Journal of Bioinformatics and Computational Biology
Vol. 8, No. 2 (2010) 247–293

DOI: 10.1142/S0219720010004604

Imperial College Press
www.icpress.co.uk

QUERYING METABOLISM UNDER DIFFERENT PHYSIOLOGICAL CONSTRAINTS*

ALI CAKMAK[†,§], GULTEKIN OZSOYOGLU[†,¶]
and RICHARD W. HANSON[‡,‖]

[†]*Department of Electrical Engineering & Computer Science*

[‡]*Department of Biochemistry, Case Western Reserve University*
Cleveland, OH 44106, USA
[§]*cakmak@case.edu*
[¶]*tekin@case.edu*
[‖]*rwh@case.edu*

Received 19 July 2009
Revised 2 November 2009
Accepted 2 November 2009

Metabolism is a representation of the biochemical principles that govern the production, consumption, degradation, and biosynthesis of metabolites in living cells. Organisms respond to changes in their physiological conditions or environmental perturbations (i.e. constraints) via cooperative implementation of such principles. Querying inner working principles of metabolism under different constraints provides invaluable insights for both researchers and educators. In this paper, we propose a metabolism query language (MQL) and discuss its query processing. MQL enables researchers to explore the behavior of the metabolism with a wide-range of predicates including dietary and physiological condition specifications. The query results of MQL are enriched with both textual and visual representations, and its query processing is completely tailored based on the underlying metabolic principles.

Keywords: Metabolism; query language; biochemical networks; metabolomics; metabolic pathways; physiological states; metabolite changes; metabolic perturbations; metabolic simulation.

1. Introduction

Metabolic pathways describe the biochemical processes that are essential to survival and adaptation of organisms in different environments. Metabolism is governed via the collaborative work of such cellular processes with complex and highly developed regulation mechanisms. Sophisticated organization and control of metabolism is crucial for organisms to maintain an adaptive and dynamic behavior in different physiological conditions. Querying metabolism under different stress conditions

*This work is supported by the National Science Foundation under grants DBI-0849956, DBI-0743705, CRI-0551603, and CCF-0820217.

enables life science researchers and students to gain insight about the possible behavior of metabolism.

In this paper, we (a) propose the *Metabolism Query Language* (*MQL*) that allows users to pose in-depth biochemistry-based queries, (b) discuss query processing needs of *MQL* at a comprehensive level of metabolic biochemistry, and (c) present query processing techniques for *MQL*. To this end, we characterize essential biochemistry principles into 20 query processing rules which serve as the underlying framework for *MQL*'s query processing specification. More specifically, *MQL* takes into consideration (i) regulative relationships between different cellular processes, (ii) functional and physical pools of metabolites, (iii) functional differentiation of biological compartments, (iv) variations in trigger conditions for activation/ inhibition of different processes, and (v) distinct enzyme regulation mechanisms.

MQL enables users to specify multiple and different classes of queries, such as (i) computing (and visualizing) "*Activated/Inactivated* (*metabolic*) *Paths*" with increased and decreased fluxes under specified physiological conditions (MQL_{AIP} queries), (ii) identifying/verifying "*Potential Futile Cycles*" (MQL_{PFC} queries), (iii) querying for required metabolic concentration change sets to prevent a particular futile cycle, (iv) searching for concentration change sets which lead to the (in)activation of a user-specified metabolic subnetwork, and (v) exploring the metabolic behavior of a set of (possibly reversible) reactions. Our framework allows users to input concentration change statements on key metabolites, and incorporates such input into its query processing. This work expands beyond *MQL*, and constitutes a computational infrastructure for an informed reasoning of metabolomics data.[1] Please note that, while *MQL* visualizes its query outputs, pathway visualization issues are not the focus of this paper, and not discussed.

To demonstrate the capabilities of *MQL* framework, we employ, as an example, computational modeling of mammalian (in particular, human) metabolism, and specifying and processing queries over its metabolic network. In this study, we focus only on MQL_{AIP} and MQL_{PFC} queries. Next, we illustrate MQL_{AIP} queries with an example.

1.1. *A query template and its instance*

A pathway consists of a consecutive sequence of biochemical reactions (steps), where each product (output) of a reaction becoming a substrate (input) of the following reaction. Each pathway has a certain set of starting substrates, intermediates, and end-products. Hence, all the reactions in a pathway work towards a common goal: converting substrate(s) of the pathway into the product(s) of the pathway.

Next we informally specify MQL_{AIP} queries via a "template" and illustrate with an example.

***MQL_{AIP} Query Template*:**

Given: I. A subset P of pathways in the human metabolic network
II. A set of biological compartments

III. A set C of conditions specifying

1. Metabolic and dietary states/physiological conditions, e.g. those that control the fuel consumption of the metabolism (such as fasting, starvation, after-a-meal-devoid-of-carbohydrates, dietary imbalance, alcohol consumption), or specific disease states such as diabetes, **and/or**
2. State changes (increases/decreases) of "key metabolites" such as increases in *lactate*, *pyruvate*, *amino acids*.

(a) **Find** activated (increased flux) and inactivated (decreased flux) paths in selected pathways in P (i.e. explicitly show the activated/inactivated flux directions).
(b) **Visualize** a selected subset of pathways in P in full and the remainder in collapsed form, for simplicity in visualization.
(c) Using the metabolic biochemistry, **explain** the reasons for blocked (i.e. inactivated) reaction directions in the selected subnetwork.

Next we present a sample query that follows from the above query template.

***A Sample MQL$_{AIP}$ Query Instance and Its Output*:**

Given: I. Selected pathways P: *Glycolysis, Gluconeogenesis, TCA Cycle, Beta Oxidation, Ketone Body Synthesis, and Fatty Acid Synthesis*

II. Selected biological compartment(s): *Mitochondrion*, *Cytosol*, and *Endoplasmic Reticulum*, all in *Liver*

III. A set C of conditions:

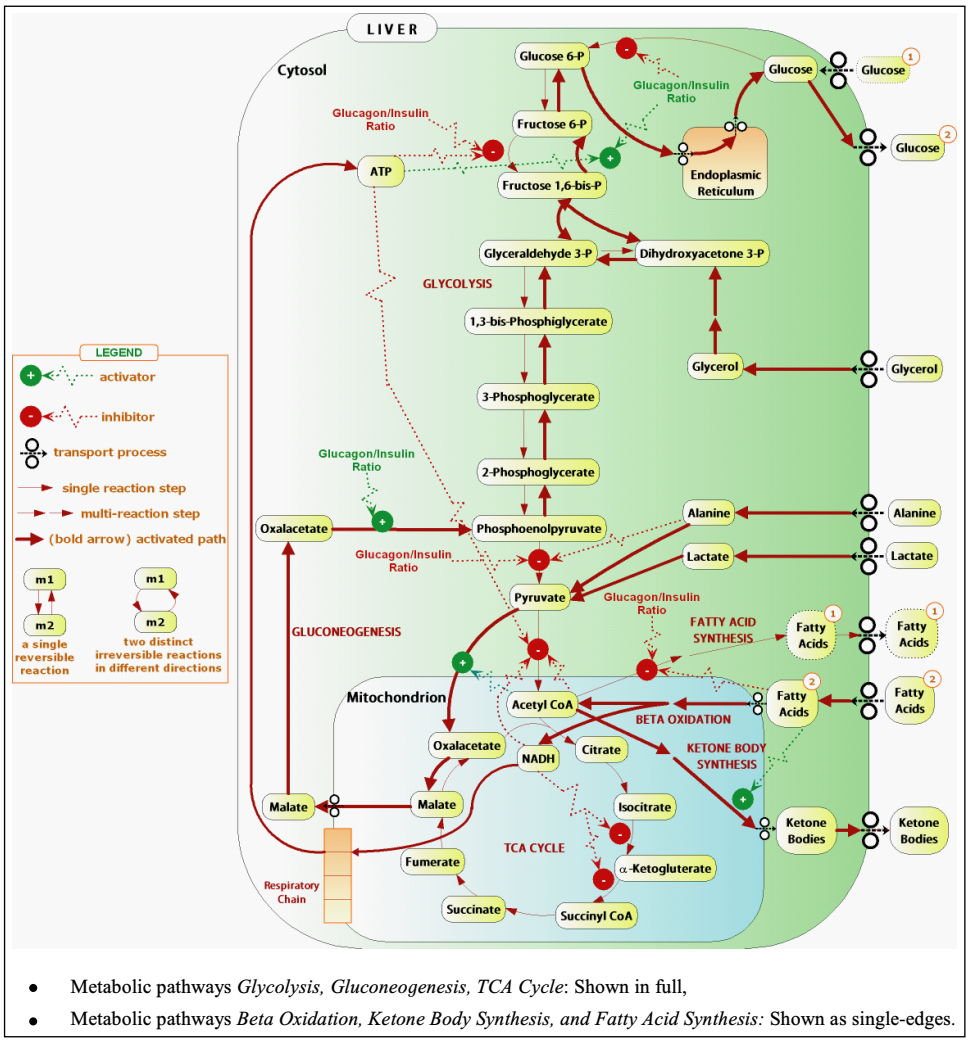
1. Dietary state(s) and/or physiological condition(s): *Fasting*
2. Some key metabolite concentration changes (increases/decreases): lactate↑, alanine↑, triglyceride constituents↑ (i.e. fatty acids ↑ and glycerol↑)

(a) **Find** activated/inactivated paths in the metabolic subnetwork that consists of *Glycolysis*, *Gluconeogenesis*, *TCA Cycle*, *Beta Oxidation*, *Ketone Body Synthesis*, *and Fatty Acid Synthesis*
(b) **Visualize** *Glycolysis*, *Gluconeogenesis*, and *TCA cycle* in full form, and the remainder of the pathways in collapsed form
(c) **Explain** the *reasons for blocked reaction directions* in pathways of P.

Query Result:

(a) Paths with increased flux rates (shown as bold edges in Fig. 1).
(b) Collapsed (shown as double-arrow edges in Fig. 1) and full forms of pathways.
(c) Explanations for blocked reaction directions in the selected subnetwork:

- *TCA cycle* is inhibited due to increased *NADH* synthesis in *Beta Oxidation* (shown in Fig. 1 with three NADH inhibitors inhibiting the productions of *Acetyl CoA, α-ketogluterate*, and *Succinyl CoA*).



- Metabolic pathways *Glycolysis, Gluconeogenesis, TCA Cycle*: Shown in full,
- Metabolic pathways *Beta Oxidation, Ketone Body Synthesis, and Fatty Acid Synthesis:* Shown as single-edges.

Fig. 1. A partial human metabolic network in liver.

- *Fatty Acid Synthesis* is inhibited due to: (1) increased concentration of *fatty acids*, and (2) elevated *glucagon/insulin ratio* in fasting state (shown in Fig. 1 as the inhibitor on the double-arrow edge for *Fatty Acid Synthesis*).
- In *Glycolysis*, the regulated enzymes, i.e. *PFK*1 (in Fig. 1, *Fructose* 1,6-*bis-P* → *Fructose* 6-*P*), *pyruvate kinase* (in Fig. 1, *Phosphoenolpyruvate* → *Pyruvate*), and *glucokinase* (in Fig. 1, *Glucose* → *Glucose* 6-*P*)) are inhibited by elevated *ATP*, *alanine*, and *glucagon/insulin ratio*.
- *Pyruvate dehydrogenase* (*PDH;* in Fig. 1, *Pyruvate* → *Acetyl CoA*) is inhibited by increased *NADH*, *acetyl CoA*, and *ATP* production.

In Fig. 1, we visually specify the running metabolic (sub)network used in this paper, with most of the examples directly employing the network of Fig. 1. Moreover, we reduce the visual complexity of Fig. 1 as follows.

- For each enzymatic reaction of Fig. 1, neither the name of the enzyme, nor its EC (enzyme commission) number is shown.
- For each reaction of Fig. 1, none of the co-factors, regulators, activators, and inhibitors is shown, unless specifically used in an example.
- For each of the three fully drawn metabolic pathways (i.e. *Glycolysis*, *Gluconeogenesis*, and the *TCA cycle*), pathway boundaries are not explicitly specified.

However, in the rest of the paper, whenever we refer to an enzyme, we explicitly specify the enzyme's position in Fig. 1 via the edge from its visually specified substrate molecule to its visually specified product molecule.

1.2. *Assumptions for MQL environment*

We make the following assumptions about the *MQL* query processing environment/model.

- A complete metabolic network is pre-captured and available in a metabolic network database.
- The metabolic network database captures tissue-level compartmentalization; that is, it is a multi-tissue (i.e. not a single cell) and a multi-compartment (such as *cytosol* and *mitochondrion*) environment; and
- The organism (represented by its metabolic network database) is queried in a *quasi-steady state*; that is, at query time, the rate of formation of every metabolite is equal to its rate of degradation, i.e. all concentrations remain constant in time.

1.3. *Contributions of this paper*

Contributions of this paper are

- To computationally capture and model mammalian metabolic networks by employing the underlying biochemistry principles,
- Design of the metabolism query templates MQL_{AIP} and MQL_{PFC} which allow in-depth biochemical queries,
- Development of query processing strategies for MQL_{AIP} and MQL_{PFC} queries.

1.4. *Overview of related work and coverage*

With the goal of creating querying environment for metabolism, in the literature, researchers present a number of specifications and tools in mainly three categories. The first category of works includes biological simulation and modeling systems (e.g. BioSim,[2] GenSim,[3]), the second category consists of query languages (e.g. PQL,[4] bcnQL,[5]) on biochemical networks, and the third category includes well-known

metabolic data sources (e.g. KEGG,[6] MetaCyc,[7] Reactome,[8] PathCase[9]) with advanced querying interfaces running on biochemical network databases. Such systems, query languages, and metabolic data sources essentially view the metabolism as a graph, and mostly focus on querying (i) structural properties of metabolic networks (e.g. paths, neighborhoods, cycles, etc.), and (ii) entity relationships (e.g. inhibitors of a reaction in a particular pathway). However, these efforts do not capture detailed biochemical working principles of a metabolism, and interrelationships between pathways under different physiological and dietary states. We compare the above-listed work briefly in Sec. 6.

Our proposed *MQL* approach, with its goal of identifying active/inactive paths in a metabolic network, can be viewed as being in the general category of metabolic analysis techniques which include *metabolic control analysis* (MCA)[10–13] *flux balance analysis* (FBA),[14–16] *metabolic flux analysis*,[17] and *metabolic pathway analysis* (i.e. *elementary flux mode analysis* and *extreme pathways*).[15,18–20] In Sec. 6.1, we summarize and compare these techniques with *MQL*.

The rest of the paper is organized as follows. In Sec. 2, we characterize the essential principles of the mammalian metabolism, and accordingly formulate *MQL*'s query processing rules. Section 3 discusses *MQL*'s data and graph representation models in association with the corresponding biochemical principles. In Sec. 4, we specify the syntax for MQL_{AIP} and MQL_{PFC} queries. Section 5 organizes the query processing rules of Section into a query processing framework for MQL_{AIP} and MQL_{PFC} queries. In Sec. 6, we discuss and compare the related studies, and Sec. 7 concludes.

2. Biochemistry-Based Query Processing Principles

The overall mammalian metabolism consists of individual metabolic pathways and processes. Adaptation of the metabolism to changes in physiological conditions as well as the dietary state of the body postulate an efficient management of variety of pathways with the same grand vision, e.g. energy production/storage, handling stress, and so on. In order to achieve a harmonious action of different pathways in an efficient way, the human metabolism employs a variety of control mechanisms that determine rates of individual pathways. Some of these control mechanisms operate at a very coarse level, while some others perform "finer tuning" at different critical points in the metabolic network.

