View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Universidade do Minho: RepositoriUM

A Framework for the Integrated Analysis of Metabolic and Regulatory Networks

Rui Mendes, Anália Lourenço, Sónia Carneiro, Miguel Rocha, Isabel Rocha, Eugénio C. Ferreira

***Abstract*— The analysis of cellular behavior and functionality is the most challenging aim of systems biology. The extensive analysis of the interactions between different classes of intracellular molecules reacting to genetic/environment changes can elucidate the mechanisms of regulation involved on different cellular processes. We propose a novel framework that enables the integrated analysis of metabolic and regulatory networks. The framework takes advantage on publicly available data repositories, sustaining the inference of knowledge from the integrated network. Since it is based on logic programming, it provides users with a powerful language to query information using both first and second order predicates. Also, it supports network topology analysis, motif finding and robustness evaluation. In this work, as an illustrative case study, our framework is used to build a model of the bacterium *Escherichia coli* K12.**

Keywords: Metabolic and Regulatory Networks, Data Integration, Systems Biology, Logic Programming.

I. Introduction

The study of biological networks has revealed to be very valuable in the understanding of cellular behavior, typically due to a large number of interacting components following a set of biological rules, which define their function on different processes, including biochemical reactions and genetic regulation [1].

In recent years, increasing attention has been paid to a systems-level understanding of the structure and functionality of both the metabolic and gene regulatory networks. The representation of metabolic networks by genome-scale models has been successfully used to describe all metabolic reactions taking place in the cell and predict metabolic performance for fully sequenced organisms [2][3]. Similarly, gene regulatory networks have been used to characterize the components of the regulatory system as well as their assembly [4][5].

The need for integrated metabolic and regulatory networks arises from the complex nature of biological systems [6][7][8]. The inclusion of regulatory information on metabolic models allows an insightful analysis of the behavior of the metabolic network and may help to identify evolving network properties that are not clear when focusing solely on the reaction level [9][10].

However, the construction of such models implies multi-source data integration. Data quality issues within a single source (e.g. null values, misspellings or multi-value fields) and multi-source inconsistencies (e.g. dubious EC number/enzyme name association, poor use of standard identification or common name versus multiple name aliases) demand for non-trivial computational aid in terms of data merging. On the other hand, the integrated analysis of biological components also requires enhanced computational tools capable of scaling up with the diversity and quantity of network components and all kinds of existing interactions.

In this work, we present a novel computational framework that aims to address the integrated representation and analysis of metabolic and regulatory networks. The metabolic and regulatory levels are fully described, in terms of components as well as interactions. Data integration from primary data sources results on a knowledge base encompassing pathways, metabolites (reactants/products and inhibitors/activators) and enzymes at the metabolic level, as well as genes, promoters, transcription factors, metabolic transcriptional regulators and sigma factors at the genetic level. Since the framework is based on logic programming, it provides users with a powerful language to query information using both first and second order predicates. Also, it supports network topology analysis and motif finding.

Currently, the framework is focused on prokaryotic models. Notwithstanding, it is possible to extend its knowledge base and adapt its query and analysis predicates to encompass eukaryotic models as well. Here, the construction of the integrated network for *Escherichia coli* K12 is used to exemplify the overall process.

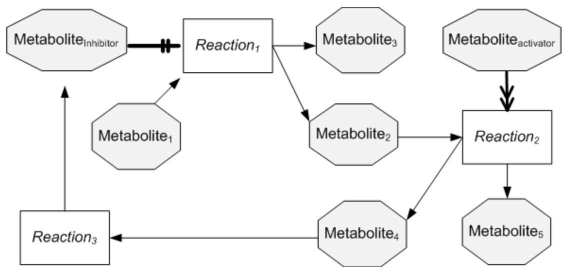

Fig. 1. Conceptual view of the biochemical network. Each square node represents a biochemical reaction transforming input metabolites (reactants) into output metabolites (products). Additionally, metabolic regulation is illustrated by metabolites acting as metabolic inhibitors (double crossed edge) and activators (double arrow edge).

