found similar concepts to be of use in defining biological hypotheses. Ebbinghaus and Flum [4] give a full description of inflation. We summarize this below.

Let \( \mathcal{M} \) be a finite model. A mapping \( F : \mathcal{P}(M) \rightarrow \mathcal{P}(M) \) is said to be inflationary if \( F(Z) \supseteq Z \) for all \( Z \subseteq M \). If we begin with the empty set, we can form a sequence where \( F_0 = \emptyset \) and \( F_{n+1} = F(F_n) \). If there is some \( n \) such that \( F_{n+1} = F_n \), then \( F \) is said to have a fixed point and we call this fixed point \( F^\infty \).

Let \( \varphi \) be a formula with variables \( x_1, \ldots, x_k \) (and possibly parameters) and let \( X \) be an (unnamed) relation symbol that occurs positively in \( \varphi \). We have \( \varphi(x_1, \ldots, x_k, X) \). Let \( \mathcal{M} \) be a model. Then for some relation \( R \), we define \( F^\varphi(R) = \{ a_1, \ldots, a_k \in M : \mathcal{M} \models \varphi(a_1, \ldots, a_k, R) \} \).

Note, \( R(a_1, \ldots, a_k) \) simply means \( a_1, \ldots, a_k \in R \). From this, we can define a new logical operation, namely \( \text{Inf}_{X(\varphi)}(t) \), where \( t = t_1, \ldots, t_k \) are terms (possibly with parameters from a model). Then we have the following semantics:

\[
\mathcal{M} \models \text{Inf}_{X(\varphi)}\bigl(\langle t_1, \ldots, t_k \rangle\bigr) \quad \iff \quad t^\mathcal{M} \in F^\infty_{X(\varphi) \lor \varphi}
\]

Inflation is useful for allowing biologists to speak of functional iteration is vague terms. Consider the case when \( \varphi(x, y, X) = (f(x), y) \in X \). Then \( \text{Inf}_{X(\varphi)}\bigl(\langle t \rangle\bigr) \) is the set of pairs \( (x, y) \in X \), where \( x = f^k(y) \), for some number \( k \). In this case, the relation \( \text{Inf}_{X(\varphi)}\bigl(\langle t \rangle\bigr) \) is well defined just in case \( f \) has a fixed point. This construction can be surprisingly useful for certain biological operations where the user (biologist) is attempting to test generic hypotheses involving a functional changes in chemical reactants.

**Example 4.7.** In the HyBrow event language, “phosphorylate” is a first-order function. A user wishing to explore the possible activities of all elements of a MAP-kinase cascade may thus choose to inflate upon phosphorylate() rather than specify each of his or her hypothesized reactions for each possible state.

### 4.2 Deflation to the Testing Level

Though short hand notations (using for example, extension by definition or inflation) are very useful they can wreak havoc on attempts to test hypotheses against existing data models. For this reason, we restrict the use of inflationary operations and short hand notations to those that can be deflated by existing information about the chemistry of the hypotheses being investigated. By deflation, we mean that an inflationary operation \( \text{Inf}_{X(\varphi)}\bigl(\langle t \rangle\bigr) \) can be unwrapped into a finite series of disjunctions given knowledge of fixed points of the function symbols appearing in \( \varphi \). Similarly, we will only allow short hand representations of conjunctions and disjunctions if they can be deconstructed into their base elements; i.e., we can construct a parse tree of the elements, where at each level of the parse tree we remove one short hand symbol from the language and replace it with its corresponding atomic constituents. In this case, the resulting languages are deflated back to the original \( \mathcal{L}_1 \) or \( \mathcal{L}_2 \).

Put more formally, we only allow inflationary operations and extension by definition on \( \mathcal{L}_1, \mathcal{L}_2 \) if the resulting languages are not truly more expressive than the original languages.

**Example 4.8.** In this example, we illustrate the difficulty of combining languages of different expressive power.

Suppose we have the event: “Gal2p transports galactose through the membrane in wild-type \( S.~cerevisiae \).” In the HyBrow framework this would be represented as:

- Subject: Gal2p
- Object: galactose (extracellular)
- Verb: transports
- Organism context: wild-type \( S.~cerevisiae \)
- Physical context: membrane

To tell if this event happens, the result, galactose in the cytosol, is derived by applying the rule from the logic for the verb transports.

An independent research effort (the Reactome project) is attempting to compile an exhaustive list of curated metabolic pathways. Reactome uses an “input/output” framework (see Reactome.org) to store information.