The processing of an *MQL* query such as the one in Sec. 1.1 requires the use of underlying metabolic biochemistry principles that control the functioning of pathways, which we characterize next. This section lists a number of metabolic biochemistry principles, and converts them into the corresponding query processing rules (*QPR*s). We think that the characterization in this section is only the first step; and that the characterization/use of more and more refined/in-depth biochemistry principles will result in the future with more accurate and realistic *MQL* query results.

In what follows, we pair a biochemistry principle i (sometimes with multiple components a-d) with its corresponding *MQL* query processing rule *QPR* i. Section 2.1 characterizes the *substrate availability* notion for a reaction, and translates it into *QPR* 1 involving product availability. In Sec. 2.2, we discuss three different types of enzyme regulations, namely, *allosteric regulation, covalent modifications*, and *enzyme synthesis and degradation*, and derive the corresponding very basic query processing rule *QPR* 2. Section 2.3 introduces the notion of *regulator precedence* (which is textbook knowledge for biochemists), and the corresponding *QPR* 3. In Sec. 2.4, we discuss *pathway level regulation*, namely, the notions of *regulatory*, *rate-controlling*, *and committed steps* of pathways, and present the corresponding *QPRs* $4 - 6$. Section 2.5 introduces the notion of a *metabolite pool* (or pools) for a metabolite in a biological compartment, presents five biochemistry principles about the effects of reaction rates on pool sizes, the notion of *pool hierarchies*, etc, and translates these rules into *QPRs* 7–11. In Sec. 2.6, we discuss *energy currency metabolites* in a cell, and modeling energy pool of a cell hierarchically, and derive the corresponding query processing rule *QPR* 12 about determining the *energy state of a cell*. Section 2.7 specifies four principles for characterizing the *functional specialization of biological compartments* (namely, *enzyme specialization/ isoforms*, *transport processes*, and *inputs/outputs of compartments*), and derives four query processing rules *QPR* 13–16. In Sec. 2.8, we characterize *metabolite availability/accumulation* at steady state via two principles, and derive the corresponding query processing rules *QPR* 17–18. Section 2.9 introduces the notion of a s*ignature for dietary states and physiological conditions*, and translates it into *QPR* 19. Finally, Sec. 2.10 discusses the notion of *product inhibition* (that occurs when a product metabolite pool size increases substantially and slows down the reaction rate) and the corresponding *QPR* 20.

2.1. *Substrate availability*

Principle 1. The availability of substrates for a particular pathway is a major driving factor that controls the rate of metabolic processes and biochemical reactions (Ref. 21, p. 863).

The liver provides many examples illustrating such substrate-driven control. As an example, concentration of *fatty acids* entering liver from blood determines the rate of *ketone body synthesis*. As another example, availability of substrates for glucose synthesis is a major control factor that increases the rate of *gluconeogenesis*. As yet another example, increased ammonia production leads to a higher rate of urea production through the urea cycle. More specifically, dietary amino acids are absorbed by the gut, and released as *citruline* into the portal vein. *Citruline* is a precursor of *ornithine* in liver, where increased *ornithine* concentration causes a higher rate of urea cycle function.

In processing *MQL* queries, principle 1 is employed using the following rule:

Query Processing Rule 1: If a substrate concentration increases (decreases), in the absence of other factors, the product concentration also increases (decreases).

Cofactors are small metabolites that bind enzymes and are necessary for biochemical reactions to occur (Ref. 21, p. 378). They may be modified during a reaction, or can go unmodified. A *cofactor in* metabolite is converted to a *cofactor out* metabolite during a reaction. Cofactor-in and cofactor-out metabolites are considered as specialized substrates and products, respectively.

2.2. *Regulation of key enzymes*

2.2.1. *Allosteric regulation*

Principle 2(a). Many enzymes have a distinct regulation site, called allosteric site (which is different than the active site of the enzyme) that provides a base for the binding of effector molecules, called allosteric regulators. Allosteric regulation may involve both inhibition (i.e. a decrease in the rate of pathway) and activation (i.e. an increase in the rate of a pathway).

Through allosteric regulation, the metabolism gains the capability for finer tuning than just substrate availability, and, as a result, many futile cycles, which would otherwise waste resources, are prevented in different metabolic processes. (Futile cycle is defined qualitatively as "*two opposing sets of enzyme-catalyzed reactions that result in release of energy as heat by the net hydrolysis of ATP*" (Ref. 21, p. 1142). We give an example.

Example 2.1: *Fructose* 2, 6-*bisphosphate* (not shown in Fig. 1) allosterically activates the enzyme *phosphofructokinase*-1 (*PFK*1; in Fig. 1, *Fructose* 1, 6-*bis-P* $\rightarrow$ *Fructose* 6-*P*), and at the same time, allosterically inhibits the enzyme *fructose* 1, 6-*bisphosphatase* (in Fig. 1, *Fructose* 6-*P* $\rightarrow$ *Fructose* 1, 6-*bis-P*). These two parallel regulations stimulate *glycolysis* while inhibiting *gluconeogenesis*, and this helps prevent a futile cycle between *fructose* 6-*phosphate* and *fructose* 1, 6 *bisphosphate*. In summary, allosteric regulators of reactions can be used to infer the activation/inhibition of reactions.

2.2.2. *Covalent modifications*

Covalent modification establishes a bridge between signaling pathways and metabolic pathways through phosphorylation of enzymes at the end of cascades of signaling steps which are often initiated by extra-cellular agents, such as hormones.

Principle 2(b). Through covalent modification, enzymes are phosphorylated or dephosphorylated by enzyme-specific kinases and phosphatases, respectively. Depending on the enzyme, either its phosphorylated or dephosphorylated form is active and available to catalyze the corresponding reaction.

Example 2.2. *Pyruvate dehydrogenase* (*PDH;* in Fig. 1, *Pyruvate* → *Acetyl CoA*) is inactivated when it is phosphorylated by *protein kinase*. *Protein kinase* is activated by *Acetyl CoA*, *NADH*, and *ATP*, while it is inhibited by *Pyruvate*. Hence, *Acetyl CoA*, *NADH*, and *ATP* act as inhibitors of *PDH*, and *Pyruvate* acts as activator of *PDH*.

2.2.3. *Regulation through enzyme synthesis/degradation*

Another regulation mechanism for a reaction is the changes in the concentration of its catalyzing enzyme. The $V_{\max}$ notion in the Michaelis-Menten equation (Ref. 21, p. 388) represents the maximum speed that a reaction can achieve given that there is unlimited amount of substrate(s). At the maximum speed, reaction rate levels off, since all available enzymes are saturated with the available substrate(s). Hence, we have the following principles.

Principle 2(c). Increase or decrease in the synthesis or degradation rates of an enzyme may change the flux rate going through that reaction.

Principle 2(d). Depending on the mechanism regulating the enzyme activity, the time required to observe a change in enzyme activity changes. More specifically, allosteric effects take place immediately; covalent modification may require minutes; and regulation through gene expression may require hours to days (Ref. 22, p. 64). (In either case, *MQL* assumes a stable steady state environment, after all effects have taken place).

Query Processing Rule 2(a): Capture in the database the type of mechanism for each regulator. In the case where multiple regulators with different mechanisms are in effect, consider the dominant regulator as controlling the regulation.

Query Processing Rule 2(b): If an activator (of any type) increases (decreases), in the absence of other factors, the reaction rate increases (decreases). And, if an inhibitor (of any type) increases (decreases), the reaction rate decreases (increases).

2.3. *Regulator precedence*

Key enzymes of the metabolism often have multiple (i.e. more than one) activators and inhibitors. And, it is not rare that an enzyme may simultaneously be acted upon by two regulators with conflicting effects (i.e. simultaneous actions of an inhibitor and an activator). In such cases, usually one of the regulators takes precedence over the other(s), and the final effect (i.e. inhibition or activation) on the working mechanism of the enzyme is shaped by that regulator. We give an example.

Example 2.3: Consider the regulator precedence that takes place during the fasting state of the body where both *beta-oxidation* (shown collapsed in Fig. 1) and *gluconeogenesis* pathways (shown in full in Fig. 1) have increased flux rates in liver. A key enzyme, control of which diverts *pyruvate* into *gluconeogenesis*, rather than

into the *TCA cycle*, is *pyruvate dehydrogenase* (*PDH*; in Fig. 1, *Pyruvate* → *Acetyl CoA*) of *mitochondrion*. *Pyruvate* is a homotropic allosteric regulator (i.e. it is both an allosteric activator and a substrate) of *PDH*. And *Acetyl-CoA* is an inhibitor of *PDH*. During the fasting state, concentrations of both *Pyruvate* and *Acetyl CoA* increase. Therefore, *PDH* is under the simultaneous allosteric effect of *Pyruvate* and *Acetyl-CoA*. However, *Acetyl-CoA* takes the precedence to determine the final effect on *PDH*, and the activity of *PDH* is inhibited under this state.

Principle 3. An enzyme may have multiple regulators which control the enzyme rate with varying degrees of effect.

Query Processing Rule 3: In cases where multiple regulators with conflicting regulatory effects (activation or inhibition) on an enzyme are in place (with this regulation information precaptured and available in the database at query time), employ the regulator with the strongest effect (highest precedence) on the enzyme, and ignore the other regulators. If no precedence value is available in the database, apply Query Processing Rule 2(a).

2.4. *Pathway-level regulation*

Most of the time, only a subset of reactions in a pathway is subject to regulation (i.e. regulatory steps), and the others simply follow the regulated reactions.

Principle 4. In order for a pathway to have increased flux (or be activated), none of its regulatory points should be inhibited.

Example 2.4: In Fig. 1, the *TCA cycle* is marked for decreased rate due to the inhibition of its regulated reactions (also called regulatory points) by *NADH*.

Query Processing Rule 4: Do not mark a pathway completely active if at least one of its regulatory steps (which are pre-captured and available in the database) is inhibited.

The regulated reactions in a pathway may be further classified as *rate-limiting* and *committed* steps.

Principle 5. If a given enzymatic reaction in a pathway forms a *rate-limiting* step, increasing the concentration of the enzyme of this reaction increases the overall rate of the pathway. Similarly, decreasing the concentration of the enzyme of the rate limiting step decreases the overall rate of the pathway.

As an example, the rate limiting step of *Glycolysis* is *phoshofructokinase* 1 (*PFK*1). A pathway rarely has more than one rate limiting step. Moreover, a rate-limiting step is usually irreversible and the slowest step in a given pathway. Thus, inhibition of the rate-limiting step usually leads to the accumulation of precursors of the pathway.

Query Processing Rule 5: Set upper-limit for a pathway's activity level as the activity level of its slowest rate-limiting step (if there is more than one).

Principle 6. Once the *committed step* takes place, other reactions in the pathway follow this reaction until the end-product is produced, provided that none of the other regulated processes are blocked or inhibited.

A committed step of a pathway is usually one of the early irreversible reactions in the pathway. As an example, in glycolysis, the committed step is the same as the rate-limiting step, *PFK*1.

Query Processing Rule 6: If the committed step of a pathway is blocked (i.e. inactive), then mark the pathway as inactive.

2.5. *Metabolite pools*

Due to the integrative and highly connected nature of the human metabolic network, key metabolites usually have multiple producer and consumer metabolic processes.

Principle 7(a). Conceptually, each metabolite m is considered to have a pool of its own that feeds the consumer processes of m, and also serves as a sink collecting m from the producer processes of m.

Principle 7(b). In a particular biological compartment, the total amount of a metabolite m in terms of molar mass (i.e. the mass of one mole of a metabolite counts as size 1) is often expressed as its *pool size*.

Query Processing Rule 7: Capture a metabolite as a collection of pools with a default pool identified in each compartment.

Principle 8. Due to biological compartmentalization (Ref. 21, p. 405) as well as different functional roles,[23] a metabolite may have more than one pool.

Query Processing Rule 8(a). Associate each reaction with particular pools of its substrates, products, and regulators.

Example 2.5: *Malate* is located in both *cytosol* and *mitochondrion*. Hence, it has separate pools for each biological compartment that it resides. And, *Malonyl CoA* is reported[23] to possibly have two pools in *cytosol* due to distinct functional roles as substrate and regulator.

Query Processing Rule 8(b). Connect two reactions, r_1 and r_2 to each other in the metabolic network if (i) r_1 and r_2 has at least one shared metabolite m, and (ii) r_1 and r_2 are associated with the same pool of m.

Principle 9. The relative contribution of each metabolic reaction into a specific metabolite pool may differ from one metabolic reaction to another. Similarly, the relative consumption by each metabolic reaction that is fed from the same pool may be distinct.

Example 2.6: In the fasting state, in liver mitochondrion, the *Acetyl CoA* pool is minimally consumed by *TCA Cycle*, and the majority of it is used by *Ketone Body Synthesis* (Ref. 21, p. 856).

Query Processing Rule 9: While the change in a particular metabolite pool size is computed, take into consideration rates of producers and consumers. If rates are available only in terms of qualitative terms (e.g. major, minor, etc.), then map such qualitative terms to consistent quantitative values — only for comparison purposes (e.g. major = 90, minor = 10).

Principle 10. The total amount of a particular metabolite m in a biological compartment c at any time is stated as the combined size of metabolic pools for m in c. Different pools of the same metabolite may have different sizes. Hence, proportional pool sizes should also be accommodated in a computational model.

Example 2.7: In the fasting state, protein turnover in the muscle increases, and aminoacids (AA) are transported from muscle to liver through blood to be used as substrates for *Gluconeogenesis*. In blood, each metabolite can be considered to have its own pool. Due to conversion of many aminoacids to *alanine* and *glutamate* in liver, the amounts (pool sizes) of different aminoacids released from muscle into blood are not the same (e.g. *alanine* and *glutamine* accounts for 80% of all AAs in the blood). Whenever total AA concentration in blood is in question, total pool size of AAs is considered by summing up the individual AA pools.

Query Processing Rule 10: While the change in the overall concentration of a metabolite m in a given compartment is computed, take into consideration relative sizes of each pool of m. That is, let $p1, p2, \ldots, p3$ be the pools of a metabolite m with sizes $s1, s2, \ldots, s3$, respectively, for a given compartment. Then, if the total size of all pools of m which are marked for increase (decrease) is larger than the total size of all pools of m which are marked for decrease (increase), then we conclude that the overall concentration change of m increases (decreases).

Principle 11. (*metabolite pool hierarchy*) Higher level metabolic reasoning involves the creation of conceptual metabolite pool hierarchies.

Example 2.8: The total available free aminoacid (AA) content is considered as the sum of the pool sizes of each individual amino acid, such as the *alanine* pool, the *glutamate* pool, etc. Hence, the conceptual AA pool is a parent pool of those individual amino acid pools. The overall AA pool size affects the direction of reversible *aminotransferase* reactions (Ref. 21, p. 746).

Query Processing Rule 11: (*Computing a metabolite concentration change in the presence of a metabolite pool hierarchy*): If pools of a metabolite m are organized in a hierarchical manner, then, while computing the overall concentration change of m, take into consideration only the leaf level pools of m in its pool hierarchy.

2.6. *Energy state of cells*

The metabolism makes an effort to maintain energy homeostasis within cells. Such a maintenance effort involves adaptive regulation of energy producing (e.g. the *TCA Cycle*) and storing processes (e.g. *Fatty Acid Synthesis*).