R. Mendes and M.Rocha are with the Department of Informatics /CCTC, University of Minho, 4710-057 Braga, Portugal {rcm,mrocha}@di.uminho.pt

A. Lourenco, S. Carneiro, E.C. Ferreira and I. Rocha are with IBB - Institute for Biotechnology and Bioengineering, Center of Biological Engineering, University of Minho, 4710-470 Braga, Portugal {analia, soniacarneiro, irocha,ecferreira}@deb.uminho.pt

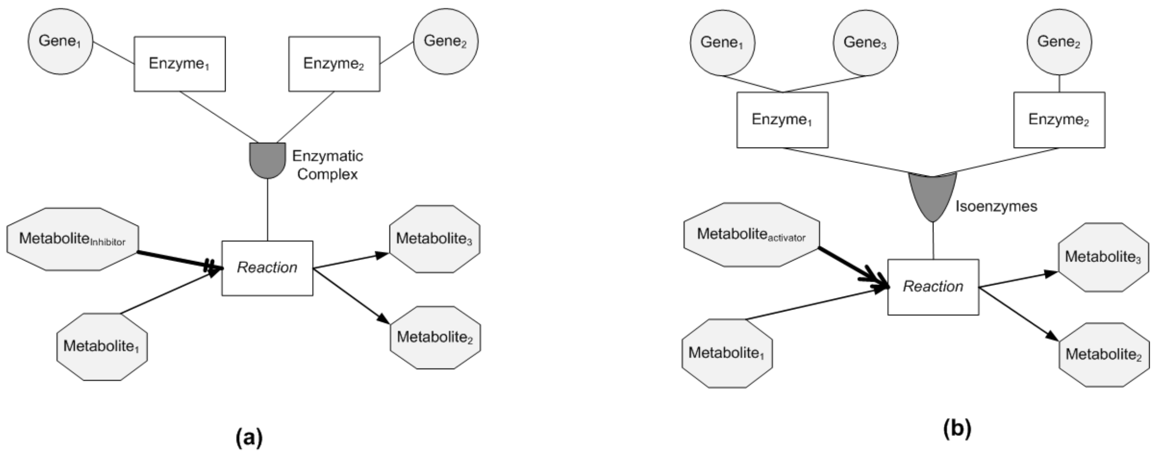

Fig. 2. Conceptual view of two particular cases of biochemical reaction catalysis. In (a) the exemplified biochemical reaction is catalyzed by an enzyme complex made by two gene products, while in (b) the catalysis of the same reaction by two different gene products (isoenzymes) is represented.

II. Genome-scale Model

A. Metabolic Level

Cellular metabolism comprises a vast set of enzymatic processes that convert various metabolites, generating biomass and energy needed to sustain cellular functions [11]. In fact, enzymes, which carry out chemical reactions or transport processes, are fundamental components in metabolic networks.

As illustrated in Figure 1, metabolic networks embrace two different classes of nodes, reactions and metabolites, and the interactions represent the participation of metabolites either as reactants/products or inhibitors/activators in metabolic reactions. The first basically represents the consumption of substrates and the production of metabolites in a enzymatic reaction. The second corresponds to the control of the enzymatic reaction, by several metabolites that can bind to enzymes activating or inhibiting the catalytic activity. This type of metabolic control is essential to adjust the excess or insufficiency of some products or reactants in the intracellular environment.

Reactions will demand for non-trivial node representation in order to encompass different enzymatic participants (Figure 2). A reaction may be simplistically viewed as being catalyzed by enzymes that consist on a single polypeptide unit (genetic product) or a complex obtained from multiple polypeptide subunits (case (a) in Figure 2). Nevertheless, there is also the case when two or more enzymes are able to catalyze the same reaction (case (b) in Figure 2). These enzymes are called isoenzymes and they differ in the amino acid sequence since they are transcribed from different genes. As a consequence of their different amino acid sequences isoenzymes may have different physicochemical properties and their transcription from different genes may lead to different regulation.

