Pathway Specification and Comparative Queries: A High Level Language with Petri Net Semantics

Saadat Anwar and Chitta Baral
CIDSE, Arizona State University, Tempe, AZ, USA

Abstract
Understanding biological pathways is an important activity in the biological domain for drug development. Due to the parallelism and complexity inherent in pathways, computer models that can answer queries about pathways are needed. A researcher may ask ‘what-if’ questions comparing alternate scenarios, that require deeper understanding of the underlying model. In this paper, we present overview of such a system we developed and an English-like high level language to express pathways and queries. Our language is inspired by high level action and query languages and it uses Petri Net execution semantics.

Introduction
Biological pathways are highly interconnected networks of biochemical processes. These processes execute autonomously in parallel, driven by natural constraints of ingredient supply and demand, i.e., a process can execute as soon as its preconditions are satisfied. For example, a metabolic reaction can occur as soon as sufficient quantities of its ingredients become available. Many processes are also governed by additional preconditions, including availability of additional substances that are not ingredients, substance gradients, inhibition, and stimulation. Collectively, these preconditions regulate the reactions through feedback loops and feed-forwards in the network. Reactions execute at different speeds, generating products on completion, which become ingredients for the down-stream reactions. The state of a pathway is defined by the available substance quantities, and the chain of reactions between a starting state and an ending state of a pathway define a trajectory of pathway’s state evolution. The interplay between limited quantity of ingredients and other preconditions can present multiple choices for alternate trajectory evolutions at each pathway state.

Understanding these pathways is of fundamental importance in biological research for disease diagnosis and drug development. However, the aforementioned complexities make it difficult for one person to retain all aspects of the pathway. As a result, computer based systems are needed to represent these pathways, allowing biologists to pose queries against them. An important class of questions in this regard are the so called ‘what-if’ questions, which compare alternate scenarios of a pathway. We find such questions in college level books that a professor may ask his students to gauge their understanding of a pathway. For example the following question from (Reece et al. 2010) appeared in a recent knowledge representation and reasoning challenge 1:

Question 1. “At one point in the process of glycolysis, both DHAP and G3P are produced. Isomerase catalyzes the reversible conversion between these two isomers. The conversion of DHAP to G3P never reaches equilibrium and G3P is used in the next step of glycolysis. What would happen to the rate of glycolysis if DHAP were removed from the process of glycolysis as quickly as it was produced?”

Answering such questions require simulating different scenarios and reasoning with the results. For example, question 1 asks for comparison of the rate of glycolysis between the nominal pathway and an alternate pathway in which DHAP is removed as quickly as it is produced.

In this paper we describe an English-like high level language to express pathways and queries. We highlight important features of its syntax and semantics and give an overview of an implementation that understands this language and answer questions about them. Our language is inspired by high level action and query languages such as (Gelfond and Lifschitz 1993; Giunchiglia and Lifschitz 1998; Lee, Lifschitz, and Yang 2013). However, compared to most existing action languages, which describe transition systems (Gelfond and Lifschitz 1993), our language describes trajectories. Our language is geared towards modeling natural systems, in which actions occur autonomously (Reiter 1996) when their pre-conditions are satisfied; and substance quantities do not become negative. Compared to most other action languages, substance quantities produced and consumed are additive 3. Our system also supports a richer query component, which is missing from most query languages that accompany action languages.

Copyright © 2014, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

1https://sites.google.com/site/2nddeepkrchallenge/
2Some languages like C+ (Giunchiglia et al. 2004) allow autonomous actions, but their query languages lack expressiveness.
3Although some languages like C+ have been extended to allow additive fluents (Lee and Lifschitz 2003).
Some aspects of our system are similar to (Baral et al. 2004), but their work is limited to signaling pathways, does not allow numeric quantities or has provision of loops inherent in biological pathways.

We use Petri Nets (Petersen 1977) for representing pathways as they mimic biological pathway diagrams and can represent parallel systems. For ease of reasoning and extensibility, we use Answer Set Programming (ASP) (Baral 2003) for simulating the pathway using the approach by (Anwar, Baral, and Inoue 2013a; 2013b).

The rest of the paper is organized as follows: we present important aspects of our high-level language syntax and semantics. Then we illustrate the use of our language with an example. After that, we briefly describe implementation of our system that understands this language and conclude with main contributions.

**Description of our system**
We present the key elements and discriminating factors of our high level language below.

**Pathway Specification Language**
The alphabet of pathway specification language \( \mathcal{P} \) consists of disjoint nonempty domain-dependent sets \( A, F, \) representing actions, and fluents, respectively; a fixed set \( S \) of firing styles; a fixed set \( K \) of keywords as syntactic sugar (shown in bold face); a fixed set of punctuations \{ ‘., ‘,’; ‘, ‘,’={ \}\}; and integers.

Each fluent \( f \in F \) has a domain \( \text{dom}(f) \) which is either integer or binary and specifies the values \( f \) can take. A state \( s \) is an interpretation of \( F \) that maps fluents to their values. We write \( s(f) = v \) to represent “\( f \) has the value \( v \) in state \( s \)”.