Example 2.9: In low energy state, the rates of *AA degradation*, *Lipolysis, Fatty Acid Oxidation, Glycogenolysis*, and the *TCA Cycle* increase. On the other hand, in high energy state, the rates of such processes decrease, and the rates of opposite processes such as *Fatty Acid Synthesis* and *Glycogenesis* increase.

Definition. (*Energy Currency Metabolite*): Certain metabolites with high energy electrons or high energy phosphate bonds serve as carriers of energy in the body. Such molecules are considered as "energy currency" of cells. That is, the energy state of the cell can be quantified by considering the amounts of such high-energy molecules. Most common energy currency molecules are *ATP, NADH, NADPH,* $FADH_2$, and *GTP* (and their oxidized or dephosphorylated forms, i.e. *AMP*, *ADP*, *NAD+*, *NADP+*, *FAD*, and *GDP*).

Principle 12. Energy pool of a cell can be modeled hierarchically where the overall energy pool of the cell is the parent of all pools of energy-currency molecules.

Query Processing Rule 12: Determine the energy state of the metabolism based on the overall change in individual ratios of energy currency metabolites to their oxidized or dephosphorylated peers, e.g. $\frac{NADH}{NAD+}$, $\frac{ATP}{AMP}$, and so on. For a ratio to increase, concentration of nominator metabolite should increase while concentration of denominator metabolite decreases. And, for a ratio to decrease, concentration of nominator should decrease while concentration of denominator increases. In cases where both denominator and nominator change in the same direction, no assessment is made.

2.7. *Functional specialization of biological compartments*

Principle 13. Enzymes in pathways of the human metabolism are highly specialized in terms of biological compartments (e.g. organs, organelles, membranes, etc.) that they reside in.

Example 2.10: Urea cycle takes place only in the liver. And, ketone bodies are produced by the liver for the peripheral tissues, but cannot be used by the liver itself as an energy source. Moreover, both muscle and liver contain the *Glycolysis* pathways, but the muscle version lacks the enzyme (*Glucose* 6-*Phosphatase*) that converts *glucose* 6-*phosphate* to *glucose*, thus, preventing muscle from directly contributing to the blood glucose level.

Query Processing Rule 13: Identify enzymes by their biological compartments, and consider their isoforms in different compartments as distinct entities.

Principle 14. Each biological compartment produces and/or consumes a certain set of metabolites.

Query Processing Rule 14: Associate each compartment with particular pools of metabolites as its input and output. And, connect two compartments in the metabolic network if they have at least one shared input and/or output metabolite pool.

Principle 15. Metabolites are transported into organs from blood or from organs into blood through (i) *complex* transport processes that are regulated by complex enzyme/hormone mechanisms, (ii) *simple* transport processes in which metabolites follow the concentration gradient, that is, they flow from high concentration compartment to low concentration compartment.

Example 2.11: *Glucose* uptake into muscle and adipose tissue is made possible through an active transport mechanism mediated by insulin with transported proteins $GLUT4$. However, glucose uptake by liver is not regulated by insulin.

Query Processing Rule 15: Associate each input and/or output metabolite of a compartment with a transport process. Consider a transport process as a metabolic reaction with regulator(s) (if any) (precaptured and modeled in the database). Connect a transport process and an enzymatic metabolic reaction if they share at least one metabolite pool (i.e. as their substrate and/or product).

Principle 16. Biological compartments are often organized in a hierarchical manner, where one compartment contains another compartment.

Example 2.12: Liver contains mitochondrion which in turn contains mitochondrial matrix.

Query Processing Rule 16: Whenever a compartment is specified in a query, automatically include all of its child compartments (if any) in the query. In terms of visualization, consider blood to be the parent of all compartments (i.e. root in the compartment hierarchy); but, during query processing, consider blood as a separate compartment with no parent or child.

2.8. *Metabolite availability vs. metabolite accumulation*

The availability of a metabolite and its accumulation are often (mistakenly) considered to refer to the same concept. However, metabolically, they mean related, but different things.

Principle 17. Availability of a metabolite m as a substrate to a metabolic process p means that, as long as p is not inactivated by some other regulatory mechanisms, p is supplied some amount of m to consume. On the other hand, in order for a metabolite m to accumulate in a particular pool of its own, the rate of m's overall consumption should be less than the rate of its overall production. In other words, at steady state, accumulation of a metabolite implies its availability, but its availability does not necessarily imply its accumulation.

Example 2.13: In the fed state, *glucose*, through *Glycolysis*, is catabolized to *Acetyl CoA*, which is converted to*fatty acids* or oxidized in the *TCA Cycle*. Although

Acetyl CoA is available to these metabolic processes (i.e. *Fatty Acid Synthesis* and the *TCA Cycle*), it does not accumulate, as the combined consumption rate by *Fatty Acid Synthesis* and the *TCA Cycle* is the same as (or larger than) its production by *Glycolysis*. On the other hand, in the fasting state, *Acetyl CoA* is produced by *Beta Oxidation*, and consumed by the *TCA Cycle* and *Ketone Body Synthesis*. In this case, accumulation of *Acetyl CoA* occurs, since its production rate by *Beta Oxidation* is much higher than its combined consumption rate by the *TCA Cycle* and *Ketone Body Synthesis*.

Query Processing Rule 17(a). (*metabolite accumulation and availability*): Given a metabolite pool m, let (i) $P_m = \{(p_1, s_1), (p_2, s_2), \ldots, (p_i, s_i)\}$ be the activated producer set of m, where each pair (p_i, s_i) refers to an activated producer p_i of m and its rate s_i, (ii) $C_m = \{(c_1, r_1), (c_2, r_2), \ldots, (c_j, r_j)\}$ be the activated consumer set of m, where (c_j, r_j) refers to an activated consumer c_j of m and its rate r_j, (iii) $(s_1 + s_2 + \cdots + s_i)$ be the *ProductionRate(m)* of m, and (iv) $(c_1 + c_2 + \cdots + c_j)$ be the *ConsumptionRate(m)* of m. Then, mark m as

- "*severely accumulated*" if $ProductionRate(m) > 0$ and $ConsumptionRate(m) = 0$ (i.e. all consumers are inactive).
- "*accumulated*" if $ProductionRate(m) > ConsumptionRate(m)$.
- "*available*" if $ProductionRate(m) \sim= ConsumptionRate(m)$ and $P_m \neq \emptyset$.
- "*unavailable*" if $ProductionRate(m) = 0$.

To be used during the query processing, we also assign an integer id for each of the above qualifiers as follows: 0: *unavailable*, 1: *available*, 2: *accumulated*, 3: *severely accumulated*.

Query Processing Rule 17(b). (*Metabolite accumulation and availability in the presence of metabolite pool hierarchy*): Given a metabolite pool m with child metabolite pool set C in a hierarchical metabolite pool organization, mark m as

- "*severely accumulated*" if there is at least one metabolite pool $p \in C$, which is marked as "*severely accumulated*," else
- "*accumulated*" if there is at least one metabolite pool $p \in C$, which is marked as "*accumulated*," else
- "*available*" if there is at least one metabolite pool $p \in C$, which is marked as "*available*," else
- "*unavailable*" in all other cases.

(We consider the above conditions in the given order, and stop considering the remaining conditions whenever a matching condition is identified)

Principle 18. Let m be a metabolite pool involved in a set R of reactions as a substrate and/or regulator. For some reactions in R, availability of m may be sufficient for them to be active through substrate availability (provided that there are no other inhibiting mechanisms) or experience the regulating effect (i.e. inhibition/activation) of m, in those cases where m is a regulator. However, some other

reactions may require the accumulation of m (at least, at moderate levels) to assume substrate availability for activation or to experience the regulating effect of m, in cases m is a regulator.

Example 2.14: *Acetyl CoA* is an allosteric activator of the first step (also the *committed step*) in *Gluconeogenesis*, which is catalyzed by *pyruvate carboxylase*. In fed state, *Acetyl CoA* is produced by *Glycolysis* (hence, available), but it does not accumulate (please see Example 2.13). Therefore, *pyruvate carboxylase* is not activated, which leads to the inactivation of *Gluconeogenesis* pathway.

Query Processing Rule 18: As part of association between a metabolite pool and a reaction either as a substrate or product, precapture the "trigger" condition (i.e. either *accumulation* or *availability*) that is necessary for the metabolite to participate in the reaction. If such information is not available, set the trigger condition by default as "*available*" for substrates, and "*accumulation*" for regulators based on the patterns observed in well-characterized metabolic scenarios (e.g. *citrate* as a regulator for *Glycolysis* and *Fatty Acid Synthesis* (Ref. 21, p. 864)). During query processing, check these trigger conditions to make sure that they are satisfied.

2.9. *Signatures for dietary states and physiological conditions*

Principle 19. In different physiological and dietary states, concentration and/or production rate of certain molecules increase or decrease. Hence, such changes can be considered as *signatures* that identify the corresponding physiological state.

Example 2.15: In the fasting state, the body uses *fatty acids* that are mobilized from adipose tissue as the primary energy source due to lack of sufficient *glucose*. *Glycerol* is also released from adipose as the other product of increased *lypolysis activity*. Furthermore, ketone bodies are produced intensively from fatty acids by the liver. At the hormonal level, body's primary response consists of lowering the *insulin* level and increasing *glucagon* levels. Based on this description, the fasting state can be represented as the following set S of concentration changes:

$$\mathrm{S} = \{\text{insulin} \downarrow, \text{glucagon} \uparrow, \text{glucose} \downarrow, \text{fatty acids} \uparrow, \text{ketone bodies} \uparrow, \text{glycerol} \uparrow\}.$$

Apart from concentration changes, the rate of certain metabolic pathways may significantly change leading to accumulation or availability pattern changes of certain metabolites.

Example 2.16: In exercise state, the rate of *Glycolysis* in muscle cells increases significantly to account for increased demand for energy. However, the *TCA Cycle* cannot keep up with this increase in the rate of *Glycolysis*, and *pyruvate* starts accumulating, and is channeled to *lactate*. Hence, the signature of the exercise state involves an increase in the contributing rate of *Glycolysis* into the *pyruvate* pool in cytosol of muscle cells.

Query Processing Rule 19: Whenever a user query involves a built-in dietary state or physiological condition predicate, (i) map these predicates to their corresponding signatures based on their definitions, (ii) consider concentration changes as user-provided concentration change input, and (iii) override pool contribution/consumption rates in the database with those rate changes included in a signature.

2.10. *Product inhibition*

Due to similarities in the way they bind to enzymes, substrates are in competition with products to bind to their enzymes. As the concentration of products increases, this competition slows down the rate of enzymes binding the substrates. Hence, the reaction rate decreases. Eventually, when the product accumulation reaches to high levels, the corresponding reaction is inhibited dramatically.

Principle 20. As the product concentration in the environment increases, the reaction rate slows down.

Example 2.17. In the fasting state, due to the slowdown of the *TCA Cycle* and inhibition of *fatty acid synthesis* in liver, *citrate* accumulates, which in turn inhibits and slows down the primary reaction that produces it (i.e. *citrate synthase*). And, the inhibition of *citrate synthase* leads to the accumulation of its driving substrate, *Acetyl CoA*.

In this work, we take a conservative approach on product inhibition by applying it when a product has no active consumer, i.e. "severely accumulated."

Query Processing Rule 20: Whenever a metabolite m is marked as "*severely accumulated*," mark those reactions that produce m as "*inactive*."

3. Data and Graph Representation Model for Metabolic Network

3.1. *Data model*

We adopt an object oriented data model to represent the mammalian metabolism. Objects are structured data types which contain basic types (e.g. string, int, etc.) or other structured data types (i.e. objects) as their fields. We employ the metabolic principles that are summarized in Sec. 2 as the main motivation, and as a guide in our modeling effort. Figure 2 shows the object definitions and their fields for the essential constituents of the metabolism, where id fields are omitted for brevity. In our data model, metabolism, at the highest level, consists of a set of pathways. Each pathway contains a collection of reactions, a set of substrates, a set of products, and a set of cofactors. To satisfy principle 1, we explicitly specify pathway inputs to prepare the infrastructure required for implementing the corresponding query processing rule in the query processing stage. Optionally, pathways may have committed and rate-limiting steps (if known). Moreover, to satisfy principles 5 and 6, we model committed and rate-limiting steps. Input and output molecules of a pathway are associated with a particular pool of a metabolite through *MetabolitePoolLink*

Pathway {
  name: string
  reactions: [ReactionStep]
  (rateLimitingSteps: [ReactionStep])
  (committedStep: ReactionStep)
  substrates: [MetabolitePoolLink]
  (products: [MetabolitePooLink])
  (cofactorsIn:[MetabolitePoolLink])
  (cofactorsOut:[MetabolitePoolLink])
}

ReactionStep {
  reaction: Reaction
  compartment: Compartment
  (direction: <forward | backward>)
}

Reaction {
  name: string
  substrates: [MetabolitePoolLink]
  products: [MetabolitePoolLink]
  (cofactorsIn:[MetabolitePoolLink])
  (cofactorsOut:[MetabolitePoolLink])
  (enzymes: [EnzymeInstance])
  (inhibitors: [Regulator])
  (activators: [Regulator])
  isTransportProcess: boolean
  isReversible: boolean
}

MetabolitePool {
  metabolite: Metabolite
  compartment: Compartment
  (name: string)
  (size: int)
  (parent: MetabolitePool)
}

MetabolitePoolLink {
  pool: MetabolitePool
  rate: float
  triggerCondition: <accumulation | availability>
  stoichiometry: float
}

Metabolite {
  name: string
  type: string
  defaultPools: [MetabolitePool]
}

EnergyCurrencyMetabolite *extends* **Metabolite** {
  peer: EnergyCurrencyMetabolite
  chargeStatus: <highEnergy | lowEnergy>
}

Regulator {
  (regulator: MetabolitePool)
  triggerCondition: <accumulation | availability>
  type: <allosteric | covalent | expressionRegulation>

  (precedence: int)
  (regulatorRatio: $\frac{\text{MetabolitePool1}}{\text{MetabolitePool2}}$)
}

Compartment {
  name: string
  (size: int)
  (parent: Compartment)
  (transportProcesses: [Reaction])
  type:<tissue | organelle | membrane>
}

Enzyme{
  name: string
}

EnzymeInstance{
  enzymeId: int
  compartment: Compartment
}

DietaryState {
  name: string
  concentrationChanges: [ConcentrationChange]
}

ConcentrationChange {
  pool: MetabolitePool
  changeDirection: string
}

PhysiologicalCondition *extends* **DietaryState** {
  (rateChanges: [RateChange])
}

RateChange {
  poolLink: MetabolitePoolLink
  rateChangeAmount: float
}

Notation: **[ ]** denotes an array of objects, < a | b | c > denotes an enumeration, and ( ) represents optional fields in an object definition.

Fig. 2. Object data model.

objects, which are associated with a particular *MetabolitePool* and have a rate field specifying the contribution (or consumption) rate to the overall pool through that link.

A pool of a metabolite has a *location* field, and optional *size* and *parent* fields, where the parent field makes hierarchical organization of pools possible for a particular metabolite. Such an organization allows for the creation of conceptual groupings of metabolite pools. To satisfy Principle 7(a) and the corresponding query processing rule, we have a *MetabolitePool* object for each metabolite. For Principle 7(b), we have a *size* field for each *MetabolitePool* object. For Principle 8, we capture the compartment information for each pool, and allow a metabolite to have more than one pool. For Principles 9 and 17, a *rate* field in each *MetabolitePoolLink* object is introduced. For Principle 10, a *size* field and a reference to its owner metabolite is created in each *MetabolitePool*. For Principle 11, an optional *parent* field is created for a hierarchical organization. For Principle 18 and its corresponding query processing rule, we include *triggerCondition* field in each *MetabolitePoolLink*.