B. Genetic Level

While at the metabolic level all the possible resources to convert metabolites are represented, it is understandable that not all enzymes are available in the cell at the same time. For example, cells do not synthesize enzymes needed for L-arabinose consumption unless this compound is present in the cultivation medium. Thus, besides the metabolic control through inhibitors/activators, metabolism is also controlled by the availability of certain gene products.

Cells usually control the expression of enzyme-coding genes by regulating the transcription initiation through regulatory proteins, including sigma factors (σ factors) and transcription factors (TFs) (Figure 3). Sigma factors are essential transcription initiating factors that allow specific binding of RNA polymerase to gene promoters. Transcription factors bind to target sequences on the so-called *cis*-regulatory regions of genes, stimulating or repressing the RNA polymerase activity. TFs encoding genes can also be regulated by other transcription factors and most importantly by input signals, such as sensor proteins responding to environmental changes and metabolites, which bind directly to TFs. These TF binding metabolites are identified in this work as ”transcriptional effectors”, i.e., metabolic transcriptional regulators and those metabolites can also be involved in metabolic reactions as reactants/products.

C. Integrated Network

In the graph representation of the integrated network, biochemical reactions are built up as nodes mediating the connection between reactants and products of the enzymatic reaction. The metabolic edges are directed and categorized as reversible or irreversible depending on the reversibility/irreversibility of the corresponding reaction. Other edges represent the enzymatic regulatory events (activation/inhibition) connecting metabolites to enzymatic reactions. The integration relies on the connection of reaction nodes to gene products with non-directed edges, representing the catalysis of the reaction by either enzymes, enzymatic complexes or isoenzymes. The decision of which gene products are expressed or not, is set by the action of sigma factors and transcription factors activating or repressing the transcription. The control of metabolism involves regulation

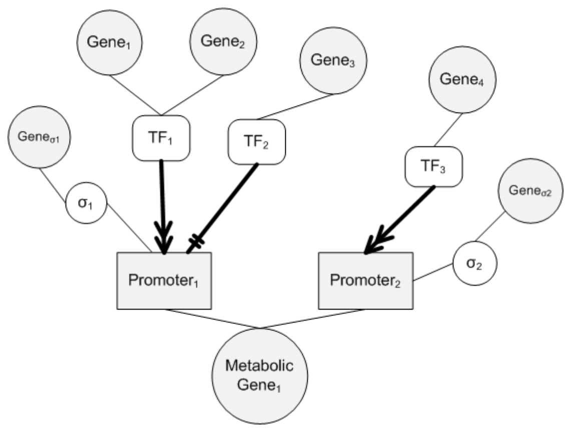

Fig. 3. Conceptual view of the gene regulatory network. Metabolic genes are regulated by transcription factors and sigma factors through promoters. Transcription factors and sigma factors are associated to the corresponding encoding genes. Whenever there is more than one promoter, regulation is determined by the presence of particular sigma factors. The case where a transcription factor regulates another transcription factor is not represented here.

of individual reactions at various levels, from enzymatic activation/inhibition by small molecules, to transcriptional regulation of enzyme-coding genes at the genetic level. Therefore, at the metabolic genes, Boolean rules represent gene regulation (i.e. encompassing transcription factor, sigma factor and transcriptional effector regulations) and enzymatic activity (i.e. encoding genes for polypeptides, complexes and isoenzymes). For the example illustrated in Figure 3, the Boolean rule is (p_1 OR p_2), i.e., ((σ_1 AND TF_1 AND NOT TF_2) OR (σ_2 AND TF_3)), when detailing genetic regulation.

III. Conciliating Data on *E. coli*

The manually-curated stoichiometric model developed by [12] and the publicly available repositories EcoCyc[1], BRENDA[2] and RegulonDB[3] supported the reconstruction of the metabolic and genetic networks of *E. coli*.