We can translate the above sample hypothesis into the Reactome framework as follows:
Note that the translation to Reactome form takes the statement from an action-based representation to an entity-based representation and the verb transports is lost.

Consider the following Reactome event:
- Inputs: alpha-D-glucose (extracellular)
- Catalyst: GLUT3 tetramer (plasma membrane)
- Outputs: alpha-D-glucose (cytosol)
- Taxon: Homo sapiens
- Cellular compartment: membrane

In the HyBrow framework, this event will translate to:
- Subject: GLUT3 tetramer (plasma membrane)
- Object: alpha-D-glucose (extracellular)
- Verb: NONE
- Perturbation context: Homo sapiens
- Physical context: plasma membrane

HyBrow’s subject is usually the ‘catalyst’ and object is the ‘input’. (In a binding reaction the subject and object are both inputs.) There is, in effect, a verb in each Reactome event, which might translate roughly to “becomes” or “makes”. This is, essentially, the only verb in the Reactome vocabulary. There are no explicit types of verbs (i.e., no event ontology) for classifying events such as a phosphorylation event or binding event. However, Reactome events can always be expressed as HyBrow events. Therefore, all relations expressed in Reactome will always be expressible in the HyBrow framework allowing us to use Reactome as a data source for HyBrow, but not the reverse.

Thus it is not wise to inflate from Reactome’s language to HyBrow’s language, as testability with respect to the Reactome data would be unavailable.

5 Future Directions

Since our goal is to simplify and automate a significant portion of the work done by the biologist, we see several natural extensions and future directions for our work. First, we will draw together the data models presented in Section 2.4 with the events and event sets, as it seems quite reasonable to associate data points and hypotheses contradicted or supported by these data points. This leads to a new type of many sorted model called a result model. The universe of a result model consists of two sets: \( D \) a finite set of data points and \( H \) a finite set of events or event sets. There are two binary relations in a results model, \( S \) and \( C \). Each relation is contained in the set \( D \times H \). Elements in \( S \) are hypotheses supported by data points, while elements in \( C \) are hypotheses contradicted by data points.

Using this type of model, we can analyze the logical structure of meta-hypotheses, i.e., hypotheses of the type “suppose data point \( d \) were found, then that would contradict or support the following hypotheses.”

Using the rule sets \( R_1 \) and \( R_2 \) along with the existing system models presented in Section 4 we can identify hypotheses and data points that will have the most impact by determining the number of relations or models they will affect if they are contradicted. Such an analysis could lead to a better characterization of the types of experiments that need to be performed to lead to the most dramatic or important new discoveries.

We can then associate standardized experimental procedures that can be carried out by researchers or laboratory robots from, for instance, the Current Protocols in Molecular Biology or from lab-specific protocols containing instructions for standard experimental procedures. Such an association will allow maximally effective task lists to be identified.

Such a system allows new procedures for experimentation to be produced and validated quickly, generating data to contradict or support those hypotheses deemed of greatest interest. Using the established results of experimental design will allow us to recycle tested experimental designs within such a system. If costs and benefits can be associated to each experimental procedure, an optimal experimentation approach can be determined using a dynamic programming method such as Markov Decision Problems [9]. Through such techniques, the potential exists to radically streamline the way biological research is done. Common ontologies, shared hypothesis libraries, and common protocol updates are important first steps to this future work. All are facilitated by our software.

6 Conclusion

In this paper we have presented a logical framework for data fusion for biological systems. Our approach addresses the severe data integration problem experienced by bioinformatics researchers by using first and second order model theoretic formalisms and a contradiction based framework. This approach, to the best of our knowledge, is unique within the bioinformatics community. Our contradiction based framework allows us to perform fusion at the logical level, using each piece of data to rule out some portion of the space of possibly hypotheses.

We have demonstrated our approach using a series of examples from our prototype hypothesis evaluation program, HyBrow. We also demonstrated how standard inflationary operations can be used to simplify biological hypothesis composition and how the deflationary operators we defined in earlier work can be used to simplify evaluation. Finally, we have presented a future program of research that will allow us to incorporate our contradiction based framework at further levels of the research process.

Our work is founded on the belief that the engineering process should serve biologists, not operate in spite of them. We believe this philosophy, coupled with the formalisms we have introduced, will serve to create a
line of research that is both mathematically sound and biologically useful.

References