Metabolite object has *name* and *type* fields, where *type* may be "basic molecule," "hormone," "protein," etc. Moreover, each metabolite stores its default pools per biological compartment. This information is required during query processing to associate a metabolite referenced in a query to one the specific pools of its own, if it has multiple pools in a biological compartment. Hormones have a single pool in blood, but they influence a large number of tissues through cascading signaling steps.

EnergyCurrencyMetabolite directly extends from *Metabolite*, and represents those metabolites which are considered to be energy carriers in a cell. An additional *peer* field links an energy metabolite to its reduced (or oxidized) peer, and the *chargeStatus* field describes whether a given metabolite is a *highEnergy* or a *lowEnergy* metabolite. For Principle 12, overall sizes of *EnergyCurrencyMetabolite* pools and their peers can be used to reason about the energy state of a cell.

Each reaction has a collection of substrates, products, cofactors, enzymes, and regulators, all of which (except enzymes) are metabolite pool instances. For Principle 4, we can obtain the reactions of a pathway, and decide which ones are regulated (through regulator field). Since some biochemical reactions are reversible, and in a particular pathway they usually work in one direction, in each pathway they participate, the *direction* information is also stored. Enzymes can reside in multiple compartments (e.g. isozymes). Hence, each reaction is associated with a specific *instance* of a reaction in a particular compartment. The location information for a reaction is implied by the location of its enzymes. For Principle 13, we associate each enzyme with a particular compartment. Compartments have *name*, and optional *size* and *parent* fields. Similar to *MetabolitePools*, *parent* field allows for the definition of a compartment hierarchy (e.g. organ → tissue → organelle → inner membrane). In our data model, an organelle in tissue A is different than the same type of organelle in tissue B (that is, mitochondrion in liver vs. mitochondrion in red blood cells), since the same type of organelles in different tissues may have

different metabolic functions and/or enzyme coverage. For Principle 16, we have a *parent* field in *Compartment* object for a possible hierarchical organization. In addition, each compartment has a set of *transportProcesses* that carry metabolites in and out of the compartment. A transport process is modeled as an instance of a regular reaction, where compartment refers to the one that a particular transport process belongs to, and substrates and products refer to different pools of the same metabolite. For Principle 14, we can figure out input and output metabolites of a compartment based on substrates/products of its transport processes. And, for Principle 15, we can find out if a transport process is complex or simple by checking whether it has regulators or not.

A regulator involves a specific pool of a metabolite and an optional precedence value which is required when multiple regulators with conflicting effects act simultaneously on the rate of the same reaction (the one with the highest precedence value determines the final effect on the rate of a reaction). Besides, a regulator may optionally be defined based on a ratio of metabolite pools (e.g. glucagon/insulin in Fig. 1). Regulator also involves a *triggerCondition* field that differentiates between reactions requiring accumulations of metabolites and those for which availability is sufficient (Principles 17 & 18). Finally, regulators have a *type* field that captures the mechanism (i.e. allosteric, covalent, expression control) of regulation as described by Principles 2(a)-2(d). We model all enzyme regulation mechanisms through *Regulator* objects. For Principle 3, we have an optional *precedence* field for each *Regulator* object.

A *DietaryState* object is the representation of a dietary state (e.g. fasting) and represented by a set of metabolite concentration changes that characterize the dietary state, which collectively represent the "signature" of a dietary state.

A *ConcentrationChange* refers to a specific pool of a particular metabolite and the direction of its concentration change (i.e. increase or decrease). As an example, fasting state can be represented by the following signature involving concentration change objects: $\{\text{insulin} \downarrow, \text{glucagon} \uparrow, \text{glucose} \downarrow, \text{fatty acids} \uparrow, \text{ketone bodies} \uparrow\}$. A P*hysiologicalCondition* stands for a condition or a disease (e.g. diabetes), and it directly extends from the *DietaryState* object, as we reuse the same representation model. In addition, *PhysiologicalCondition* allows for the specification of a set of changes on the shares of different reactions in metabolite pools. By allowing rate changes, we allow representation of physiological conditions where the rate of a metabolic process can increase/decrease significantly, leading to changes in its contribution or consumption rates for a particular metabolic pool. The modified behavior may affect the accumulation or availability of different metabolites.

3.2. *Graph representation model*

In our graph representation model, compartments (e.g. liver in Fig. 1) are modeled as large "super-nodes" which contain subnetworks of the overall metabolism, as well as other compartments (e.g. mitochondrion in liver in Fig. 1). In each subnetwork,

nodes represent metabolite pools (e.g. *Acetyl CoA* in *mitochondrion* in Fig. 1). Reactions are represented as hyper-edges, which connect multiple end-points (i.e. substrates and products) (e.g. the reaction that converts *oxalacetate* and *Acetyl CoA* to *citrate* in the *TCA Cycle* in Fig. 1). Regulation is represented by an edge between a metabolite pool (i.e. a regulator) and a hyper-edge (i.e. a reaction) (e.g. *NADH* as inhibitor for two different reactions in the *TCA cycle* in Fig. 1).

4. Query Specification

In this section, we discuss the formulations of two types of *MQL* queries, namely MQL_{AIP} and MQL_{PFC} queries. For each query type, we present a descriptive English statement, show the corresponding query template in *MQL*, and finally, provide an example. We adopt an *SQL*-like[24] database query specification scheme for *MQL* queries where (i) the *select* clause of the query defines the output, (ii) the *from* clause defines the objects/relations involved in the query, and (iii) the *where* clause specifies additional predicates about compartment, dietary state, and so on.

4.1. *Exploring activated/inactivated paths:* MQL_{AIP} *queries*

MQL_{AIP} queries involve finding activated/inactivated paths in a particular sub-network of the metabolism under specified dietary and/or physiological conditions, and a given set of concentration changes of key metabolites.

Example 4.1: Please see Sec. 1.1 for a sample query of this type.

The generic query template is formulated as shown in Fig. 3 where the notation is as follows.

- Names in italics refer to database values, e.g. *cytosol* or the *TCA-cycle.*
- Names in regular fonts (i.e. non-italic and non-bold-face) refer to variables, e.g. P1, C1, etc.
- Bold-face words refer to keywords of the query language, e.g. **select** or **paths** or **dietaryState**.

select {**activated** | **inactivated** | * (default)} **paths**
from pathways *Pi1* P1 **in** *Ck1*{,*Ck2*..*Ckm*} C1{, *Pi2* P2 **in** *Cr1*{,*Cr2*..*Crm*} C2, .., *Pin* Pn **in** *Ct1*{,*Ct2*..*Ctm*} Cm }
where {**dietaryState** = [**fasting** | **after a meal devoid of AAs** | **after a meal devoid of carbs** | **exercising** | **after alcohol consumption**]}
{**and physiologicalCondition** = *a-set-of-conditions*}
{**and concentrationChanges** = *a-set-of-metabolite concentration*-changes}
{**visualize** {*a subset of input pathways in P1 [, P2, .., Pn]* | * (default)} {**as collapsed** (default) | **as full**}
{**explain** {**inactivated** (default) | **activated** | **regulated**} **reactions** [**in** *a-subset-of-pathways* | *]}

Fig. 3. MQL_{AIP} query template.

- The entry in between brackets, such as [a | b | c], enumerates exactly one possible input value for the corresponding field. That is, [a | b | c] denotes *exactly one of* (a *or* b *or* c).
- The entry in between curly brackets, such as {a|b|c}, enumerates exactly one or zero possible input value for the corresponding field. That is, {a|b|c} denotes *exactly one or zero of* (a *or* b *or* c).
- The parenthesis, ( ), has no particular meaning, and is used solely for grouping purposes to disambiguate query formulations.
- ".." denotes zero or more repetitions.
- "*" stands for the quantifier "all" as in the standard SQL specification of database query languages.[25] In the select clause, using * as path specification computes all activated/inactivated paths.
- If, for a particular field, nothing is specified as part of the query, then default selections are assumed. Default selections are marked as "(default)" in the query template.
- · notation in compartments is used to specify root-to-node path expressions (as in the path expressions of object-oriented query languages[25]) in the compartment hierarchy.
- Visualizing pathways in full/collapsed forms or providing additional explanations about inactivated and/or activated reactions are specified as optional separate clauses. As an example, for visualization, default action is to display a pathway in the query output in collapsed form, provided that no specification is included in the query.
- *a-set-of-conditions* refers to a set of database physiological conditions, such as *diabetes.*
- *a-set-of-metabolite concentration-changes* refers to a set of database metabolite concentration changes, each in a specific compartment.
- *a-subset-of-pathways* refers to a subset of the pathway variables $P1, P2, \ldots, Pn$ specified in the from clause.
- The concentration change direction symbols ↑ and ↓ are replaced with ∧ and ∨, respectively, since arrow symbols are not supported by simple query editors. Similarly, horizontal arrow symbol, →, appearing in some query types is replaced by its simple form --→.

Example 4.2: The following query represents the MQL_{AIP} specification of the query discussed in Sec. 1.1.

select * paths
from pathways *Glycolysis* P1 **in** *liver.cytosol* C1, *Gluconeogenesis* P2 **in** *liver.cytosol* C1, *TCA-Cycle* P3 **in** *liver.cytosol.mitochondrion* C2, *Beta-Oxidation* P4 **in** *liver.cytosol.mitochondrion* C2, *Ketone-Body-Synthesis* P5 **in** *liver.cytosol.mitochondrion* C2, *Fatty-Acid-Synthesis* P6 **in** *liver.cytosol* C1

where dietaryState = fasting
and concentrationChanges = {*lactate* ∧ **in** *blood*, *alanine* ∧ **in** *blood*, *fatty acids* ∧ **in** *blood*, *glycerol* ∧ **in** *blood*}
visualize P1, P2, P3 **as full**
explain blocked reactions in *

4.2. *Exploring potential futile cycles: MQL_{PFC} queries*

This query type involves exploring/verifying potential futile cycles that are prevented, and the control mechanisms that prevent them in a given metabolic setting, such as the one discussed for MQL_{AIP} queries. We give an example for MQL_{PFC} queries.

Example 4.3: Given the same metabolic setting as described in Example 1.2, enumerate (i) potential futile cycles that are prevented by the metabolism, (ii) the regulatory mechanisms that prevent them.

The generic query template for MQL_{PFC} queries extends the template designed for MQL_{AIP} queries. The only required changes are in the "select" clause, as well as in the optional visualization and explanation fields. In the "select" clause, "futileCycle" phrase must be included for this type of queries, and the "regulation" phrase is optional. The control mechanisms preventing potential futile cycles are included in the output only if "regulation" phrase is included in the "select" clause. The semantics of the other parts in the query template are the same as in MQL_{AIP} queries. Finally, the visualization part does not apply to this type of queries, and the explanation part is the only result of this query. Hence, these two parts are not included in the query template.

select futileCycle{, **regulation**}
from pathways *Pi1* P1 **in** *Ck1*{.*Ck2*..*Ckm*} C1{, *Pi2* P2 **in** *Cr1*{.*Cr2*..*Crm*} C2, .., *Pin* Pn **in** *Ct1*{.*Ct2*..*Ctm*} Cm }
where {**dietaryState** = [**fasting** | **after a meal devoid of AAs** | **after a meal devoid of carbs** | **exercising** | **after alcohol consumption**]}
{**and physiologicalCondition** = *a-set-of-conditions*}
{**and concentrationChanges** = *a-set-of-metabolite concentration-changes*}

Fig. 4. MQL_{PFC} query template.

Example 4.4: The *MQL* specification of the query in Example 4.3 is formulated as follows.

select futileCycle, **regulation**
from pathways *Glycolysis* P1 **in** *liver.cytosol* C1, *Gluconeogenesis* P2 **in** *liver.cytosol* C1, *TCA-Cycle* P3 **in** *liver.cytosol.mitochondria* C2, *Beta-Oxidation* P4 **in** *liver.cytosol.mitochondria* C2, *Ketone-Body-Synthesis* P5 **in** *liver.cytosol.mitochondria* C2, *Fatty-Acid-Synthesis* P6 **in** *liver.cytosol* C1

where dietaryState = fasting
and concentrationChanges = {*lactate* ∧ **in** *blood*, *alanine* ∧ **in** *blood*, *fatty acids* ∧ **in** *blood*, *glycerol* ∧ **in** *blood*}

Given the metabolic network of Fig. 1, such a query returns the following result, where superscript M and C stands for mitochondria and cytosol, respectively, and $m_1 \rightarrow^{**} x^{**} \rightarrow m_2$ represents the steps (i.e. reactions) whose inhibition prevents the corresponding futile cycle.

- Potential futile cycle 1: $Pyruvate^M \rightarrow Oxalacetate^M \rightarrow {}^{**}x^{**} \rightarrow Phosphoenolpyruvate^C \rightarrow Pyruvate^C \rightarrow Pyruvate^M$ is prevented **because** the enzyme *pyruvate kinase* for $Phosphoenolpyruvate^C \rightarrow Pyruvate^C$ is inhibited by *NADH*, *ATP*, and *Acetyl CoA*.
- Potential futile cycle 2: *Fructose* 1,6-*bis-P* → *Fructose* 6-*P* →$^{**}x^{**}$→ *Fructose* 1,6-*bis-P* is prevented **because** the enzyme *phoshofructokinase*-1 (*PFK*1) for *Fructose* 6-*P* → *Fructose* 1,6-*bis-P* is inhibited by *ATP* and increased *glucagon/insulin ratio*.
- Potential futile cycle 3: *Glucose*-6-*P* → *Glucose* →$^{**}x^{**}$→ *Glucose*-6-*P* is prevented **because** the enzyme *glucokinase* for *Glucose* → *Glucose*-6-*P* is inhibited by increased *glucagon/insulin ratio*.

5. Query Processing

In this section, we discuss query processing of *MQL*. We employ the biochemical principles of Sec. 2 and the data model of Sec. 3 for processing an *MQL* query and constructing its result set. Related biochemistry principles/*Query Processing Rules* (*QPR*s) are often referenced in parenthesis in italic font. Next, we discuss processing each type of *MQL* query in the order presented in Sec. 4.

5.1. *Preliminaries*

Some parts of the query processing employs an auxiliary data structure, called the *dependency graph*, to properly manage the regulation-wise inter-dependency between different pathways.

5.1.1. *Pathway/reaction dependency graph*

Def'n (*Dependency Graph*): A dependency graph $G(V, E)$ consists of a set V of vertices and a set E of edges, where nodes in V correspond to distinct pathways (or reactions), and a directed edge $e(p_1, p_2)$ represents that pathway/reaction where p_1 (i.e. e.*source*) is dependent on pathway/reaction p_2 (i.e. e.*destination*).