The stoichiometric model is the core of the metabolic model, supplying the following data: identification of the biochemical reactions, determination of the reaction stoichiometry and definition of metabolites compartmentalization and reaction localization. The description of biochemical reactions includes the chemical equation, the common name of the enzyme(s), the corresponding EC number(s) and the genetic Boolean rule associated with the enzyme-coding genes. The stoichiometric data and cellular localization (e.g. periplasm, citoplasm or extracellular) is extracted from the chemical equations and further information on activating and inhibiting compounds is retrieved from BRENDA.

RegulonDB provided information on genetic regulation, namely: transcription factor coding genes (regulatory genes) and regulation (regulated genes), and sigma factor coding genes and regulation. Information on promoters was retrieved from EcoCyc based on the genetic Boolean rules whereas transcriptional effectors were manually collected from literature. Gene and promoter data are linked through transcription unit data. Additional manual curation was required when multiple regulatory interactions (RI functions) were assigned for a given transcription factor/gene regulation and for associating sigma factors with multiple promoter regions.

The integration process undertook some challenging issues both at the biochemical and genetic levels. The data from the stoichiometric model and BRENDA were merged through EC numbers and enzyme common names. The EC number is a standard identification and was therefore preferred over the enzyme's name. Unfortunately, this identification is not known for almost half of the reactions, leading to shallow reaction search attempts based on common names. The genes involved on the catalysis of a reaction are delivered by the stoichiometric model in the form of a genetic Boolean rule. Thus, it is fairly simple to identify different enzymatic participation (e.g. polypeptids, complexes or isoenzymes). However, for the same EC number, there might be different Boolean rules associated, because EC number is related to the enzymatic activity rather than the particular structure of an enzyme. As such, the relationship between EC number or common enzyme name and Boolean rule is not bidirectional.

Currently, the metabolic and genetic components of the network consist of:

- 2077 enzymatic reactions for 30 pathways, involving 1276 different metabolites;
- 1037 metabolites participating as reactants or products and 435 acting as enzymatic inhibitors or activators;
- 1243 direct transcriptional interactions between transcription factors and the genes they regulate;
- 914 regulated genes from a total of 1805 cataloged genes;
- 158 transcription factors and 165 regulatory genes;
- 1350 gene-sigma factor associations for 7 sigma factors;
- 69 transcriptional effectors.

IV. Logic-Based Framework

The main goal of this work is to provide a framework to allow scientists to easily represent their data. This body of knowledge entails information about reactions, pathways, metabolites, genes, enzymes, transcription factors, sigma factors, transcriptional effectors and so forth. The framework enables scientists to easily query the system to find relationships in the data. Prolog was chosen since it is a logic language that provides many advantages for such a task:

- It is well suited to represent relationships between data and to easily query the database (e.g., to find all the genes that are activated by a given transcription factor);
- It allows users to write queries in a declarative fashion, i.e., users may query the system by simply describing the relationships they are trying to find;
- It is a high-level language: users may easily write their queries in a way that is very easy to understand;
- It is a very powerful language: a few lines of code are enough to find complex relationships and can be written

[1]http://www.biocyc.org/
[2]http://www.brenda-enzymes.info/
[3]http://regulondb.ccg.unam.mx/

in a matter of minutes.

The database is divided in three conceptual layers. One is concerned with the metabolite level, describing pathways and their reactions, products, reactants and metabolic inhibitors and activators; the intermediate level is the integration layer that has information about the enzymes (and their coding genes) which catalyze metabolic reactions and about the interaction between metabolites and the transcription factors (denoted transcriptional inducers in this database); finally, the genetic layer collects all the information about regulatory genes, transcription factors, promoters, regulated genes and sigma factors.

The next sections will explain each of these layers in detail, listing the used predicates and explaining their usage.