States are indexed, such that consecutive states \( s_i \) and \( s_{i+1} \) represent an evolution over one time step from \( i \) to \( i+1 \) due to firing of an action set \( T_i \) in \( s_i \). We illustrate the role of various constructs using a hypothetical example from the biological domain \(^*\):

\[
\begin{align*}
\text{domain of } sug \text{ is integer, } fac \text{ is integer, } \\
\text{acoa is integer, } h2o \text{ is integer} & \quad (1) \\
\text{box may execute causing } fac \text{ change value by } -1, & \\
\text{acoa change value by } +1 & \quad (2) \\
\text{if } h2o \text{ has value } 1 \text{ or higher} & \quad (3) \\
\text{inhibit box if } sug \text{ has value } 1 \text{ or higher} & \quad (4) \\
\text{initially } sug \text{ has value } 3, fac \text{ has value } 4, & \\
\text{acoa has value } 0, h2o \text{ has value } 0 & \quad (5)
\end{align*}
\]

Above query is about a process called beta-oxidation represented by ‘box’, the effect of which is captured in lines (2)-(3). Line (1) declares fluents for the substances used in the pathway and their domain, i.e. sugar (‘sug’), fatty-acids (‘fac’), acetyl-CoA (‘acoa’), and water (‘h2o’) are represented by numeric fluents. Line (2) describes the effect of beta oxidation on its inputs and outputs, i.e. when beta oxidation occurs, it consumes 1 unit of fatty-acids and produces 1 unit of acetyl-CoA. It implicitly defines a precondition that beta-oxidation cannot occur unless there is at least 1 unit of fatty-acids available. Line (3) describes an explicit precondition (or guard) of beta oxidation, i.e. it cannot occur unless there is at least 1 unit of water available. The water is, however, not consumed during beta oxidation. Line (4) explicitly inhibits beta-oxidation when there is any sugar available; and line (5) sets up the initial conditions of the pathway, s.t. initial quantities of sugar, fatty-acids, acetyl-CoA, and water are 3,4,0,0, respectively.

Now we introduce various statements and clauses, give their intuitive definitions, and show how they are combined to construct a pathway specification (or domain description).

In the following description, \( f \) is a fluent, \( a \) is an action, \( w \in \mathbb{N}^+ \cup \{0\}, d \in \mathbb{N}^+, \) \( S \in \{1, \ast, \max\}, e \in (\mathbb{N} \setminus \{0\}) \cup \{\ast\} \) for integer or \( e \in \{1, -1, \ast\} \) for binary fluents.

An effect clause has the form:

\[ f \text{ change value by } e \quad (6) \]

A guard cond clause can take the forms:

\[ f \text{ has value } w \text{ or higher} \quad (7) \]

\[ f \text{ has value lower than } w \quad (8) \]

The domain description contains statements of the forms:

\[
\begin{align*}
\text{domain of } f & \text{ is } \{\text{integer} | \text{binary}\} & (9) \\
\text{a may execute causing effect}_1, \ldots, \text{effect}_m & \quad (10) \\
\text{if guard cond}_1, \ldots, \text{guard cond}_n & \quad (11) \\
\text{initially } f \text{ has value } w & \quad (12) \\
\text{a executes in } d \text{ time units} & \quad (13) \\
\text{firing style } S & \quad (14)
\end{align*}
\]

where \( m > 0 \) and \( n \geq 0 \).

A fluent domain declaration statement (9) declares the values a fluent \( f \) can take. While binary domain is commonly used for representing substances in a signaling pathway, metabolic pathways use positive numeric values. Since the domain is for a physical entity, we disallow fluents with negative values. A may-execute statement (10) captures the pre-conditions of an action \( a \) and its impact. A single may-execute statement must not have effect\(_i\), effect\(_j\) with \( e_i < 0 \) and \( e_j < 0 \); or \( e_i > 0 \) and \( e_j > 0 \) for the same fluent. An inhibit statement (11) captures explicit inhibition conditions for an action \( a \). An initial condition statement (12) captures the initial state of pathway (e.g. as substance distribution). A duration statement (13) represents the duration \( d \) an action \( a \) takes to execute. A firing style statement (14) specifies how many actions may execute in parallel, where, \( S \) is either “1”, “\ast”, or “max” for serial execution, arbitrary amount of parallelism, and maximum parallelism. Actions execute automatically when fireable, subject to the available fluent quantities.

**Definition 1** (Pathway Specification). A pathway specification is composed of one or more domain, may-execute, inhibit, initially, duration statements, and one firing style statement.

A pathway specification is consistent if (i) there is at most one firing style statement (ii) at most one duration statement for an action \( a \); (iii) the guard cond\(_1\), \ldots, guard cond\(_n\) from a may-execute are disjoint from any other may-execute
for the same action $^5$; (iv) domain of fluents, effects, conditions and numeric values are consistent, i.e., effects and conditions on binary fluents must be binary; and (v) the pathway specification does not cause it to violate fluent domains by producing non-binary values for binary fluents.

When missing, duration of an action is assumed to be 1; and initial fluent quantity for a fluent is assumed to be 0.