During query processing, dependence graphs are used for two purposes. First, sometimes users may specify a query where not all the pathways that connect

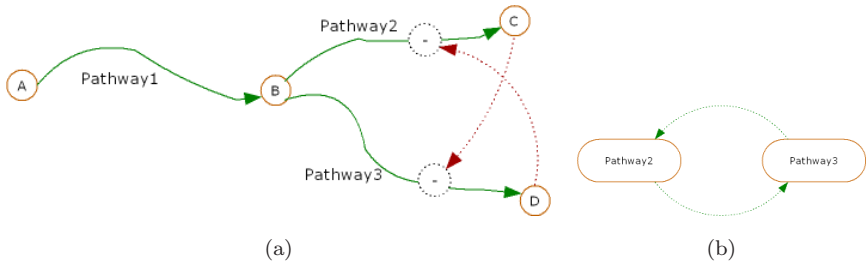


Fig. 5. A Sample query sub-network and the associated dependency graph.

the specified blood metabolites are explicitly named in the query. In such cases, the dependency graph can be used to identify those pathways that are needed to make the query subnetwork a connected graph. In the algorithms of this section, we assume that these "connecting" pathways are pre-identified, and added into the metabolic network of the query (called the *query network*).

Secondly, the dependency graphs are used to identify groups of pathways/reactions with race conditions, and that may impose negative regulatory effects on each other. We illustrate with a sample scenario.

Example 5.1. Consider the sample subnetwork of Fig. 5(a), where edges represent pathways visualized in "collapsed" form, and with A, B, C, and D are the connecting metabolites. Dotted edges denote inhibitor relationships between pathways. In this network, depending on the evaluation of the race condition for the availability of inhibitors, two different query results (in terms of active pathways) can be computed: (i) *Pathway*2 may become activated before *Pathway*3, resulting in metabolite C inhibiting *Pathway*3; thus, *Pathway*1 and *Pathway*2 become active; and *Pathway*3 becomes inactive, or (ii) *Pathway*3 may become activated before *Pathway*2, resulting in metabolite D inhibiting *Pathway*2; thus *Pathway*1 and P*athway*3 become active; *Pathway2* becomes inactive. The query processing algorithm of this section marks both *Pathway*2 and *Pathway*3 as inactive, and places them (in stage 2, step 2.6 of the algorithm in Sec. 5.2) to a special pathway set, namely, P^{sink} for the user to observe. *MQL* detects the cycles in Fig. 5(a) via the dependency graph of Fig. 5(b), and places *Pathway*2 and *Pathway*3 both into P^{sink}.

5.1.2. *Condition-based modeling*

In a cycle of reactions, in the metabolic network, an input to a reaction is produced by the last reaction completing the cycle. In other words, the first reaction cannot be decided to be active (or inactive) unless the last reaction in the cycle is known to be active, and vice versa. If we are to follow the query processing rule 1 which states that all substrates should be available to a reaction in order for this reaction to be

marked as active, none of the reactions in the cycle could be marked as active due to their inter-dependence. Hence, *QPR* 1 needs to be "relaxed" with *condition-based* rules. We give an example.

Example 5.2. Consider the following simple *MQL* query which involves only the *TCA Cycle* of liver in its query subnetwork, and increase in *Acetyl CoA* (Acetyl CoA↑) is provided as part of the query by the user. Note that this is a different activation environment visualized in Fig. 1; for this example, we are only making use of the network of Fig. 1, not the activation/inactivation info visualized there.

select * paths
from pathways *TCA-Cycle* P1 **in** *liver. mitochondrion* C1
where concentrationChanges = {*acetyl CoA* ∧ **in** *liver. mitochondrion*}
explain blocked reactions in *

From Fig. 1, the first reaction (i.e. *citrate synthase*; not shown in Fig. 1) in the *TCA Cycle* requires *Oxalacetate* and *Acetyl CoA* as its input. For this reaction to be active, all of its substrates should be available. Availability of *Acetyl CoA* is already provided as part of the query. However, activation of the reaction that produces *Oxalacetate* (hence the availability of *Oxalacetate*) is cyclically dependent upon the current reaction, *citrate synthase*. And, once the cycle is completed, *Oxalacetate* will be available to *citrate synthase*, as well. Obviously, *QPR* 1 will not allow us to mark *citrate synthase* as active to start the cycle in the first place, and the query result will not contain an active path, which is not correct. Nevertheless, if we assume the *conditional* existence of *Oxalacetate*, and then apply *QPR* 1 to start the cycle first, and, at the end, check if this condition is satisfied by a reaction that produces *Oxalacetate*, we will be able to compute, as the result of the query, that, in this small subnetwork in mitochondrion of liver, the *TCA Cycle*, the only available pathway, is active based on the satisfaction of the query predicates.

The above example illustrates a specific case within a single pathway. However, in general, such cyclic dependences may span multiple pathways, which may not be immediately obvious. Next, we characterize the conditions for reactions (and thus pathways) to be active.

Definition (*Condition*): A condition C is a pair $\langle q, m\rangle$, denoted as $C\langle q, m\rangle$, of metabolite pool status qualifier q (from QPR 17.1), and a metabolite pool m.

Example 5.3. *Ketone Body Synthesis* requires the accumulation of *Acetyl CoA* to use it as a substrate. Then, the required condition can be stated as $C\langle$"*accumulated*," *Acetyl CoA*$\rangle$.

Definition (*Satisfaction of a Condition*): A condition $C\langle q, m\rangle$ is said to be *satisfied* if m is marked with a qualifier q' where either (a) $0 < q \cdot id \leq q' \cdot id$ or (b) $q \cdot id = q \cdot' id = 0$ holds.

Example 5.4. The condition C⟨"*accumulated*," *Acetyl CoA*⟩ is said to be satisfied when the corresponding pool of *Acetyl CoA* has the status "*accumulated*" (id: 2) or "*severely accumulated*" (id: 3).

Definition (*Negation of a Condition*): Negation of a condition $C\langle q, m\rangle$ is denoted as $\neg C\langle q, m\rangle$, and $\neg C\langle q, m\rangle$ is *satisfied*, if m is marked with a qualifier q' such that either (a) $q' \cdot id < q \cdot id$ or (b) $q \cdot id = 0$ and $q' \cdot id > 0$ holds.

Example 5.5. The negation of the condition from Example 5.3, i.e. $\neg C\langle$"*accumulated*," *Acetyl CoA*⟩, is said to be satisfied only when *Acetyl CoA* is marked as "*available*" (id: 1) or "*unavailable*" (id: 0) (i.e. no active producer), and it will not be satisfied by any other qualifier that *Acetyl CoA* may be marked.

Definition (*Conflicting Conditions*): Two conditions $C_1\langle q_1, m\rangle$ and $C_2\langle q_2, m\rangle$ which are defined on the same metabolite m are said to be *in conflict* if there is no possible pool status qualifier for m that would *satisfy* both C_1 and C_2.

Example 5.6. $\neg C_1\langle$"*available*," *Acetyl CoA*⟩ is in conflict with $C_2\langle$"*accumulated*," *Acetyl CoA*⟩

In on our data model, each reaction and pathway involves a set of participating metabolites along with their trigger conditions (i.e. metabolite pool status qualifiers). Then, each reaction (or a pathway) can be considered to be associated with a set of conditions, which are created based on the participating metabolites and their trigger conditions.

Definition (*Condition Set of a Reaction/Pathway*): Condition set of a reaction (or a pathway) r, denoted as $CS(r)$, is constructed as follows.

- For all molecules m such that m is either a substrate, or a cofactor-in, or an activator of r, and t is a triggering status qualifier for m to activate $r, C\langle t, m\rangle \in CS(r)$.
- For all molecules m such that m is an inhibitor of r, and t is a triggering status qualifier for m to inhibit $r, \neg C\langle t, m\rangle \in CS(r)$.
- For all m such that m is a product or cofactor-out of $r, \neg C\langle$"*severely accumulated*", $m\rangle \in CS(r)$ (Product Inhibition[6]).
- If ratio $T = m_1/m_2$ of energy metabolite pairs m_1 and m_2 is specified as an activator for r, then C$_1$("*accumulated*," m_1) $\in CS, \neg C_2$("*accumulated*," m_2) $\in CS(r)$. If T is an inhibitor, then $\neg C_1$("*accumulated*," m_1) $\in CS(r), C_2$("*accumulated*," m_2) $\in CS(r)$.

Note that the triggering status t in the above definition is a reaction-specific metabolite status. As an example, for some reaction, t may be "accumulated" while, for others, it may be "available."

Condition set for the reverse direction of a reversible reaction is created similarly by first switching the roles for metabolites that act as substrate and product, and cofactor-in and cofactor-out in the forward (default) direction of the reaction.

Then, according to switched roles, the condition set of the reaction is created as described in the above definition. Note that, since inhibitors and activators are enzyme-specific, no role switch is performed for such regulator metabolites even if the direction of the reaction is specified as reversed in a user-formulated *MQL* query.

Example 5.7. For a reaction r with (i) a single substrate m_1 and its triggering condition "*accumulated*" for m_1, and (ii) a single product m_2, the reaction condition set *CS(r)* is $\{C\langle AC, m_1\rangle, \neg C\langle SAC, m_2\rangle\}$.

Definition (*status of a reaction*): The status of a reaction in the metabolic network is one of *active*, *inactive*, or *unknown*.

Initially, status of all reactions in the metabolic network is "unknown."

Next, we define the "status" (i.e. active or inactive) of a reaction or pathway based on the satisfaction of its associated conditions.

Definition (*Active Reaction*): Given a reaction r with an associated set of conditions defined on the participating metabolites in r, r is considered to be *active* (i.e. r has increased flux) if (i) all the conditions that involve substrates, cofactors, and products are satisfied, and (ii) among the conditions (if any) involving regulators, those conditions that include regulator(s) with the highest precedence are satisfied.

Definition (*Completely Active Pathway*): Given a pathway p with an associated set of conditions defined on the participating metabolite pools in p, p is said considered to be *completely active* if all of its reactions are active. That is, (i) all the conditions that involve its substrates, cofactors, and products are satisfied, and (ii) all regulated reactions of p are *active*.

Then, it implicitly follows from the above definitions that (a) a reaction is *inactive* (i.e. it has zero/decreased flux), when at least one of the conditions in its associated condition set is *not* satisfied, and (b) a pathway is not completely active when at least one of its reactions is inactive.

5.2. *Processing MQL_{AIP} queries*

5.2.1. *Stages of the query processing*

Input to MQL_{AIP} queries consists of a subnetwork of the metabolic network as defined by a set of pathways in specific compartments, a dietary state condition, a physiological condition, and a set of initial concentration changes. The query processing has three stages.

(a) Stage 1 (*Query compilation and preparation stage*)

1.1 Convert dietary state and physiological condition predicates in the query into their pre-defined concentration change sets. Take the union of such concentration change sets, and let the resulting set of concentration changes be *CCS*.(*QPR* 19)

1.2 Associate the user-specified concentration changes in the *where* clause of the query with the default pool of the corresponding metabolite (only in terms of increase and decrease) in the specified biological compartment (*QPR* 7).
1.3 Take the union of user-provided set *CC* of concentration changes in the *where* clause of the query and set *CCS* (from the previous step 1.1). Let the resulting set be U. Mark those pools in U with increased concentrations as "*accumulated*," and those with decreased concentrations as "*unavailable*."
1.4 Let P be the list of pathways in the "from" clause of the query. Expand P with additional pathways that connect user-provided metabolites (i.e. with their default pools) to the pathways in P (e.g. *Glycerol* to *Didydroxyacetone* 3-*P* pathway in Fig. 1, which is originally not a part of *Gluconeogenesis*).
1.5 Let *CO* be the compartment set specified in the query. Extend *CO* with the descendants of compartments in *CO* (*QPR* 16)
1.6 Construct a metabolic network with the pathways in P and compartments in *CO* based on the graph model of Section 3 (*QPRs* 8(b), 14, 15). We refer to this metabolic subnetwork as "*the query subnetwork*."
1.7 Create four initially empty pathway sets, P^{active}, $P^{\text{inactive1}}$, $P^{\text{inactive2}}$, and P^{sink}. Initially assume that all pathways in P are inactive, i.e. initialize $P^{\text{inactive1}} = P$. Also, create a set, $Ctrl^P$, to store regulation information to be returned to the user in the query output.

In stages 2 and 3, whenever an inhibition/activation is in question, we record the associated regulation information and the effected reactions/pathways in $Ctrl^P$. Since this process does not involve much complexity, for brevity, in the remaining part of this discussion, we will not refer to the process of keeping track of this regulation information.

(b) Stage 2 (*Identifying completely active pathways*)

This is the main query processing stage where the set P^{active} of pathways that are completely active are identified by following the biochemical principles and the corresponding query processing rules of Sec. 2. It has two substages: expansion and shrinking.

***b.1. Expansion*:** Expansion is an iterative process, where, in each iteration, the set of completely active pathways are expanded based on the availability of substrates.

2.1 Mark each pathway $p \in P^{\text{inactive1}}$ as *conditionally active*, and move p into P^{active}, if there is at least one condition $C(t, m) \in CS(p)$ where (i) m is a substrate in p, and (ii) C is *satisfied*. (*QPR* 1).
2.2 Update metabolite pool marks (*QPRs* 17.1, 17.2).
2.3 If the content of P^{active} has changed in step 2.1, go to step 2.1 for another iteration. Otherwise, continue with the next stage.

***b.2. Shrinking*:** Shrinking is also an iterative process, where, in each iteration, the set of completely active pathways is shrunk based on the accumulation or availability of energy currency metabolite pools, other cofactors, and regulators.

2.4 For each pathway $p \in P^{\text{active}}$, if there exists one condition $C \in CS(p)$ such that (i) C is *not satisfied*, and (ii) there is no other *satisfied* condition $C' \in CS(p)$ involving a regulator of higher precedence with opposite effects, move p into $P^{\text{inactive2}}$ (Item (ii) is applicable only for conditions that involve regulators). Update metabolite pool marks (*QPRs* 17(a), 17(b)).

2.5 If the content of P^{active} has changed in step 2.4, go to step 2.4 for a new iteration. Otherwise, continue with the next step.

2.6 (*Locating inhibitory cycles of Sec.* 5.1.1, *and moving them into* P^{sink}) Initialize a pathway dependency graph $G(V = P, E = \emptyset)$. Identify each pathway p_1 in $P^{\text{inactive2}}$ where p_1 was put into $P^{\text{inactive2}}$ due to a set UC $\subseteq CS(p_1)$ of *unsatisfied* conditions involving a regulator of higher precedence with opposite effects. However, now all conditions (except those that are surpassed due to regulator precedence) in $CS(p_1)$ are satisfied, owing to the modified pool marks in step 2.4. Add an edge $e(p_1, p_2)$ in G for each $p_2 \in P^{\text{inactive2}}$ where p_2 is a consumer/producer of a metabolite pool which is included in a condition in *UC*. Next, identify cycles in G, move members of each cycle into P^{sink}. Finally, for each edge e that is not part of a cycle in G, move e.*source* into $P^{\text{inactive1}}$.

2.7 If content of P^{active} or $P^{\text{inactive1}}$ has changed in step 2.4 or 2.6, go to step 2.1 for another iteration. Otherwise, terminate this stage; now P^{active} contains all completely active pathways.

(c) Stage 3 (*Identifying partially active pathways*)

In the previous stage of query processing, only completely active pathways (those in P^{active}) are identified. However, it is possible that some pathways may be *partially active* (i.e. a subset of its reactions have flux through them) mainly to provide bridges between completely active pathways. The last query processing stage focuses on identifying such pathways, by locating all active reactions and placing them into the set of reactions R^{active}.

Our approach in this step builds upon the general idea that there will be an active flux through non-regulated reactions as long as their substrates are *available* (or *accumulated*, depending on the trigger condition), and their products are consumed. Thus, in this step, we revise and adapt the approach that we developed for locating active/inactive pathways in stage 2 to reactions.