A. Metabolic Layer

- *in_pathway(ReactionName, Pathway)* - Identifies the pathway to which the reaction belongs.
- *consumes(MetaboliteDescription, ReactionDescription)* - Describes that a given metabolite is consumed by a reaction. MetaboliteDescription is a fact of the form *met(Metabolite, CellRegion)* with the name of the metabolite and the cellular localization (c, p, or e for cytoplasm, periplasm or extracellular). *ReactionDescription* is either the name of a reaction or a fact of the form *reversible(ReactionName, Direction)* where direction is either left or right. If the reaction is reversible, there will be two facts, one for each direction.
- *produces(MetaboliteDescription, ReactionDescription)* - Describes that a given metabolite is consumed by a reaction. The arguments are similar to *consumes*.
- *inhibits(Metabolite, ReactionName)* - Describes that a given metabolite inhibits a reaction, given the name of the metabolite and the name of the reaction.
- *activates(Metabolite, ReactionName)* - Describes that a given metabolite activates a reaction.

To illustrate the use of these predicates, we present a small excerpt of the information compiled. In the case of reaction `ACKr`, metabolites `acetate` and `atp` are consumed and `acetyl phosphate` and `adp` are produced (in the *left* direction) or `acetyl phosphate` and `adp` are consumed and `acetate` and `atp` are produced (in the *right* direction). Reaction `ACKr` is inhibited by eight metabolites.

```
in_pathway('ACKr','pyruvate metabolism').
inhibits('acetyl phosphate','ACKr').
inhibits('li+','ACKr').
inhibits('mercuric chloride','ACKr').
inhibits('n-ethylmaleimide','ACKr').
inhibits('na+','ACKr').
inhibits('p-chloromercuribenzoate','ACKr').
inhibits('p-mercuribenzoate','ACKr').
inhibits('propionate','ACKr').
consumes(met('acetate',c),
  reversible('ACKr','left')).
consumes(met('atp',c),
  reversible('ACKr','left')).
consumes(met('acetyl phosphate',c),
  reversible('ACKr','right')).
consumes(met('adp',c),
  reversible('ACKr','right')).
produces(met('acetyl phosphate',c),
  reversible('ACKr','left')).
produces(met('adp',c),
  reversible('ACKr','left')).
produces(met('acetate',c),
  reversible('ACKr','right')).
produces(met('atp',c),
  reversible('ACKr','right')).
```

B. Integration Layer

- *reaction_has_enzymes(ReactionName, Enzyme)* - Identifies the enzymes that catalyze a given reaction. The enzyme is a tuple of the form *enzyme(Name, BooleanRule)* with the identifier of the enzyme and the Boolean rule giving the set of genes involved in the enzyme's coding. If a reaction has more than one *reaction_has_enzymes fact* we are describing isoenzymes and the Boolean rule is obtained by the disjunction of the Boolean rules for each isoenzyme.
- *transcriptional_effector(TranscriptionFactor, BooleanRule)* - Associates a Boolean rule to a given transcription factor expressing which metabolites affect its functioning.

The integration layer can be demonstrated by the following example:

```
reaction_has_enzymes('ACKr',
  enzyme('acetate kinase',b3115)).
reaction_has_enzymes('ACKr',
  enzyme('acetate kinase',b2296)).
reaction_has_enzymes('ACKr',
  enzyme('acetate kinase',b1849)).
reaction_has_enzymes('ACLS',enzyme(
 'acetolactate synthase',(b0077 and b0078))).
reaction_has_enzymes('ACLS',enzyme(
 'acetolactate synthase',(b3670 and b3671))).
transcriptional_effector('AlaS','l-alanine').
transcriptional_effector('AllR','glyoxylate').
transcriptional_effector(CdaR,
  glyc-r or galct-d or manglyc or glcr)
transcriptional_effector('FNR',not 'o2').
```

The ACKr reaction is catalyzed by isoenzymes acetate kinase encoded by b3115, b2296 and b1849. Other reactions are catalyzed by enzymes which are encoded by more than one gene, producing a complex protein. As exemplified, the ACLS reaction is catalyzed by two isoenzymes identified as acetolactate synthases. The subunits of the first isoenzyme are encoded by genes b0077 and b0078 and of the second one are encoded by genes b3670 and b3671. The integration layer includes also the influence of metabolites at the genetic regulation. In some cases, transcription factors' activity is affected through metabolite binding changing the expression of target genes.