Intuitively, a pathway specification $\mathbf{D}$ represents a set of trajectories of the form: $\sigma = s_0, T_0, s_1, \ldots, s_{k-1}, T_k, s_k$. Each trajectory encodes an evolution of the pathway. Starting from an initial state $s_0$, each $s_i, s_{i+1}$ pair represents the state evolution in one time step due to execution of the action set $T_i$ in state $s_i$ producing $s_{i+1}$. An action set $T_i$ is only executable in state $s_i$, if the total decrease of fluent values due to $e_i < 0$ and $e_i = 0$ will not result in any of the fluents becoming negative. Changes to fluents due to $e_i > 0$ for the action set $T_i$ occur over subsequent time-steps depending upon the durations of actions involved. Thus, the state $s_i(f_i)$ is the sum of $e_i > 0$ for actions of duration $d$ that occurred $d$ time steps before (current time step) $i$, i.e. $a \in T_i - d$.

Query Specification Language

The alphabet of query language $Q$ consists of the same sets $A,F$ from $P$ representing actions, and fluents, respectively; a fixed set of reserved keywords $K$ shown in bold in syntax below; a fixed set $\{\cdot, :, ;, ', ',', ', '+', '-', '|', '\rangle', '\langle', ';', '(', ')', '<', '>', '==', '!=\}$ of punctuations; a fixed set of $\{<, >, ==\}$ of directions; and constants. Our query language asks questions about biological entities and processes in a biological pathway described by a pathway specification (or domain description). A query statement is composed of a query description (the quantity, or property being sought by the question), interventions (changes to the pathway), observations (about states and actions of the pathway), and initial setup conditions. For example, the following query indirectly determines the direction of change in the rate of glycose by comparing the rate of production of ‘bpg13’ w.r.t. the glycose pathway given in (Reece et al. 2010, Figure 9.9); and corresponds to question (1).

direction of change in average
rate of production of bpg13 is $d$
when observed between time step $0$ and time step $k$;
comparing nominal with modified pathway obtained due to interventions:
remove dhap as soon as produced;
using initial setup:
continuously supply $f16hp$ in quantity 1; $^6$

Intuitively, a query statement is evaluated against the trajectories of a pathway (or domain description). The pathway is first modified by first applying the initial setup conditions, and the interventions. The modified pathway is then simulated and its trajectories are filtered to retain only those which satisfy the observations specified in the query statement. Next we define the syntax of the query language and its elements $^6$, give their intuitive meaning, and how these components fit together to form a query statement. In the following description, $a$’s are actions, $f$’s are fluents, $n$’s are numbers, $q$’s are positive integer numbers, $d$ is one of the directions from $\{<, >, ==\}$.

\[
\langle \text{point} \rangle ::= \text{time step } t s \quad (16)
\langle \text{interval} \rangle ::= (\langle \text{point} \rangle \text{ and } (\langle \text{point} \rangle \quad (17)
\langle \text{aggop} \rangle ::= \text{minimum} | \text{maximum} | \text{average} \quad (18)
\langle \text{quant intf} \rangle ::= \text{rate of production of } f \text{ is } n \quad (19)
| \text{rate of firing of } a \text{ is } n \quad (20)
\langle \text{quant ptf} \rangle ::= \text{value of } f \text{ is higher than } n \quad (21)
| \text{value of } f \text{ is } n \quad (22)
\langle \text{qual intf} \rangle ::= f \text{ is accumulating} \quad (23)
\langle \text{qual ptf} \rangle ::= a \text{ occurs}\quad (24)
| a \text{ does not occur}\quad (25)
| a_1 \text{ switches to } a_2 \quad (26)
\langle \text{quant agg intf} \rangle ::= \langle \text{aggop} \rangle \text{ rate of firing of } a \text{ is } n \quad (27)
| \langle \text{aggop} \rangle \text{ rate of production of } f \text{ is } n \quad (28)
\langle \text{quant agg ptf} \rangle ::= \langle \text{aggop} \rangle \text{ value of } f \text{ is } n \quad (29)
\langle \text{quant cagg intf} \rangle ::= \text{direction of change in}\quad (30)
\langle \text{aggop} \rangle \text{ rate of production of } f \text{ is } d
| \text{direction of change in}
\langle \text{aggop} \rangle \text{ rate of firing of } a \text{ is } d \quad (31)
\langle \text{quant cagg ptf} \rangle ::= \text{direction of change in}
\langle \text{aggop} \rangle \text{ value of } f \text{ is } d \quad (32)
\langle \text{simp intf} \rangle ::= \langle \text{quant intf} \rangle | \langle \text{qual intf} \rangle \quad (33)
\langle \text{simp ptf} \rangle ::= \langle \text{quant ptf} \rangle | \langle \text{qual ptf} \rangle \quad (34)
\langle \text{int obs} \rangle ::= \langle \text{simp ptf} \rangle | \langle \text{simp ptf at } (\langle \text{point} \rangle) | \langle \text{simp intf} \rangle \quad (35)
| \langle \text{simp intf when observed between } (\text{interval}) \rangle \quad (36)
\langle \text{qdesc} \rangle ::= \langle \text{int obs} \rangle \quad (37)
| \langle \text{int obs in all trajectories} \rangle \quad (38)
| \langle \text{quant agg intf} \rangle
\quad \text{when observed between } (\text{interval}) \quad (39)
| \langle \text{quant agg ptf when observed at } (\langle \text{point} \rangle) \quad (40)
\langle \text{qdesc} \rangle ::= \langle \text{quant agg intf} \rangle
\quad \text{when observed between } (\text{interval}) \quad (41)
| \langle \text{quant agg ptf when observed at } (\langle \text{point} \rangle) \quad (42)
\langle \text{interv} \rangle ::= \text{remove } f \text{ as soon as produced} \quad (43)
| \langle \text{disable } a \rangle \quad (44)
\langle \text{continuous supply } f \text{ in quantity } q \rangle \quad (45)
\langle \text{add delay of } q \text{ time units in availability of } f \rangle \quad (46)
\langle \text{set value of } f \text{ to } q \rangle \quad (47)
\langle \text{cond} \rangle ::= \langle \text{45} \rangle | \langle \text{47} \rangle \quad (48)
\langle \text{qstmt} \rangle ::= (\langle \text{45} \rangle \quad (49)
\quad \text{due to interventions} : (\langle \text{interv} \rangle)_{1}, \ldots, (\langle \text{interv} \rangle)_{N_{1}};
\quad \text{due to observations} : (\langle \text{int obs} \rangle)_{1}, \ldots, (\langle \text{int obs} \rangle)_{N_{2}};
\quad \text{using initial setup} : (\langle \text{cond} \rangle)_{1}, \ldots, (\langle \text{cond} \rangle)_{N_{3}}; \quad (49)