3.1 Let R be all the reactions in the metabolic network. Create four initially empty reaction sets, $R^{\text{active}}, R^{\text{inactive1}}, R^{\text{inactive2}}$, and R^{sink}. Initialize R^{active} with all the reactions of pathways in P^{active}, put all the remaining reactions in R into $R^{\text{inactive1}}$. Also, create a set, $Ctrl^R$, to store reaction regulation information to be returned to the user in the query output.

***c.1. Expansion*:** Expansion is an iterative process, where, in each iteration, the set of active reactions are expanded based on the availability of substrates.

3.2 Mark each reaction $r \in R^{\text{inactive1}}$ as *conditionally active*, and move r into R^{active} if there is at least one condition $C(t, m) \in CS(r)$ where (i) m is a substrate in r, and (ii) C is *satisfied*. (*QPR* 1).
3.3 Update metabolite pool marks (*QPRs* 17.1, 17.2).
3.4 If the content of R^{active} has changed in step 3.2, go to step 3.2 for another iteration. Otherwise, continue with the next stage.

***c.2. Shrinking*:** Shrinking is also an iterative process, where, in each iteration, the set of active reactions is shrunk based on the accumulation or availability of energy currency metabolite pools, other cofactors, regulators, and substrates.

3.5 For each reaction $r \in R^{\text{active}}$, if there exists one condition $C \in CS(r)$ such that (i) C is *not satisfied*, and (ii) there is no other satisfied condition $C' \in CS(r)$ involving a regulator of higher precedence with opposite effects, move r into $R^{\text{inactive2}}$ (Item (ii) is applicable only for conditions that involve regulators). Update metabolite pool marks (*QPRs 17*(a), 17(b)).
3.6 If the content of R^{active} has changed in step 3.5, go to step 3.5 for a new iteration. Otherwise, continue with the next step.
3.7 (*Locating inhibitory cycles of Section* 5.1.1*, and moving them into* R^{sink}) Initialize a dependency graph $G(V = R, E = \emptyset)$. Identify each reaction r_1 in $R^{\text{inactive2}}$ where r_1 was put into $R^{\text{inactive2}}$ earlier due to a set $UC \subseteq CS(r_1)$ of *unsatisfied* conditions involving a regulator of higher precedence with opposite effects. However, now all conditions (except those that are surpassed due to regulator precedence) in $CS(r_1)$ are satisfied, owing to the modified pool marks in step 3.5. Add an edge $e(r_1, r_2)$ in G for each $r_2 \in R^{\text{inactive2}}$ where r_2 is a consumer/producer of a metabolite pool which is included in a condition in *UC*. Next, identify cycles in G, move members of each cycle into R^{sink}. Finally, for each edge e that is not part of a cycle in G, move e.*source* into $R^{\text{inactive1}}$.
3.8 If contents of R^{active} or $R^{\text{inactive1}}$ have changed in step 3.5 or 3.7, go to step 3.2 for a new iteration. Otherwise, terminate query processing; now R^{active} contains all active reactions, and $R^{\text{inactive1}}$ and $R^{\text{inactive2}}$ contain all inactive reactions.

5.2.2. *A complete example*

Consider the query in Example 4.2.

Stage 1:

1.1 Converting the fasting stage into its signature concentration change, we have:

$$S = \{\text{insulin} \downarrow, \text{glucagon} \uparrow, \text{glucose} \downarrow, \text{fatty acids} \uparrow, \text{ketone bodies} \uparrow, \text{glycerol} \uparrow\}.$$

Each concentration change statement refers to a pool of the corresponding metabolite in S. Each hormone (e.g. insulin, glucagon) has a single pool (not shown in Fig. 1). In terms of associated pools in Fig. 1, *glucose* in S refers *to*

blood *glucose pool* #1 in Fig. 1, *fatty acids* in S refer to blood *fatty acids pool* #2 in Fig. 1, and others in S have single pools in blood as shown Fig. 1.

1.2 User-provided concentration changes (i.e. *lactate*↑, *alanine*↑, *fatty acids*↑, *glycerol*↑) in the select query of Example 4.2 are mapped to their default pools in blood. Figure 1 shows only a single pool for each of *lactate*, *alanine*, and *glycerol* in blood, so they are the default pools. For *fatty acids*, in this example, the default pool in blood is pool #2.

1.3 Take the union of user-provided set of metabolite pools and those in S:

$$U = \{\text{insulin} \downarrow, \text{glucagon} \uparrow, \text{glucose} \downarrow, \text{fatty acids} \uparrow, \text{ketone bodies} \uparrow, \text{lactate} \uparrow, \text{alanine} \uparrow\}$$

1.4 Original set P of pathways = {*Glycolysis*, *Gluconeogenesis*, *TCA-Cycle*, *Beta-Oxidation*, *Ketone-Body-Synthesis*, *Fatty-Acid-Synthesis*}. In order to provide the connection between these pathways, and the input concentration changes in U, the following pathways (here we use *substrate*2*product* naming convention for single step pathways or non-standard secondary pathways): *Glycerol*2*Dihyroxyacetone*3-*P*, *Alanine*2*Pyruvate*, *Lactate*2*Pyruvate*, *Pyruvate*2*AcetylCoA*, and *Respiratory Chain*. Hence, we expand P as

$$P = \{\textit{Glycolysis}, \textit{Gluconeogenesis}, \textit{TCA-Cycle}, \textit{Beta-Oxidation}, \textit{Ketone-Body-Synthesis}, \textit{Fatty-Acid-Synthesis}, \textit{Glycerol2Dihyroxyacetone3-P}, \textit{Alanine2Pyruvate}, \textit{Lactate2Pyruvate}, \textit{Pyruvate2AcetylCoA}, \textit{Respiratory Chain}\}.$$

1.5 Revise C.

1.6 Construct the query subnetwork.

1.7 $P^{\text{inactive1}} = P$. Set P^{active}, $P^{\text{inactive2}}$, and P^{sink} as empty. Associate each pathway with its set of conditions.

Stage 2 (*Expansion*):

- Iteration 1:

2.1 Revise P^{active} = {*Beta-Oxidation*, *Glycerol*2*Dihyroxyacetone*3-*P*, *Alanine*2*Pyruvate*, *Lactate*2*Pyruvate*}
Revise $P^{\text{inactive1}}$ = {*Glycolysis*, *Gluconeogenesis*, *TCA-Cycle*, *Ketone-Body-Synthesis*, *Fatty-Acid-Synthesis*, *Pyruvate2AcetylCoA*, *Respiratory Chain*}

2.2 *Newly available metabolites due to recent additions into* P^{active} :

$$\{\textit{AcetylCoA} \uparrow, \textit{NADH} \uparrow, \textit{Pyruvate} \uparrow, \textit{Dihydroxyacetone3-P} \uparrow\}$$

//we only show end-products, as intermediates do not make any difference //for this example based on the wiring in Fig. 1.

2.3 Since P^{active} has changed, perform a new iteration over steps 2.1 through 2.3.

- Iteration 2:

2.1 Revise $P^{\text{active}} = \{$*Beta-Oxidation*, *Glycerol2Dihyroxyacetone3-P*, *Alanine2 Pyruvate*, *Lactate2Pyruvate*, ***Gluconeogenesis***, **TCA-Cycle, Ketone-Body-Synthesis**, **Respiratory Chain**$\}$

//***bold ones*** are those that are newly added in ***this iteration.***

Revise $P^{\text{inactive1}} = \{$*Glycolysis*$\}$

2.2 ***Newly available metabolites due to recent additions*** into P^{active}:

$\{\text{Glucose} \uparrow, \text{Ketone Bodies} \uparrow, \text{Fatty Acids}, \text{NADH} \uparrow, \text{ATP} \uparrow\}$

//*we only show end-products, as intermediates do not make any*

//*difference for this example based on the wiring in Fig* 1.

2.3 Since P^{active} has changed, perform a new iteration over steps 2.1 through 2.3.

- Iteration 3:

2.1 $P^{\text{active}} = \{$*Beta-Oxidation*, *Glycerol2Dihyroxyacetone3-P*, *Alanine2 Lactate2Pyruvate*, *Gluconeogenesis*, *TCA-Cycle*, *Ketone-Body-Synthesis*, *Respiratory Chain*, *Fatty-Acid-Synthesis*, *Pyruvate2 AcetylCoA*$\}$

$P^{\text{inactive1}} = \{$*Glycolysis*$\}$

2.2 Newly available metabolites (due to recent additions into P^{active}) : $\{\ \}$

2.3 Since P^{active} has not changed, continue with step 2.4.

Stage 2 (shrinking)

- Iteration 1:

2.4 The energy currency metabolite *NADH* has producers, *Beta-Oxidation* and the *TCA-Cycle*, which outpaces the consumption through the respiratory chain. Hence, the *NADH* pool accumulates. *NADH* inhibits two rate limiting steps of the *TCA-Cycle*. In addition, *Fatty Acid Synthesis* and *Pyruvate2AcetylCoA* are inhibited by *Fatty Acids* and *Glucagon*. Hence,

$$P^{\text{active}} = \{\textit{Beta-Oxidation}, \textit{Glycerol2Dihyroxyacetone3-P}, \textit{Alanine2Pyruvate}, \textit{Lactate2Pyruvate}, \textit{Gluconeogenesis}, \textit{Ketone-Body-Synthesis}, \textit{Respiratory Chain}\}$$

$$P^{inactive2} = \{\textbf{\textit{TCA-Cycle, Fatty-Acid-Synthesis}}, \textbf{\textit{Pyruvate}2AcetylCoA}\}$$

//***bold*** *ones are those that are removed from* P^{active} *in this iteration.*

2.5 Since the content of P^{active} has changed, go to step 2.4 for a new iteration.

- Iteration 2: There are no changes in the content of P^{active}. Hence continue with step 6.

  2.6 (locating cycles and moving them into P^{sink} as discussed in Section 5.1.1) $P^{\text{inactive2}} = \{TCA\text{-}Cycle,\ Fatty\text{-}Acid\text{-}Synthesis,\ Pyruvate2AcetylCoA\}$, and *TCA-Cycle* is still inhibited by *NADH*. The dependency graph is empty.

  2.7 Contents of P^{active} have changed. A new iteration is required starting from step 2.1. In the last iteration, the sets of active and inactive pathways do not change. There are no inhibitory cycles; thus, P^{sink} is empty. That is, we have:

$$\begin{aligned} P^{\text{active}} &= \{Beta\text{-}Oxidation, Glycerol2Dihyroxyacetone3\text{-}P,\\ &\quad Alanine2Pyruvate, Lactate2Pyruvate, Gluconeogenesis,\\ &\quad ketone\text{-}Body\text{-}Synthesis, Respiratory\ Chain\}\\ P^{\text{inactive1}} &= \{Glycolysis\}\\ P^{\text{inactive2}} &= \{TCA\text{-}Cycle,\ Fatty\text{-}Acid\text{-}Synthesis,\ Pyruvate2AcetylCoA\}\\ P^{\text{sink}} &= \{\} \end{aligned}$$

  Stage 2 terminates. Go to Stage 3.

Stage 3:
There is no partially active pathway in this example due to the fact that no reaction in *inactive* pathways has all of its conditions satisfied based on the definitions in Sec. 5.1. Hence, R^{active} does not grow in this stage.

Final query Result: The list of completely active and completely inactive pathways (no partially active pathways):

$$\begin{aligned} P^{\text{active}} &= \{Beta\text{-}Oxidation,\ Glycerol2Dihyroxyacetone3\text{-}P,\\ &\quad Alanine2Pyruvate,\ Lactate2Pyruvate, Gluconeogenesis,\\ &\quad Ketone\text{-}Body\text{-}Synthesis,\ Respiratory\ Chain\}\\ Inactive\ pathways &= \{Glycolysis,\ TCA\text{-}Cycle,\ Fatty\text{-}Acid\text{-}Synthesis,\\ &\quad Pyruvate2AcetylCoA\}\\ P^{\text{sink}} = \{\} \quad &\text{and} \quad R^{\text{sink}} = \{\} \end{aligned}$$

Please see Sec. 1.1 for visualization and explanation of results.

5.2.3. *Handling inconsistent input to MQL_{AIP} queries*

It is possible that the user-provided concentration change statements in a query may be inconsistent with respect to the activated/inactivated set of pathways included in the query result. Given a metabolite pool m and a user-provided concentration change cc of m, such an inconsistency may occur in two different

symmetric cases: (i) *cc* involves an increase in m, and major producers of m are determined to be inactive in the query result (and probably major consumers are determined to be active in the query result), or (ii) *cc* involves a decrease in m, and major producers of m are determined to be active in the query result (and probably major consumers are determined to be active in the query result). We give an example.

Example 5.2: Consider the metabolite pool of *acetyl CoA* in liver mitochondrion. Assume that, in the result of a query Q, the consuming pathway (*Fatty Acid Synthesis*) of *acetyl CoA* is inactive (perhaps, due to the inhibition of its rate-limiting step *acetyl CoA carboxylase*), and, at least one of the contributing pathways (e.g. *Beta Oxidation*) of the *acetyl CoA* pool is active (perhaps due to increased *Fatty Acid* in blood, *Fatty Acid* is transported into liver). In such a setting, a certain amount of *acetyl CoA* will accumulate, and if Q involves a concentration decrease for *acetyl CoA* in liver mitochondrion, Q is inconsistent.

In characterizing such types of query input inconsistencies, we employ the *closed world assumption*[25] in that there cannot be a reaction r that (a) consumes/produces m, and (b) r is not included in the metabolic query subnetwork.

Definition (*Closed World Assumption*): Given a metabolite pool m, and the set R of reactions within a set of pathways P which are included in a query, let $C(m)$ be the consumers of m, and $P(m)$ be the producers of m. Then, $C(m) \subseteq R$ and $P(m) \subseteq R$ holds.

Now, we are in a position to formally define query input inconsistency based on the *Closed World Assumption* and the Query Processing Rules 17(a) and 17(b).

Definition (*Inconsistent Query Input*): Given an MQL_{AIP} query Q, let C be the set of metabolite concentration change pairs (m_i, c_i) where m_i is a metabolite pool, and c_i is a concentration change statement (i.e. "increase" or "decrease"), that are either provided directly by the user, or obtained from the built-in concentration change set of a dietary state or physiological condition included in the query. Then, Q is called an *inconsistent query* if there is at least one concentration change pair $(m_i, c_i) \in C$ such that

- m_i is marked as *unavailable* in the query result, and c_i = increase, and/or
- m_i is marked as *severely accumulating* or *accumulating* in the query result, and c_i = decrease.

Note that the above definition includes query inconsistency involving conflicts between user-provided metabolite concentration changes and those that are included in the signatures of a dietary state or a physiological condition predicate.

At the end of Stage 3 of query processing, the above definition is employed to determine if the query is inconsistent. Inconsistent queries return empty query result sets. However, as an explanation, users are also provided with those particular metabolite-concentration change pairs that render the query inconsistent.

5.2.4. *Discussion*

In this section, we present a brief discussion on the termination behavior of our query processing algorithm which contains looping structures among/within different steps. It is crucial that the algorithm does not get into infinite loops. In the algorithm, since there is no loop that spans over multiple query processing stages, each stage can be analyzed independently.

Stage 1 does not contain any looping structure.