C. Genetic Regulation

- *transcription_factor(RegulatoryGene, TranscriptionFactor)* - Identifies the gene that encodes a given transcription factor.
- *genetic_regulation(TranscriptionFactor, Promoter, RIFunction)* - Identifies the genetic regulation, the action of a given TF and its effect (usually inhibition (-) or activation (+)) over a given promoter.
- *promoter(Promoter, RegulatedGene)* - Identifies the relationship between the promoter and the regulated gene.
- *sigma_factor(CodingGene, Sigma)* - Identifies the sigma factor's name and its coding gene.

- *sigma_regulation(Promoter, Sigma)* - Associates a sigma factor with a given promoter assuming that it has always a positive role.

In the previous example, the ArcA transcription factor encoded by gene b4401, activates the ackAp promoter and inhibits the aceEp promoter. Sigma factor 28, encoded by fliA, also affects transcription of several genes, like genes under the control of aerp and flgKp.

```
transcription_factor(b4401, 'ArcA').
genetic_regulation('ArcA', aceEp, -).
genetic_regulation('ArcA', ackAp, +).
promoter(aceEp, b0114).
promoter(aceEp, b0115).
promoter(ackAp, b2296).
sigma_factor(fliA, 'Sigma28').
sigma_regulation(aerp, 'Sigma28').
sigma_regulation(flgKp, 'Sigma28').
```

Based on this knowledge, the framework is able to automatically generate the regulatory Boolean rules for all metabolic genes.

V. Query Examples

A. Querying the Database

To obtain information about the facts in the database, one simply has to write the fact using variables (denoted in Prolog by a capitalized name) at the prompt. Thus, supposing one wants to find the metabolites that are produced by reaction ACLS, one simply has to write the following query:

```
?- produces(M, 'ACLS').
M = met('(s)-2-acetolactate', c)
```

Note that you only get the first answer. However, Prolog expects input from the user at the end and, if the user wants other answers, he/she simply needs to input a semicolon to ask for an alternative answer. In this way, we can find all the metabolites that are produced by this reaction.

```
?- produces(M, 'ACLS').
M = met('(s)-2-acetolactate', c) ;
M = met(co2, c) ; fail.
```

B. Second Order Predicates

If we are interested in finding all the metabolites that are produced by a given reaction, we could ask Prolog to give us all solutions at the same time by using one of the second order predicates. A second order predicate collects all the results to a given predicate in a list. We could thus ask the system for all the metabolites produced in the cytoplasm by reaction ACLS:

```
?- setof(M, produces(met(M, c),
   'ACLS'), Metabolites).
 Metabolites = ['(s)-2-acetolactate', co2].
```

The setof predicate has some interesting characteristics: if there are unbound variables in the predicates that are not collected in the first argument, setof will instantiate (i.e., assign a value) them with one of the possible values and only give the alternatives in this case. To find alternatives for other values of the variables, we have to ask Prolog to give us the alternatives explicitly.

```
setof(M, produces(met(M, CellRegion),
       'AGt3'), Metabolites).
CellRegion = c, Metabolites = ['h+'] ;
CellRegion = e, Metabolites = [silver].
```

In case we wanted to find all the metabolites produced independently of where they were produced, we would existentially quantify the variable CellRegion.

```
?- setof(M,
|     CellRegion^produces(met(M, CellRegion),
       'AGt3'), Metabolites).
 Metabolites = [ag, 'h+'].
```

