Search and Rescue: logic and visualisation of biochemical networks

Nicos Angelopoulos[1], Paul Shannon[2], and Lodewyk Wessels[1]

[1] Netherlands Cancer Institute, Amsterdam, Netherlands
[2] Fred Hutchinson Cancer Research Center, Seattle, USA
n.angelopoulos@nki.nl

Abstract. We utilise a recently introduced Prolog package that allows communication with the *R* statistical software to develop a graph-centric suit of procedures that allow the exploration of biological data and hit-lists from within Prolog. We show how a number of public protein-protein interaction databases can be intuitively be represented as facts before we present search algorithms that are naturally expressed in logical terms. Visualisation of the resulting graphs is done via low-level communication to *Bioconductor's RCytoscape* package. We illustrate the utility of the integrative Prolog platform using two public databases and a graph search to connect elements of genes involved in cell motility.

1 Introduction

Constraint logic programming is a powerful yet under-valued platform for research and analysis in bioinformatics and computational biology. Scripting, high-level of abstraction, interpreter-base and automatic memory management are all features of logic programming (LP) that make it ideal for the development of research-led code in the above areas.

The areas in which LP is deficient are: the lack of user-based package extensions as typified by code repositories, the limited number of statistical packages and its graphical abilities. The deployment of LP in areas that have a strong ethos with regard to backing conceptual ideas and research results with functioning code will help create the necessary conditions for code repositories. This has already been born out for the communities of *R* [7], *Perl* and *Python*. Where, their use in bioinformatics has substantially bolstered their community-contributed code base. One might argue, that a surge in the use of LP in computational biology would be beneficial for its expansion. This is seriously hampered by the second and third shortcomings, i.e. the lack of statistical reasoning and graphical output software suits. The use of LP has been previously argued and used in bioinformatics and particular in ontology reasoning [6].

In this paper we take advantage of recent developments in integrating *R* within Prolog [1] to explore graph searching and visualisation within logic programming. The strengths of Prolog in data representation and search are put into representing and reasoning with biological knowledge. Furthermore, the complementary strengths of *R* in visualisation are brought to bear within a logic programming environment, thus doing away with need of re-implementing such procedures.

The remainder of the paper is organised as follows. Section 2 describes protein-protein interactions networks (PPIs). Section 3 presents graph operations on gene lists in the context of PPIs and develops these ideas on a specific PPI and motility gene list. The paper's concluding remarks are in the Section 4.

2 Protein-Protein interaction networks

The last decades have witnessed a phenomenal increase in the amount of biological knowledge that has been published and codified. This acceleration can be directly attributed to the evolution of high throughput technologies such as genome wide expression assays, microscopy, and deep sequencing.

One important way in which biological knowledge is codified is in the form of protein-protein interaction (PPI) databases such as STRING [9] and HPRD [4]. STRING collates information from a variety of sources including predicted interactions and gives weight scores to each edge based on the strength of the evidence supporting the corresponding interaction. It currently contains information on $5,214,234$ proteins from 1133 organisms and holds $224,346,017$ interactions. HPRD holds human proteins and interactions between them. Currently there are $39,194$ interactions in HPRD.

Apart from direct interactions it is also straight forward to represent interactions passing through metabolites, such as the interactions present in the metabolic pathways present in the KEGG database [3]. Visualising PPIs are often in the form of networks/graphs which provide an overall picture of the connectivity between the various pathways mapping biological functions.

Representing these types of interactions depends to a large extent on the operations one plans to perform. One way in which directional HPRD interactions can be stored is as interaction facts:

```
interaction( From, To, Types, References ).
```

Proteins *From* and *To* are mapped to Entrez IDs and *Types* are the types of evidence provided in *References*. Alternatively, when type of interaction needs to be explicitly represented, as is the case in our KEGG database example below, one can represent the separate reaction types as separate facts:

```
activation( From, To, Organism, Pathway ).
inhibition( From, To, Organism, Pathway ).
phosphorylation( From, To, Organism, Pathway ).
ubiquination( From, To, Organism, Pathway ).
```

The represented interaction of the involved proteins is known to occur in the specified *Organism* and is part of the KEGG pathway identified by $Pathway$.

As regulatory, metabolic and signalling networks become better known, due largely to advances in laboratory technique, we will nonetheless be faced by the condition-specific, and cell-line specific nature of all molecular interactions within the cell. Logic programming offers new capabilities to understand these interactions, and in our efforts to predict and control cellular behaviour.