In Stage 2, pathways are moved between three different sets that may potentially lead to infinite loops: $\boldsymbol{P}^{\text{inactive1}} \rightarrow$ (Steps 2.1 $\cdot\cdot$ 2.3) $\boldsymbol{P}^{\text{active}} \rightarrow$ (Step 2.4) $\boldsymbol{P}^{\text{inactive2}} \rightarrow$ (Step 2.6) $\boldsymbol{P}^{\text{inactive1}}$. If a pathway p is continuously circulated through these three pathway sets, then the algorithm never terminates. Such cases may happen when there is a set of pathways which are inter-dependent on each other in a cyclic manner through regulatory relationships. Hence, such pathways with cyclic interdependency should be eliminated from consideration in Stage 2. Construction of a dependency graph in step 2.7 and removal of pathways (by moving them into P^{sink}) that form cycles is integrated into the algorithm to prevent such infinite looping cases.

Finally, in Stage 3, the only looping structure takes place in step 3.4 which iterates over itself. This step performs backtracking due to product inhibition at intermediate steps of a pathway. Hence, it only expands the set of inactive reactions, and does not manipulate the set of active reactions. Therefore, the number of iterations is bounded by the total number of reactions in the query sub-network.

5.3. *Processing MQL_{PFC} queries*

In order to answer MQL_{PFC} queries, in the first step, the query processor needs to compute the active and inactive pathways/reactions as it is done for MQL_{AIP} queries. For this step, the same query processing model which is discussed in Sec. 5 is employed. In the second step, the annotated (i.e. active, inactive paths) metabolic query subnetwork needs to be analyzed for *prevented futile cycles* and their regulation. We first give a definition for the concept of "prevented futile cycle."

Definition (*Prevented Futile Cycle*): Given a query subnetwork graph M with annotations regarding active and inactive paths, a prevented futile cycle is a simple cycle in M with exactly one edge annotated as "inactive," while other edges annotated as "active."

The overall query processing for MQL_{PFC} queries involves the following steps:

- Create the query subnetwork with edge labels "active" and "inactive" by invoking the process described in Section 5.
- On the labeled (i.e. annotated) query subnetwork, perform depth-first-search traversal, and locate all possible cycles ignoring annotations on edges regarding activation/inactivation.
- Check each cycle against the *Prevented Futile Cycle* definition to see if it is a *prevented futile cycle*. If it is a prevented futile cycle, locate the "inactive" edge.

Add into the query result set (i) the prevented futile cycle with inactive edge shown explicitly, and (ii) form an English explanation regarding the regulation of the blocked edge based on its activators and inhibitors.

Please see Example 4.4 for a sample query and its query result.

6. Related Work

Two studies in the literature focus on the design of query languages for biochemical networks. *PQL*, the pathway query language,[4] employs a basic graph model where nodes represent metabolites, enzymes, or processes, and edges represent participation of a metabolite in a reaction in different roles, or inhibition/activation relationships between two different enzymes. *PQL* allows formulation of different query types that include relationship queries (e.g. reactions catalyzed by an enzyme), neighborhood queries, and path queries. *bcnQL*[5] is another query language with XQuery-like syntax designed for biochemical networks. *bcnQL* employs an object oriented graph model where nodes and edges have the same semantics as in *PQL*. *bcnQL* supports the formulation of almost the same types of queries as *PQL* with the improvement that *bcnQL* provides additional capabilities to specify multiple predicates on path queries. Both *PQL* and *bcnQL* employ simpler models of metabolic networks, not sufficiently capturing the metabolism: they do not accommodate (i) dynamic behavior of the metabolism under different physiological or dietary conditions, (ii) the compartmentalized structure of the metabolism over tissues and organelles, and (iii) regulatory relationships between pathways. And, neither *PQL* nor *bcnQL* provide a capability to specify an initial set of concentration changes on key metabolites to guide query processing, and do not eliminate biologically infeasible query results.

There are also many Web-based metabolic databases (e.g. *KEGG*,[6] *MetaCyc*,[7] *Reactome*,[8] *PathCase*,[9] etc.) with query languages. Such data sources serve well for basic database querying via built-in or dynamically constructed queries (e.g. *AQI*,[26] Structured Advanced Query Page and Advanced Query Form[27]). *KEGG*[6] provides basic keyword search, while *MetaCyc*[7] and *Reactome*[8] include advanced query forms. LISP framework in *Pathway Tools/BioCyc/MetaCyc* provides a custom metabolism-specific query capability.[27] However, the query languages and forms of these data sources also have the same drawbacks listed for *PQL* and *bcnQL*. Besides, data models of these data sources do not capture different trigger mechanisms for pathways and/or regulators, distinct contributions of each pathway/reaction into a particular metabolite pool, and varying occurrences of the same pathway in different tissues.

Qualitative Physics[28] and *Qualitative Reasoning* (*QR*)[29–30] are employed to model systems and environments where measured quantitative information is not available or appropriate to use, but there exists high level (i.e. commonsense) knowledge about the working principles of underlying systems. Studies that adapt qualitative reasoning (or physics) into biochemistry and molecular biology[2,31] (see Ref. 32

for a review) are, at a high level, related to the *MQL* framework. *BioSim*[2] is a *QR* application for simulation of pathways via automated creation of "process" and "object" models based on reactions and their participants (i.e. substrates, products, enzymes), respectively, in a pathway. Then these created models of *BioSim* are executed by a Prolog-based simulation engine which creates a "behavior tree" describing concentration changes for each involved metabolite and enzyme. King *et al.*[31] extends *BioSim*'s simulation approach, and proposes a model identification framework, where given a qualitative time series set of metabolite measurements, the goal is to identify the structure of the metabolic subnetwork. *GenSim*[3] is another simulation environment proposed for biochemical systems. In *GenSim*, users manually construct object (i.e. molecules) and process (i.e. reaction) knowledgebases via a Lisp-based representation scheme, and define preconditions for the processes to be active, as well as their effects, once they are active. There are also many other similar qualitative simulation studies,[33–40] which we do not discuss in this paper due to the lack of space.

MQL does not directly compare to these simulation works, as it is not designed as a simulation system, but as a query language and its query processing engine. However, *MQL* employs similar ideas regarding the qualitative reasoning, and (pre)conditions (constraints) that are checked for reactions to decide their activity status. *MQL* is different from these simulation studies in that it employs (i) an extensive and detailed metabolism data model, and (ii) fine-tuned biochemical principles, which are not considered by the above-listed works.

6.1. *Metabolic network analysis techniques: A brief comparison with MQL query processing*

As we have stated in Sec. 1.3, *MQL* query processing technique can be viewed as being in the general category of metabolic analysis techniques. In this section, we summarize the existing metabolic network analysis techniques, and briefly compare with *MQL* query processing.

Over the last 30+ years, a number of powerful mathematical modeling approaches and their corresponding computational tools have been proposed and used to study the dynamics of cellular metabolism. These techniques have many goals such as determining the metabolic fluxes of reactions in the metabolic network, or finding all the "optimal" routes, etc. They include metabolic control analysis (*MCA*),[10–13] flux balance analysis (*FBA*)[14–16] (also known as *constrained optimization*), *metabolic flux analysis*,[38] and *metabolic pathway analysis* (more specifically, *elementary flux modes* and *extreme pathways*).[15,18–20] Next we briefly summarize these techniques, and compare them with *MQL* query processing.

Metabolic control analysis. *MCA* is a mature mathematical methodology for characterizing metabolic systems using the response, control, and elasticity coefficients. It is a systems-level approach to the study of metabolism, and aims to characterize the sensitivity of metabolic responses with respect to changes

in enzyme activities or parameters *without* the use of full mathematical models (since complete and accurate models of metabolic models are usually not available and are not expected to be in the near future). The structure of the metabolic system is represented by its stoichiometry. Then there are parameters and variables. Parameters are quantities that can be changed independently; they typically remain constant during the evaluation of the system. Examples are kinetic constants, enzyme concentrations, and external inhibitors. Variables are determined by the system, and are time-dependent before reaching their steady-state. Examples are metabolite concentrations. *MCA* is primarily concerned with the description of metabolic regulation at the steady state, quantifying how changes in "parameters" modify steady-state responses. To determine how a steady-state response is affected by changes in a parameter, *MCA* relies on three types of coefficients: elasticity, response, and control. In general, three different types of software packages can be used for modeling kinetics and control analysis of biochemical networks:

- Generic mathematical modeling software, such as *Mathematica* or *Matlab*, which require a mastery of the mathematics involved, and are not suitable for life-scientists,
- Dedicated metabolic simulators such as *Jarnac*,[41] "recommended *only* for very experienced users,"[42]
- "SBML-capable" software packages with *GUI* front-end such as *COPASI*,[43,44] *JDesigner*,[45] or *RoadRunner*,[46] which allow users to specify the models and run simulations with reduced knowledge of the mathematics involved. A comparison of a number of these packages is available on the web.[47] Our own experience with the use of *JDesigner* and *Roadrunner* is that these software packages, while quite easy to use, have many issues,[48] which we list only two here:

Lack of feedback from simulators: When an *SBML* model is provided as input to a simulator, if there is any problem with the *SBML* file or with simulation process itself, the simulator either (i) returns no error message (and sometimes just freezes or exits abruptly), or (ii) returns a very generic message which often points to an interprocess-communication error with no particular pointers on the nature of the problem. Such a behavior leaves users almost clueless about the actual causes of a particular problem, which turns debugging the problem into a blind trial & error process.

No simulation parameters included in SBML: Unambiguous simulation of a model requires specification of additional parameters (not included in *SBML*) such as tolerance threshold for convergence, number of steps in each time point, and so on. Missing specifications for such simulation parameters dramatically affects the behavior of a simulator, as much as from producing the correct result to producing no results.

Flux balance analysis. *FBA* is a widely applied method for the computation of stationary fluxes in large-scale metabolic networks; it is based on convex analysis imposing an objective function subject to several constraints, to determine the metabolic flux vector. The advantage of *FBA* in comparison with kinetic modeling is that it also requires only (basically) the knowledge of the stoichiometry of the network. *FBA* relies on the hypothesis that the most likely distribution of stationary fluxes in the network has to be optimal with respect to a feasible optimization criterion linking the fluxes with the cellular functions. Usually, the fluxes are determined to maximize a specific network output, e.g. the production of biomass[14] which is a reasonable objective for primitive cells such as bacteria, but not necessarily for complex eukaryotic cells. As a more general optimization criterion, the principle of (internal) flux minimization[49] and its extensions[50] are proposed. Critiques of *FBA* include: (a) *FBA* identifies only one optimal solution (while there may be other optimal/suboptimal solutions that exist), (b) flux distributions predicted by *FBA* are hypothetical (because they depend on the choice of the flux criteria used),[50] and (c) it has high (exponential) computational (time) complexity.

Metabolic pathway analysis. In general, metabolic pathway analysis identifies the topology of cellular mechanism based only on the stoichiometric structure and thermodynamic constraints of reactions, also without requiring kinetic parameters of reactions. The two main techniques in metabolic pathway analysis are *elementary flux mode analysis* (*EMA*)[18,51] and *extreme pathways analysis* (*EPA*).[19] In comparison with *FBA*, metabolic pathway analysis can identify all metabolic flux vectors; but it also has high computational complexity. However, constraints such as nondecomposibility and systematic independence can result in finite solutions. Trinh *et al.*[20] gives an excellent review of *elementary mode analysis*; and Klamt and Stelling[52] compare the common features, differences, and the applicability of *EMA* and *EPA* techniques.

Elementary Flux Mode analysis. *EMA* reduces the metabolic network into all possible, unique, non-divisible paths. Elementary flux modes are basically linearly independent basis vectors in the admissible flux space, satisfying nondecomposibility and thermodynamics constraints. Put another way, given a set of *EMA* vectors (each representing a flux distribution), by adding or subtracting multiples of them, one can obtain all admissible flux distributions. Each *EMA* specifies a *minimal* set of enzymes in that if only the enzymes of a given *EMA* are operating, inhibition of any of the enzymes would eliminate the steady-state flux in the system. As compared with *FBA, EMA* analysis (a) provides the set of all *EMA*s containing optimal/suboptimal routes converting a certain metabolite to a product, subject to the thermodynamics constraint and nondecomposibility constraint, (b) identifies enzyme subsets that always have to operate together (i.e. structurally need each other), (c) enables the user to determine the relative importance of individual reactions for system performance under different environments, (d) allows the

identification of nondecomposable steady-state flows, including cyclic flows, and (e) can locate pairs of reactions that never occur together in an *EMA*.[52]

Extreme Pathway Analysis. *EPA* adds one more constraint to make all extreme pathways systematically independent,[19,52] which means that none of the extreme pathways can be expressed as a non-negative combination of two or more extreme pathways.

Even though there has been excellent algorithmic and conceptual progress, metabolic pathway analysis remains a computationally challenging problem, due to the high computational complexity. Free applications that compute elementary flux modes include *COPASI*,[43] *Metatool*,[53–55] *SNA*,[56] and *FluxAnalyzer*.[57] *YANA*[58] provides a *GUI* interface built on top of *Metatool*. *YANAsquare*[59–60] extends *YANA* with the capability to automatically import reconstructed metabolic networks of different microorganisms from *KEGG*.[6] For computing extreme pathways, *ExPas*[61] is made available.

Comparison of MCA, EMA, and MQL approaches. Next, we briefly list the differences between the *MCA* (or *FBA*), *EMA*, and *MQL* approaches:

Different goals. The four approaches are useful in different contexts, focus on providing different sets of information to users, and have different goals.

(a) *MCA* focuses on "control as a property of the whole system": One can (i) measure (at quasi-steady state) the effect of single enzyme perturbations on the system, and (ii) calculate the control distribution, relating the system behavior to individual reactions.
(b) *EMA* can be used for tasks like the recognition of operational modes, finding all optimal paths, analysis of network flexibility (structural robustness, redundancy).[52] Under steady-state conditions, the metabolic fluxes of an organism can be expressed as non-negative, linear, weighted combinations of elementary flux modes[62]; however, identifying the weighting factors to determine the fractional contributions of each elementary mode is difficult, if not impossible.[62,63] Visualizations of elementary flux modes within a given *KEGG* pathway are also available (via *YANAsquare*).
(c) *MQL*, working with possibly the whole (and possibly large) metabolic network within a multi-tissue (organ) environment (i.e. not within a cell) and assuming steady-state behavior, returns to users one metabolic action scenario as well as their visualizations within the metabolic network, allowing users to quickly concentrate on locating possibly activated paths for a given set of observed metabolite concentration changes. *MQL* does not derive (steady-state) flux values of the *MCA* (*FBA*) method, and, thus, there are no control-related (i.e. rate limitation) conclusions (of the *MCA* method).

Different underlying fundamentals. MQL is rule-based, and employs graph search algorithms across the whole metabolic network. In comparison, *MCA* and *FBA*

involve solving a set of underconstrained differential equations corresponding to a possibly smaller metabolic network at hand. *EMA* determines elementary fluxes via a linear combination of "null space basis vectors" of the stoichiometry matrix.[55]

Ease of use. MCA (or *FBA*), even with the easiest-to-use *GUI*-oriented software tools (such as *COPASI*), requires (i) additional information to be collected and provided by the users including the stoichiometry information, and (ii) setup and usage expertise, for biologists to use them. The *EMA* tools *YANA* and *YANAsquare* do provide user-friendly elementary flux derivations and their visualizations. In comparison, *MQL* uses a metabolic pathways database, which already contains the metabolic network, biochemistry-based rules and other information (e.g. stoichiometry) so that all that a user is expected to provide is a set of observed metabolite changes (in the form of increase/decrease/no change).