\footnote{Note that ‘$f$ has value 5 or higher’ overlaps with ‘$f$ has value 7 or higher’.

\footnote{Although some of our single-trajectory queries can be represented as LTL formulas, we have chosen to keep the current representation as it is more intuitive for our biological domain.}
Given a pathway specification (domain description) \( P \) with trajectories of the form \( \sigma = s_0, T_0, s_1, \ldots, s_k \), where each \( s_i, T_i \) is an evolution from state \( s_i \) to state \( s_{i+1} \) due to firing of \( T_i \). Intuitively, a point is a time-step and interval defines a continuous range of time-steps on a trajectory, then point formulas (represented by \( pt_f \)) are evaluated w.r.t. a point on the trajectory, while interval formulas (represented by \( int f \)) are evaluated w.r.t. an interval on a trajectory, e.g., rate of production of \( f \) over interval \([i, j]\) is given by \( (s_j(f) - s_i(f))/(j - i) \). Quantitative formulas (represented by \( quant \)) are evaluated for some quantity \( n \), while qualitative formulas (represented by \( qual \)) are evaluated for some qualitative attribute of a state or trajectory. Aggregate quantitative formulas (represented by \( quant agg \)) are evaluated for some quantity \( n \), which is an aggregate of quantities \( n_1, \ldots, n_m \) for trajectories \( \sigma_1, \ldots, \sigma_m \), e.g., the average aggregate is computed as \( n = (n_1 + \cdots + n_m)/m \). Comparative quantitative formulas (represented by \( quant cagg \)) are evaluated for some direction of change between aggregate quantities \( n \) (over trajectories \( \sigma_1, \ldots, \sigma_m \)) and \( n' \) (over trajectories \( \sigma'_1, \ldots, \sigma'_m \)), e.g., the change direction is ‘\( > \)’ if \( n' > n \).

Intuitively, an internal observation (represented by \( int ob \)) is a simple point formula that holds at any point on a trajectory, a simple point formula that holds at a specific point, a simple interval formula that holds over any interval on a trajectory, or a simple interval formula that holds over a specific interval. Intuitively, an internal observation filters the set of trajectories produced by a pathway specification.

Intuitively, a query description (represented by \( qdesc \)) is one of the possible point formulas that holds at a specific point, an interval formula that holds over a specific range, an internal observation, an internal observation over all trajectories in a set of given trajectories. A comparative query description (represented by \( cqdesc \)) is one of the quantitative comparative aggregate point formula at a specific point on two sets of trajectories, or a quantitative comparative aggregate interval formula over a specific interval on two sets of trajectories. Intuitively, a query description or a comparative query description specifies the property that we want to have hold true over the trajectories of the domain pathway.

Intuitively, an intervention (represented by \( interv \)) specifies a modification to the pathway specification, e.g., intervention (43) modifies the pathway such that all quantity of \( f \) is removed as soon as it is produced.

Intuitively, a query statement (represented by \( qstmt \)) is a comparative query statement if it contains a comparative query description, and non-comparative otherwise. A query statement is composed of a query statement, interventions to the pathway, internal observations to filter the trajectories, and initial conditions. Intuitively, a query statement asks whether a query description holds in a pathway, after modifying it with initial setup, interventions and observations.

While, a comparative query statement asks whether a query description holds when a nominal pathway is compared to a modified pathway, subject to same initial setup, but interventions and observations only applied to the modified pathway.

**Pathway Semantics**

The semantics of our pathway are given by a Guarded-Arc Petri Net, which allows choice between different effects (arc-sets) of a transition, such that only one effect (arc-set) is active for a transition in a given state determined by its arc-guard.

**Definition 2 (Guard).** A guard condition takes one of the following forms: \((f < v), (f \leq v), (f > v), (f \geq v)\), or \((f = v)\), where \( f \) is a fluent; and \( v \) is a fluent or a numeric constant. A guard is a propositional formula of guard conditions, with each guard condition treated as a proposition.

An interpretation of a guard \( G \) is a possible assignment of a value to each fluent \( f \in G \) from the domain of \( f \). A guard \( G \) is satisfied w.r.t. a state \( s \), written \( s \models G \) iff \( G \) has an interpretation in which each of its fluents \( f \) has the value \( s(f) \) and \( G \) is true.