C. Conjunction and Disjunction

We are usually interested in combining information about different entities. For instance, we could try to find the names of the reactions that produce a metabolite and are inhibited by the same metabolite. The conjunction in Prolog is written by simply separating the predicates by a comma. A semicolon is used if we are interested in the disjunction:

```
?- produces(met(M, c), R), inhibits(M, R).
M = '(r)-pantothenate', R = 'PANTS'
```

If we are interested in finding all the reactions that produce and are inhibited by a certain metabolite, we would use a second order predicate:

```
?- setof(R, M^(produces(met(M, c), R),
  inhibits(M, R)), Rs).
Rs = ['ABUTD', 'ACGAMK', 'ACGS'|...].
```

Suppose we want to find the names of the reactions that either produce or consume a certain metabolite:

```
?- setof(R, CellRegion^(
|        consumes(met(Met, CellRegion), R);
|        produces(met(Met, CellRegion), R)),
|    Reactions), length(Reactions, N).
Met = '12dgr120',
Reactions = ['12DGR120tipp', 'DAGK120',
  'PAPA120', 'PAPA120pp'], N = 4
```

As exemplified, metabolite '12dgr120' participates in four reactions as reactant or product.

D. Rules

It makes sense to write rules that will spare us having to write very complex queries or simply saving them for later use. Let us suppose that we are interested in finding pairs of reactions where one reaction produces a given metabolite and the other consumes it. Let us write a rule, called edge with three arguments to denote this relation:

```
edge(R1, R2, Met):-
        produces(met(Met, CellRegion), R1),
        consumes(met(Met, CellRegion), R2).
```

Now we can use this predicate in a query:

```
?- edge(R1, R2, M).
R1 = '12DGR120tipp', R2 = 'DAGK120',
M = '12dgr120'
```

We can identify that metabolite '12dgr120' is produced in reaction '12DGR120tipp' and consumed in reaction 'DAGK120'.

E. More Examples

Suppose we want to find the names of the genes that are regulated by a given TF (e.g. 'AcrR'). The following query finds all promoters P that are inhibited by AcrR, the bnumber of the genes that are regulated by these promoters and finally, the common name associated to the bnumber. As demonstrated in the following example, the transcription factor AcrR inhibits genes acrA and acrR.

```
?- setof(Name, P^G^(
|        genetic_regulation('AcrR', P, '-'),
|        promoter(P, G),
|        gene_alias(G, Name)
| ), L).
L = [acrA, acrR]
```

A more complex example would be to find metabolites acting both as an enzymatic activactor/inhibitor and also as a transcriptional effector, which affects the genetic regulation of a protein coding gene associated with the same reaction. As represented below, metabolite 'galct-d' is simultaneously an enzymatic inhibitor of glucarate dehydratase that catalyzes the GLCRD reaction, and a transcriptional effector that affect transcriptional activity of CadR, which activates the enzyme coding genes of the same reaction (GLCRD).

```
?- inhibits(M, R),
|        reaction_has_enzymes(R,
   enzyme(Name, Index, GeneRule)),
|        get_genes(GeneRule, Genes),
|        member(Gene, Genes),
|        promoter(P, Gene),
|        genetic_regulation(TF, P, RI),
|        transcriptional_effector(TF, MRule),
|        get_mets(MRule, Mets),
|        member(M, Mets).
M = 'galct-d', R = 'GLCRD',
Name = 'glucarate dehydratase',
Index = 0, GeneRule = b2787, Genes = [b2787],
Gene = b2787, P = gudPp, TF = 'CdaR', RI = +,
MRule = 'glyc-r' or 'galct-d' or manglyc or glcr,
Mets = ['glyc-r', 'galct-d', manglyc, glcr]
```

This query retrieves one of the metabolites that inhibit a reaction and then selects one of the Boolean rules (GeneRule) of one of the enzymes of the reaction; then it takes one of the Genes, the promoter that regulates the gene, the TF that regulates the promoter and the transcriptional effector rule for this promoter. Finally, it verifies that the metabolite is part of the metabolite rule.