Modeling-related restrictions/assumptions. As listed above, *MCA* has a number of assumptions (such as requiring a connected network of pathways)[64] which are not needed for *MQL*. *EMA* also requires connectivity.

Computational Complexity. Computational complexity of *MCA* is exponential in the number of reactions involved, forcing users to use various compaction, aggregation, and clustering/merging, etc. techniques. Computational complexity of *EMA* is also exponential,[52] and various approaches to tackle the high complexity are proposed such as parallel computing,[65] network decomposition and "functional conversion of flux cones." In its worst case (with as mant backtracking iterations as the number of reactions), *MQL* is also exponential in the number of paths. However, metabolic networks form sparse graphs, and, for the prototype metabolic network used in Sec. 5, the worst-case complexity has not been a limiting factor.

7. Conclusion

Querying the metabolic behavior of organisms is important for systems biology researchers as well as students of biochemistry. In this paper, we have presented a metabolism query language, *MQL*, which enables researchers to explore the metabolism with different constraints. The query processing of *MQL* is designed in accordance with the fundamental principles that organize the cellular actions of metabolism into a coherent and complex system. *MQL* is presently being implemented and integrated into the Metabolomics Analysis Workbench.[1,66]

Future research on MQL includes

- Extending MQL's single answer into an "answer set," and give users alternative results where each result corresponds to a specific resolution of a race condition (as illustrated in Section 5.1.1).
- Incorporating the use of the stoichiometry of each reaction into MQL query processing,

- Handling special cases such as multiple enzymes cooperating to perform a reaction,
- Extending and/or refining the biochemistry rules of Section 2 for more precise and correct MQL query processing.

References

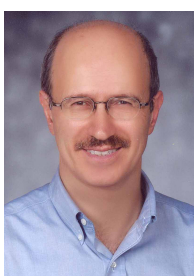
1. Cakmak A, Dsouza A, Hanson RW, Ozsoyoglu ZM, Ozsoyoglu G, Analyzing metabolomics data for automated prediction of underlying biological mechanisms, *Submitted for publication.*
2. Heidtke KR, Schulze-Kremer S, BioSim — A new qualitative simulation environment for molecular biology, *in Proc of 6th Int Conf Intell Syst Mol Biol*, pp. 85–94, 1998.
3. Karp PD, A qualitative biochemistry and its application to the tryptophan operon, in *Artificial Intelligence and Molecular Biology*, pp. 289–324, 1993.
4. Leser U, A query language for biological networks, ECCB/JBI 2005, *Bioinformatics* **21**(Suppl 2):ii33–ii39, 2005.
5. Yang H, Sunderraman R, Tian H, bcnQL: A query language for biochemical networks, *IEEE International Conference on Bioinformatics and Biomedicine*, 2008.
6. Kanehisa M *et al.*, From genomics to chemical genomics: new developments in KEGG, *Nucleic Acids Res* **34**:D354–7, 2006.
7. Caspi R *et al.*, MetaCyc: A multiorganism database of metabolic pathways and enzymes, *Nucleic Acids Res* **34**(Database issue):D511–16, 2006.
8. Joshi-Tope G *et al.*, Reactome: A knowledgebase of biological pathways, *Nucleic Acids Res* **33**:D428–32, 2005.
9. Elliott B, Kirac M, Cakmak A *et al.*, PathCase: Pathways database system, *Bioinformatics* **24**(21):2526–2533, 2008.
10. Fell DA, Metabolic control analysis: A survey of its theoretical and experimental development, *Biochem J* **286**:313–330, 1992.
11. Fell DA, *Understanding the Control of Metabolism*, Portland Press, 1997.
12. Liao JC, Delgado J, Advances in metabolic control analysis, *Biotechnol Prog* **9**:221–233, 1992.
13. Wildermuth MC, Metabolic control analysis: Biological applications and insights, *Genome Biol* **1**(6):1031.1–1031.5 2000.
14. Varma A, Palsson B, Metabolic capabilities of *Escherichia coli.* 2. Optimal growth patterns, *J Theor Biol* **165**:503–522, 1993.
15. Schilling CH, Schuster S, Palsson BO, Heinrich R, Metabolic pathway analysis: Basic concepts and scientific applications in the post-genomic era, *Biotechnol Prog* **15**:296–303, 1999.
16. Edwards JS, Palsson BO, Systems properties of the Haemophilus Influenzae Rd metabolic genotype, *J Biol Chem* **274**:17410–17416, 1999.
17. Stephanopoulas G, Aristidou A, Nielsen JH, *Metabolic Engineering: Principles and Methodologies*, Academic Press, 1998.
18. Schuster S, Higetag S, On elementary flux modes in biochemical reaction systems at steady state, *J Biol Syst* **2**:165–182, 1994.
19. Schilling CH, Letscher D, Palsson BO, Theory for systemic definitions of metabolic pathways and their use in interpreting metabolic function from a pathway-oriented perspective, *J Theor Biol* **203**:229–248, 2000.
20. Trinh CT *et al.*, Elementary mode analysis: A useful metabolic pathway analysis tool for characterizing cellular metabolism, *Appl Microbiol Biotech* **81**:813–826, 2009.

21. Devlin TM, *Textbook of Biochemistry with Clinical Correlations*, 6th ed. Hoboken NJ, John Wiley & Sons, 2006.
22. Champe PC, Harvey RA, Ferrier DR, *Lippincott's Illustrated Reviews: Biochemistry*, Lippincott Williams & Wilkins; Fourth Edition, 2007.
23. Saggerson D, Malonyl-CoA, A key signaling molecule in mammalian cells. *Annu Rev Nutr* **28**:253–272, 2008.
24. Wildermuth MC, Metabolic control analysis: Biological applications and insights, *Genome Biol* **1**(6):1031.1–1031.5, 2000.
25. Bertino E, Negri M, Pelagatti G, Sbattella L, Object-oriented query languages: The notion and the issues, *IEEE Transactions on Knowledge and Data Engineering* **4**(3):223–237, 1992.
26. Elliott B, Mayes S, Cakmak A *et al.*, Advanced querying interface for biochemical network databases, *25th ACM Symposium On Applied Computing (SAC)*, Sierre, Switzerland, March 22–26, 2010.
27. Krummenacker M, Paley S, Mueller L, Yan T, Karp PD, Querying and computing with biocyc databases, *Bioinformatics* **21**:3454–3455, 2005.
28. Weld DS, De Kleer J, *Readings in Qualitative Reasoning About Physical Systems*, Morgan Kaufmann, San Mateo, CA, 1990.
29. Forbus KD, Qualitative process theory, *Artificial Intelligence* **24**:85–168, 1984.
30. Kuipers B, *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge*, MIT Press, Cambridge MA, 1994.
31. King RD, Garrett SM, Coghill GM. On the use of qualitative reasoning to simulate and identify metabolic pathways, *Bioinformatics* **21**:2017–2026, 2005.
32. De Jong H, Modeling and simulation of genetic regulatory systems: A literature review, *J Comput Biol* **9**(1):67–103, 2002.
33. Meyers S, Friedland P, Knowledge based simulation of genetic regulation in bacteriophage lambda, *Nucleic Acids Res* **12**(1):1–9, 1984.
34. Thomas R, Regulatory networks seen as asynchronous automata: A logical description, *J Theor Biol* **153**:1–23, 1991.
35. Brutlag DL, Galper AR, Millis DH, Knowledge-based simulation of DNA metabolism: Prediction of enzyme action, *Computer Applications in Biosciences* **7**(1):9–19, 1991.
36. Shimada T, Hagiya M, Arita M, Nishizaki S, Tan C, Knowledge-based simulation of regulatory action in lambda phage, *Journal of Artificial Intelligence Tools* **4**(4):511–524, 1995.
37. Clancy DJ, Kuipers B, Model decomposition and simulation: A component based qualitative simulation algorithm, *14th National Conference on Artificial Intelligence (AAAI-97)*, 1997.
38. Karp PD, Mavrovouniotis MM, Representing, analyzing, and synthesizing biochemical pathways, *IEEE Expert* **9**(2):11–22, 1994.
39. Kazic T, Reasoning about biochemical compounds and processes, in *Proceedings of the International Conference on Bioinformatics, Supercomputing and the Human Genome Project*, pp. 35–49, Singapore. World Scientific, 1993.
40. Kazic T, Representation, reasoning and the intermediary metabolism of *E. coli*, in *Proceedings of the Hawaii International Conf on System Sciences*, Vol. 1, pp. 853–862, 1993.
41. Sauro HM, Jarnac: A system for interactive metabolic analysis, *Animating the Cellular Map, 9th International BioThermoKinetics Meeting*, Stellenbosch University Press, Ch. 33, 221–228, 2000.
42. Recommendation at the home page of Jarnac site at http://www.sys-bio.org/software/jarnac.htm.

43. Hoops S, Sahle S, Gauges R, Lee C, Pahle J, Simus N, Singhal M, Xu L, Mendes P, Kummer U, COPASI — a complex pathway simulator, *Bioinformatics* **22**:3067–3074, 2006.
44. COPASI: Complex Pathway Simulator, available at http://www.copasi.org.
45. JDesigner site, available at http://www.sys-bio.org/software/jdesigner.htm.
46. Roadrunner site, available at http://www.sys-bio.org/sbwWiki/sbw/roadrunner.
47. "Comparing SBML-capable simulators," available at http://www.sys-bio.org/sbwWiki/compare.
48. Cakmak A, Lai N, Ozsoyoglu G, Ozsoyoglu ZM, PathCase-SB Model Simulation Interface: Issues and problems—Status Report. *Technical Report. Dept of EECS*, CWRU, Cleveland OH, 2009, Available at http://dblab.case.edu/docs/ProgressReports/2009.5.26.PathCase-SB.SimulationInterface.StatusReport.pdf.
49. Holzhutter H, The generalized flux minimization method and its application to metabolic networks affected by enzyme deficiencies, *Biosystems* **83**:98–107, 2006.
50. Hoppe A *et al.*, Including metabolite concentrations into flux balance analysis: Thermodynamic realizability as a constraint on flux distributions in metabolic networks, *BMC Syst Biol* **1**:23, 2007.
51. Schuster S, Fell DA, Dandekar T, A general definition of metabolic pathways useful for systematic organization and analysis of complex metabolic pathways, *Nature Biotech* **18**:326–332, 2000.
52. Klamt S, Stelling J, Two approaches for metabolic pathway analysis? *Trends in Biotech.* **21**(2):64–69, 2003.
53. Pfeiffer T *et al.*, METATOOL: For studying metabolic networks, *Bioinformatics* **15**:251–257, 1999.
54. von Kamp A, Schuster S, Metatool 5.0: Fast and flexible elementary mode analysis, *Bioinformatics* **22**:15, 1930–1931, 2006.
55. Urbanczik R, Wagner C, An improved algorithm for stoichiometric network analysis: Theory and applications, *Bioinformatics* **21**:1203–1210, 2005.
56. Urbanczik R, SNA-A toolbox for the stoichiometric analysis of metabolic networks, *BMC Bioinformatics* **7**:129, 2006.
57. Klamt S, Stelling J, Ginkel M, Gilles ED, FluxAnalyzer: Exploring structure, pathways, and flux distributions in metabolic networks on interactive flux maps, *Bioinformatics* **19**:261–269, 2003.
58. Schwarz R, Musch P, von Kamp A, Engels B, Schirmer H, Schuster S, Dandekar T, YANA-a software tool for analyzing flux modes, gene expression and enzyme activities," *BMC Bioinformatics* **6**:135, 2005.
59. Schwarz R *et al.*, Integrated network construction, visualization and analysis using YANAsquare, *BMC Bioinformatics* **8**:313, 2007.
60. Schwarz R *et al.*, Observing metabolic functions at the genome scale, *Genome Biol* **8**:R213, 2007.
61. Bell SL, Palsson BO, Expa: A program for calculationg extreme pathways in biochemical reaction networks, *Bioinformatics* **21**:1739–1740, 2005.
62. Wlaschin AP *et al.*, The fractional contributions of elementary modes to the metabolism of *Escherichia coli* and their estimation from reaction entropies, *Metabolic Eng* **8**:338–352, 2006.
63. Poolman MG, A method for the determination of flux in elementary modes, and its application to *Lactobasillus rhamnosus, Biotehnol Bioeng* **88**(5):601–612, 2004.
64. Glykys DJ, Banta S, Metabolic control analysis of an enzymatic biofuel cell, *Biotechnology and Bioengineering* **102**(6):1625–1635, 2009.

65. Klamt S *et al.*, Algorithmic approaches for computing elementary modes in large biochemical networks, *Syst Biol* **152**:245–255, 2005.
66. Cakmak A, Dsouza A, Hanson RW, Ozsoyoglu, ZM, Ozsoyoglu G, A web-based data source for metabolomics, *24th International Symposium on Computer and Information Sciences (ISCIS)*, Guzelyurt, Northern Cyprus, September 14–16, 2009.

Ali Cakmak received his B.Sc. degree in 2003 from the Computer Engineering Department at Bilkent University, Ankara, Turkey, and his Ph.D. degree in 2008 from the Electrical Engineering and Computer Science Department at Case Western Reserve University, Cleveland, Ohio, USA. After completing his Ph.D., he worked as a post-doctoral research associate for a year in the same department. His research interests broadly lie in the fields of Bioinformatics, Data Mining, Biological Data Management, Systems Biology, Information Extraction, and Digital Libraries. His works have been published in prestigious journals and conferences including Bioinformatics, BMC Bioinformatics, VLDB Journal, PSB, CSB, and EDBT. He has served on the program committees of several conferences, and also as a reviewer for major Bioinformatics journals. In 2008, he was nominated by the Case School of Engineering and selected for participation in the American Association for the Advancement of Science Program for Excellence in Science.



Gultekin Ozsoyoglu is a professor of the Department of Electrical Engineering and Computer Science, Case Western Reserve University, Cleveland, Ohio, USA. He received his BS degree in electrical engineering and the MS degree in computer science from the Middle East Technical University, Ankara, Turkey, in 1972 and 1974, respectively, and the Ph.D. degree in computing science from the University of Alberta, Edmonton, Alberta, Canada, in 1980. Prof. Ozsoyoglu's current research interests include data management and database-related issues in bioinformatics, web computing, and web data mining. He has published in all major database conferences and journals such as ACM Transactions on Database Systems, IEEE Transactions on Software Engineering, IEEE Transactions on Knowledge and Data Engineering, IEEE Computer, and Journal of Computer and System Sciences as well as in bioinformatics journals. He has served in program committees and panels of all major database conferences such as ACM SIGMOD, VLDB, IEEE Data Engineering. He was an ACM national lecturer, and general and program chairs of a number of conferences and workshops.

Richard W. Hanson is the Leonard and Jean Skeggs Professor of Biochemistry at Case Western Reserve University School of Medicine. Dr. Hanson's major research interest has been in the regulation of metabolism in mammals. This research includes detailed studies of the hormonal control of transcription for the gene for the cytosolic form of phosphoenolpyruvate carboxykinase and the role of this enzyme in the regulation of gluconeogenesis in the liver and kidney. His research led to the discovery of the pathway of glyceroneogenesis (in collaboration with Lea Reshef and John Ballard). He has been the Associate Editor of the *Journal of Biological Chemistry* for 25 years; he has 270 publications related to metabolism. Dr. Hanson is a scientific founder of *Copernicus Therapeutics*, a Cleveland-based bio-technology company and is a dedicated teacher of biochemistry and metabolism.