**Definition 3 (Guarded-Arc Petri Net).** A Guarded-Arc Petri Net is a tuple \( PN^G = (P, T, G, E, R, W, D, TG, L) \):

- \( P \) is a finite set of places
- \( T \) is a finite set of transitions
- \( G \) is a set of guards as defined in definition (2)
- \( TG : T \rightarrow G \) are the transition guards
- \( E \subseteq (T \times P \times G) \cup (P \times T \times G) \) are the guarded arcs
- \( R \subseteq P \times T \times G \) are the guarded reset arcs
- \( W : E \rightarrow \mathbb{N}^+ \) are arc weights
- \( D : T \rightarrow \mathbb{N}^+ \) are the transition durations
- \( L : P \rightarrow \mathbb{N}^+ \) specifies maximum tokens for each place

subject to constraints: (i) \( P \cap T = \emptyset \) (ii) \( R \cap E = \emptyset \) (iii) Let \( t \in T \) be a transition, and \( gg = \{ g : (t, p, g) \in E \} \cup \{ g : (p, t, g) \in R \} \cup \{ g : (p, t, g) \in R \} \) be the set of arc-conditions for normal and reset arcs connected to it, then any two distinct guards \( g_1 \in gg_1, g_2 \in gg_2 \) must not have an interpretation that makes both \( g_1 \) and \( g_2 \) true.

We make a simplifying assumption that all places are readable by using their place names. Execution of the \( PN^G \) occurs in discrete time steps. The marking (or state) of a Guarded-Arc Petri Net \( PN^G \) is the token assignment of each place \( p_i \in P \). Initial marking is given by \( M_0 : P \rightarrow N^\geq0 \), while the token assignment at step \( k \) is written as \( M_k \). Next we define the execution semantics of \( PN^G \). We start with terminology used below. Let (i) \( s_0 = M_0 \) represent the initial marking (or state), \( s_k = M_k \) represent the marking (or state) at time step \( k \), (ii) \( s_k(p_i) \) represent the marking of place \( p_i \) at time step \( k \), such that \( s_k = [s_k(p_0), \ldots, s_k(p_n)] \), where \( P = \{p_0, \ldots, p_n\} \) (iii) \( T_k \)

---

3 Our model is similar to the model in (Jensen, Kristensen, and Wells 2007) in many aspects with differences in certain key semantics related to biological modeling, such as reset arcs.
be the firing-set that fired in step \( k \), (iv) \( e_{nk} \) be the set of enabled transitions in state \( s_k \), (v) \( del_k(p, \{ t_1, \ldots, t_n \}) \) be the sum of tokens that will be consumed from place \( p \) if transitions \( t_1, \ldots, t_n \) fired in state \( s_k \), (vi) \( overc_k(\{ t_1, \ldots, t_n \}) \) be the set of places that will have over-consumption of tokens if transitions \( t_1, \ldots, t_n \) were to fire simultaneously in state \( s_k \), (vii) \( sel_k(f) \) be the set of possible firing-set choices in state \( s_k \) using \( f \) firing style (viii) \( add_k(p) \) be the total production of tokens in place \( p \) in state \( s_k \) due to actions terminating in state \( s_k \), (ix) \( s_{k+1} \) be the next state computed from state \( s_k \) due to firing transition-set \( T_k \). Then, the execution semantics of the Guarded-Arc Petri Net PN^G starting from state \( s_0 \) using firing-style \( f \) is given as follows:

\[
en_k = \{ t : t \in T, s_k = TG(t), \forall (p, t, g) \in E, (s_k \models g, s_k(p) \geq W(p, t, g)) \}
\]

\[
del_k(p, \{ t_1, \ldots, t_n \}) = \\
\sum_{i=1,...,n} W(p, t, g) : (p, t, g) \in E, s_k \models g
\]

\[
+ \sum_{i=1,...,n} s_k(p) : (p, t, g) \in R, s_k \models g
\]

\[
overc_k(\{ t_1, \ldots, t_n \}) = \{ p : p \in P, \}
\]

\[
s_k(p) < del_k(p, \{ t_1, \ldots, t_n \})
\]

\[
sel_k(1) = \{ s \in s_k \}
\]

\[
sel_k(*) = \{ s \in s_k \}
\]

\[
sel_k(max) = \{ s \in s_k \}
\]

\[
add_k(p) = \sum_{j=0,...,k} W(t, p, g)
\]

\[
T_k = sel_k(f)
\]

\[
add_k(p) = \sum_{j=0,...,k} W(t, p, g)
\]

Definition 4 (Trajectory). \( \sigma = s_0, T_0, s_1, \ldots, s_{k-1}, T_k-1, s_k \) is a trajectory of PN^G if given \( s_0 = M_0 \), each \( T_i \) is a possible firing-set in \( s_i \) whose firing produces \( s_{i+1} \) per PN^G’s execution semantics in (51).