VI. Conclusions

The construction of integrated metabolic and regulatory networks is of paramount importance to understand the overall structure and behavior of biological systems. In recent years, increasing attention has been paid to a systems-level understanding of the structure and functionality of both the metabolic and gene regulatory networks in the field of systems biology. However, the construction and analysis of integrated metabolic and regulatory networks demands for non-trivial network representation and computation algorithms. Both information querying and network analysis have to account for the number and diversity of network components and the multiple kinds of existing interactions.

Our logic-based framework aims at tackling integrated networks, providing adequate support for data extraction and merging, knowledge base query, network topology analysis, motif finding and robustness evaluation. The framework allows us to easily write queries that combine information from different levels (e.g. the genetic and metabolic level) in a high-level language that is easily understood by humans and that provides a much needed abstraction when searching for information.

We are currently using the framework for motif finding and generation of boolean rules that encompass available information about cellular functionality. This will entail the information necessary to perform automatic robustness evaluation.

References

[1] L.A.N. Amaral and J.M. Ottino. Complex systems and networks: challenges and opportunities for chemical and biological engineers. *Chemical Engineering Science*, 59:1653–1666, 2004.

[2] J.L. Reed, T.D. Vo, C.H. Schilling, and B.O. Palsson. An expanded genome-scale model of escherichia coli k-12 (ijr904 gsm/gpr). *Genome Biology*, 4:R54, 2003.

[3] R.A. Notebaart, F.H.J. van Enckevort, C. Francke, R.J. Siezen, and B. Teusink. Accelerating the reconstruction of genome-scale metabolic networks. *BMC Bioinformatics*, 7:296, 2006.

[4] M.M. Babu and S.A. Teichmann. Evolution of transcription factors and the gene regulatory network in Escherichia coli. *Nucleic Acids Research*, 31(4):1234–1244, 2003.

[5] Radu Dobrin, Qasim K. Beg, Albert-László Barabási, and Zoltán N. Oltvai. Aggregation of topological motifs in the escherichia coli transcriptional regulatory network. *BMC Bioinformatics*, 5:10, 2004.

[6] S.H. Strogatz. Exploring complex networks. *Nature*, 410:268–76, 2001.

[7] Z. Jinq, Y. Hong, L. Jianhua, Z.W. Cao, and Y.X. Li. Complex networks theory for analyzing metabolic networks. *Chinese Science Bulletin*, 51:1529–37, 2006.

[8] M.J. Herrgård, B.S. Lee, V. Portnoy, and B.Ø. Palsson. Integrated analysis of regulatory and metabolic networks reveals novel regulatory mechanisms in Saccharomyces cerevisiae. *Genome Research*, 16(5):627–635, 2006.

[9] M.W. Covert and B.O. Palsson. Transcriptional Regulation in Constraints-based Metabolic Models of Escherichia coli. *Journal of Biological Chemistry*, 277(31):28058–28064, 2002.

[10] I. Shmulevich, E.R. Dougherty, and W. Zhang. From Boolean to probabilistic Boolean networks as models of genetic regulatory networks. *Proceedings of the IEEE*, 90(11):1778–1792, 2002.

[11] A. Gutteridge, M. Kanehisa, and S. Goto. Regulation of metabolic networks by small molecule metabolites. *BMC Bioinformatics*, 8(1):88, 2007.

[12] A.M. Feist, C.S. Henry, J.L. Reed, M. Krummenacker, A.R. Joyce, P.D. Karp, L.J. Broadbelt, V. Hatzimanikatis, and B.Ø. Palsson. A genome-scale metabolic reconstruction for escherichia coli k-12 mg1655 that accounts for 1260 orfs and thermodynamic information. *Molecular Systems Biology*, 3(121), 2007.