Query Semantics

First we give the semantics of the domain modification due to an intervention by examples. Consider intervention (43), we create the modified domain description \( D' = D \ominus f \) (remove \( f \) as soon as produced) as follows:

\[
D' = D + \{ t_j \text{ may execute causing } f \text{ change value by } + \}
\]

Applying intervention (45) \( D' = D \ominus f \) (continuously supply \( f \) in quantity \( q \) results in the following changes:

\[
D' = D + \{ t_j \text{ may execute causing } f \text{ change value by } + q \}
\]

Next we define the semantics of some common formulas and observation using LTL-style. Let \( \sigma = s_0, T_0, s_1, \ldots, T_{k-1}, s_k \) represent a trajectory of domain \( D \) with initial marking \( s_0 \). Let actions \( T_i \) firing in state \( s_i \) be observable in \( s_i \) such that \( T_i \subseteq s_i \).

Let \( \langle s_i, \sigma \rangle \models F \) represent \( F \) holds at point \( i \) in \( \sigma \);

\[
\{ (s_i, \sigma_1), \ldots, (s_i, \sigma_m) \} \models F \text{ represent } F \text{ holds at point } i \text{ in trajectories } \sigma_1, \ldots, \sigma_m;
\]

\[
\{ (s_i, \sigma_1), \ldots, (s_i, \sigma_m) \} \models F \text{ represent } F \text{ holds at point } i \text{ in both sets } \sigma_1, \ldots, \sigma_m \text{ and } \{ \sigma_1, \ldots, \sigma_m \};
\]

\[
\{ (s_i, \sigma), (s_i, \sigma) \} \models F \text{ represent } F \text{ holds over interval } [i,j] \text{ in } \sigma;
\]

\[
\{ (s_i, \sigma_1), \ldots, (s_i, \sigma_m) \} \models F \text{ represent } F \text{ holds over interval } [i,j] \text{ in } \sigma_1, \ldots, \sigma_m; \text{ and } \{ (s_i, \sigma_1), \ldots, \{ (s_i, \sigma_1), \ldots, (s_i, \sigma_m) \}, (s_i, \sigma_1), \ldots, (s_i, \sigma_m) \} \models F \text{ represent } F \text{ holds over interval } [i,j] \text{ over two sets } \sigma_1, \ldots, \sigma_m \text{ and } \{ \sigma_1, \ldots, \sigma_m \}; \text{ then the semantics of a rate query are given as follows:}
\]

\[
(\langle s_i, \sigma \rangle, j) \models \text{ rate of production of } f \text{ is } n
\]

\[
(\{ (s_i, \sigma_1), \ldots, (s_i, \sigma_m) \}, j) \models \text{ average rate of production of } f \text{ is } r
\]

\[
\text{ if } \exists t_1, \ldots, t_n : (\langle s_i, \sigma_i \rangle, j) \models \text{ rate of production of } f \text{ is } r_1, \ldots, (\langle s_i, \sigma_m \rangle, j) \models \text{ rate of production of } f \text{ is } r_m \text{ and } r = \frac{r_1 + \ldots + r_m}{m}
\]

\[
\{ (\langle s_i, \sigma_1 \rangle, \ldots, (s_i, \sigma_m) \}, (\langle s_i, \sigma_1 \rangle, \ldots, (s_i, \sigma_m) \}, j) \models \text{ direction of change in average rate of production of } f \text{ is } d
\]

\[
\text{ if } \exists t_1, \ldots, t_n : (\langle s_i, \sigma_1 \rangle, \ldots, (s_i, \sigma_m) \}, j) \models \text{ average rate of production of } f \text{ is } n_1 \text{ and }
\]

\[
\exists t_2, \ldots, t_n : (\langle s_i, \sigma_1 \rangle, \ldots, (s_i, \sigma_m) \}, j) \models \text{ average rate of production of } f \text{ is } n_2 \text{ and } n_2 \neq n_1
\]

\[
\{ (\langle s_i, \sigma_1 \rangle, \ldots, (s_i, \sigma_m) \}, (\langle s_i, \sigma_1 \rangle, \ldots, (s_i, \sigma_m) \}, j) \models \text{ direction of change in (average rate of production of } f \text{ is } d
\]

Definition 5. Let \( D \) be a domain description and \( Q \) be a query statement (49) with query description \( U \), interventions \( V_1, \ldots, V_{\mid V \mid} \text{, internal observations } O_1, \ldots, O_{\mid O \mid} \text{, and initial conditions } I_1, \ldots, I_{\mid I \mid} \text{. Let } D_1 \equiv D \oplus I_1 \oplus \cdots \oplus I_{\mid I \mid} \oplus V_1 \ominus \cdots \ominus V_{\mid V \mid} \text{ be the modified domain description constructed by applying the initial conditions and interventions from } Q \text{ and } \sigma_1, \ldots, \sigma_m \text{ be } \begin{align*}
\text{ its trajectories. Let } D_1 \equiv D \oplus I_1 \oplus \cdots \oplus I_{\mid I \mid} \oplus \begin{cases} V_1 & \text{if } V_1 \text{ is the alternate domain description constructed by applying the initial conditions and interventions from } Q \text{ and } \sigma_1, \ldots, \sigma_m \text{ be its trajectories that sat-}
\end{align*}
\end{align*}
isfy $O_1, \ldots, O_{|O|}$. Then, $D$ satisfies $Q$ (written $D \models Q$) if \((\{\sigma_1, \ldots, \sigma_m\}, \{\bar{\sigma}_1, \ldots, \bar{\sigma}_m\}) \models U$.

The following propositions follow from the above description.

**Proposition 1.** Let $D$ be a domain description and $Q$ be a non-comparative query statement. Then, $D \models Q$ iff $D' \models Q'$ where $D' \equiv D \circ I_1 \cdots \circ I_{|I|} \circ V_1 \cdots \circ V_{|V|}$ and $Q' \equiv Q$ with $I_1, \ldots, I_{|I|}, V_1, \ldots, V_{|V|}$ removed, where $I_1, \ldots, I_{|I|}$ are the initial conditions in $Q$ and $V_1, \ldots, V_{|V|}$ are the interventions in $Q$.

**Proposition 2.** Let $D$ be a domain description and $Q$ be a non-comparative query statement. Then, $D \models Q$ iff $\{\sigma_1, \ldots, \sigma_n\} \models Q'$, where $\sigma_1, \ldots, \sigma_n$ are trajectories of $D'$ that satisfy $O_1, \ldots, O_{|O|}$,

$D' \equiv D \circ I_1 \cdots \circ I_{|I|} \circ V_1 \cdots \circ V_{|V|}$ and $Q' \equiv Q$ with $I_1, \ldots, I_{|I|}, V_1, \ldots, V_{|V|}, O_1, \ldots, O_{|O|}$ removed; $I_1, \ldots, I_{|I|}$ are the initial conditions in $Q$, $V_1, \ldots, V_{|V|}$ are the interventions in $Q$, and $O_1, \ldots, O_{|O|}$ are internal observations in $Q$.

**Proposition 3.** Let $D$ be a domain description and $Q$ be a comparative query statement. Then, $D \models Q$ iff $\{\sigma_1, \ldots, \sigma_n\}, \{\sigma'_1, \ldots, \sigma'_n\} \models Q'$, where $\sigma_1, \ldots, \sigma_n$ are trajectories of $D''$, and $\sigma'_1, \ldots, \sigma'_n$ are trajectories of $D'''$ that satisfy $O_1, \ldots, O_{|O|}$,

$D'' \equiv D \circ I_1 \cdots \circ I_{|I|}, \text{ or } V_1 \cdots \circ V_{|V|}$,$\text{ or } I_1, \ldots, I_{|I|}$ are the initial conditions in $Q$, $V_1, \ldots, V_{|V|}$ are the interventions in $Q$, and $O_1, \ldots, O_{|O|}$ are internal observations in $Q$.

Intuitively, proposition 1 states that a domain description $D$ satisfies a (non-comparative) query statement $Q$ whenever the modified domain description $D'$ constructed by applying interventions and initial conditions from $Q$ to $D$ satisfies the modified query statement $Q'$ constructed by removing interventions and initial conditions from $Q$. Proposition 2 states that a domain description $D$ satisfies a (non-comparative) query statement $Q$ whenever the trajectories of the modified domain description $D'$ constructed by applying interventions and initial conditions from $Q$ to $D$ that satisfy observations in $Q$ satisfy the modified query statement $Q'$ constructed by removing interventions, initial conditions, and observations from $Q$. Proposition 3 states that a domain description $D$ satisfies a comparative query statement $Q$ whenever the comparison between trajectories of the nominal domain description $D'$ constructed by applying initial conditions from $Q$ and the trajectories of the alternate domain description $D''$ constructed by applying interventions and initial conditions from $Q$ that satisfy the observations in $Q$ satisfy the query statement $Q'$ constructed by removing interventions, initial conditions and observations from $Q$.

Proof of the above propositions are based on the definitions $5, 6$ of satisfiability of a query statement $Q$ by domain $D$: the semantics of the query description $U$ in the query statement; and the structure of the query statements. The forward direction is proven by showing that the trajectories of the modified domain descriptions (constructed by applying initial conditions and interventions) filtered by the internal observations (as appropriate) satisfy the query description $U$ in the query statement $Q$. The reverse direction is proven by showing that one can select a domain description $D$ that, when modified through selected initial setup conditions and interventions represents the same trajectories as $D'$, and one can select observations that filter these trajectories to only those satisfied by the query description $U$. The selected initial setup conditions, interventions, and the observations when added to $Q'$ give us $Q$.

**Illustrative Example**

We illustrate our high level language by applying it to question (1) and the relevant glycolysis pathway.

The following pathway specification encodes the domain description $D$ for question (1) and produces the PN in Fig. 1 minus the $tr, 13$ transitions:

- **domain of** $f16bp$ is integer, $dhap$ is integer, $g3p$ is integer, $bpg13$ is integer
- $t4$ may execute causing $f16bp$ change value by -1, $dhap$ change value by +1, $g3p$ change value by +1
- $t5a$ may execute causing $dhap$ change value by -1, $g3p$ change value by +1
- $t5b$ may execute causing $g3p$ change value by -1, $dhap$ change value by +1
- $t6$ may execute causing $g3p$ change value by -1, $bpg13$ change value by +2
- **initially** $f16bp$ has value 0, $dhap$ has value 0, $g3p$ has value 0, $bpg13$ has value 0
- **firing style** max

The question is asking for the direction of change in the rate of glycolysis when the nominal pathway is compared against a modified pathway in which $DHAP$ is removed as soon as it is produced. Since this rate can vary with the trajectory of the world evolution, we consider the average change in rate. Using domain knowledge (Reece et al. 2010, Figure 9.9), we measure the rate of glycolysis indirectly by measuring the rate of production of $bpg13$, a downstream product, which is converted into equivalent quantity of the end product of glycolysis. We also add continuous supply of source ingredient $f16bp$ at the rate of 1 unit per time step to prevent starvation. This results in a query statement (15) $Q$ for some simulation length $k$. We evaluate it for the direction $d$ using
our deep reasoning system, which gives us \( d \neq '<'; \) suggesting that the rate of glycolysis will be lowered when DHAP is removed as soon as it is produced.

**Implementation**

We implemented a system \(^8\) that understands a subset of our high level language using Python as the driver as well as the high level language parser; and Clingo (Gebser et al. 2011) as the ASP solver. The system takes a pathway specification, a query specification, and simulation parameters, such as simulation length and maximum possible tokens at any place for query evaluation. Our system uses Petri Net semantics for modeling the biological pathway and ASP for simulating it. Interventions in questions are modeled as Petri Net extensions and translated into ASP using the approach in (Anwar, Baral, and Inoue 2013a; 2013b). This gave us the translations of extensions such as reset arcs, inhibit arcs (Peterson 1977), read arcs (Christensen and Hansen 1993), colored tokens (Peterson and others 1980), and timed transitions (Ramchandani 1974) etc. We augmented the encoding further to include additional extensions, such as conditional effects of actions, and more generalized inhibitions supported by our language.

Next, we briefly summarize how our system processes a non-comparative aggregate quantitative query statement, then describe how the comparative quantitative query from previous section is evaluated.

To evaluate a non-comparative query, the system builds a Petri Net model from the pathway specification. It then applies the initial conditions and interventions from the query statement to this model. The model is then translated into ASP. The resulting ASP code is augmented with constraints for (internal) observations. Answer sets of the augmented code provide the filtered trajectories. Atoms needed to compute the quantity specified in the query (e.g. rate of production) are extracted from the answer sets and quantity values aggregated across answer sets. The aggregated value is compared against any aggregate value provided for query state.

To evaluate comparative quantitative query statement from the previous section is decomposed into subqueries, \( Q_0 \) for nominal case and \( Q_1 \) for the modified case:

\[
Q_0 \equiv \text{average rate of production of } bpg13 \text{ is } n_{avg} \text{ when observed between time step } 0 \text{ and time step } k; \text{ using initial setup: } \\
\text{continuously supply } f16bp \text{ in quantity 1;}
\]

\[
Q_1 \equiv \text{average rate of production of } bpg13 \text{ is } n'_{avg} \text{ when observed between time step } 0 \text{ and time step } k; \text{ due to interventions: } \\
\text{remove dhap as soon as produced; } \\
\text{using initial setup: } \\
\text{continuously supply } f16bp \text{ in quantity 1;}
\]

The queries are evaluated w.r.t. the same initial conditions for average rates \( n_{avg}, n'_{avg} \) that satisfy \( d \) in the original query statement. The Petri Net model in Figure 1 shows nominal case (\( D_0 \)) in solid lines and the interventions added for the alternate case (\( D_1 \)) as dotted lines. The average results are compared using \( d \) to determine if they specify the direction given in the comparative query statement.

**Conclusion**

We have presented a high level (English like) language for specifying pathways and asking queries against them. Our pathway specification language uses Petri Net semantics as the simulation model and our query language is inspired by action languages. Our query language is one of the main contributions of this paper. It allows aggregate queries over a set of trajectories, comparative aggregate queries over two sets of trajectories, and interventions that are more general than actions which can be used to modify the pathway as specified in a query statement. Our approach improves on the query languages associated with action languages, by implementing comparative queries.

Though some aspects of our language may appear cumbersome to a person with background in action languages, we retained the syntax to allow coherent description of a variety of complex interventions and queries.

We illustrated how an example question comparing alternate scenarios of a biological pathway is encoded in our high level language; and summarized how our implementation performs query evaluation. We will elaborate on these aspects in companion papers.

Our approach fits into a larger problem of representing and reasoning about biological pathways gaining interest lately, geared towards developing an end-to-end system that extracts biological knowledge from texts, assembles it into pathways, determines the right level of abstraction eliminating irrelevant details, and answers such questions about them that require understanding of the inner workings of these pathways; with the ability to pose questions in natural language and present results in a natural language or a visual representation such as graphs. These aspects serve as guidelines for future extension to our work, and indeed there is existing work on many of these, including from our group, e.g., see (Tari et al. 2009; 2010) for pathway construction from text extraction and translating natural language questions to formal queries that we will extend.

The importance of determining the right level of abstraction by eliminating irrelevant details is two fold. On one hand it can improve performance by reducing problem size and computational resources needed, while on the other, it presents cleaner results that are easier to interpret by eliminating extraneous information. In this regard, techniques from multi-scale modeling (see (Dada and Mendes 2011; Heiner and Gilbert 2013)) could be applicable.

Additional areas of improvements include encapsulating time information in the queries such that it is hidden from the user, as well as the ability to apply interventions at arbitrary points during the simulation. Our current implementation uses Clingo, which handles discrete quantities only. We intend to extend this work to include continuous real numbers based on work by (Lee and Meng 2013).

---

\(^8\)https://sites.google.com/site/deepqa2014/
References