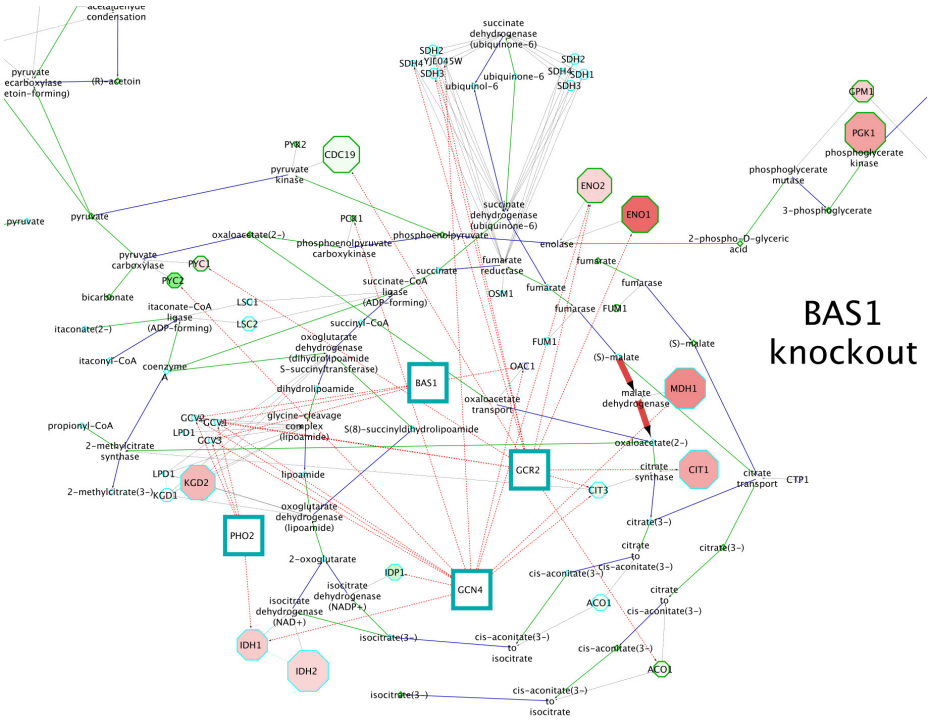

Fig. 1. Transcription factors control metabolic pathways activity in yeast [2].

The complexity and contingency of regulatory networks is an emerging, recurrent theme demonstrated in a plethora of computational biology papers. For instance, [2] describes a combination of metabolite flux measurements and protein abundance across 119 transcription factor knockout strains in yeast, to identify a small number of transcription factors which regulate a crucial step in the TCA cycle. Using *RCytoscape*, we displayed these on top of the yeast consensus carbon cycle metabolic network (`http://www.comp-sys-bio.org/yeastnet/`).

Logic programming is a natural complement to this network visualisation of the data in Fendt et. al [2]- which is multi-dimensional, and representative of the rich networks which will become increasingly available. Representing and reasoning with the multiple levels of such networks that include regulatory, metabolic and signalling components can be an important future research area for logic programming. Its AI heritage can be put in good use in elucidating the intricate details of such structures and inferentially associate or predict the outcome of interventions.

3 Gene-lists in Graphs

A variety of bioinformatics analyses have as an end or intermediate product the generation of a list of genes. Visualising these lists in the context of protein interaction

Fig. 2. KEGG interactions for a subset of the members of an adhesome library. Nodes are proteins and edges denote interactions. Blue nodes are connecting proteins that do not appear in the library. Edges are coloured as per type of interaction.

networks is a useful tool through which results can be presented. Visualisation is particularly powerful when communicating results to experimental biologists.

Cell motility is a complex biological process that plays a critical role in development, wound healing and cancer metastasis. It is mediated by focal adhesion complexes which are dynamic structures that may involve a large number of proteins. There exists a large body of literature that studies the molecular mechanisms by which cells move in-vitro and in-vivo. The main core of the proteins involved in complexes has been placed by some studies to 156 [11] while when considering potential encoding genes for the whole motility apparatus, the total number can be substantially greater. Such broadly defined libraries play an important role in screening programmes. Here we explore a set of 570 motility related genes that were gathered from a variety of sources [10].

3.1 Graph operations

We mapped the list of motility genes to the KEGG interaction pathways. We construct a graph by adding an edge between any pair of genes with a known interaction in KEGG. As only a limited number of pairs have direct interactions we extended the graph by implementing a depth first search algorithm based on the representation of KEGG interactions we already discussed. The most connected sub-graph is designated as the seed of the main graph and attempts are made to expand it. For each of the remaining sub-graphs or disconnected motility genes, a breadth first attempt is made to connect

it to the main graph by adding n additional nodes. If n such nodes can be found, the current sub-graph is removed from the list of sub-graphs to be connected, otherwise the algorithm repeats the test for $n = n + 1$. The algorithm is greedy in that it only analyzes the first extending path of n additional genes it encounters. The sub-graph is removed from the list of sub-graph to be connected if there exists no n such that it can be connected to the main graph. The algorithm terminates when it encounters an empty sub-graphs list. The implementation of the algorithm in Prolog is elegant and easy to communicate and maintain. We hope it will be added to the standard graph operations library that exists in many Prolog systems.

3.2 Visualisation software

The results of the algorithm on the motility list are shown in Figure 2. Note that not all genes can be connected in this case. Some of these disconnected genes are shown at the bottom of the graph. Visualisation has been one of the areas in which Prolog has been weak. Here we utilise *r..eal* [1], a recently developed Prolog library that allows efficient interactions with the *R* statistical software system. We developed software that allows the visualisation of Prolog graphs via the *RCytoscape Bioconductor* package [8]. Our interface code can be used with Prolog represented graphs from the standard graph library and provides convenient options for rendering a variety of aspects for all graph elements. It is worth noting that the interaction between Prolog and *Cytoscape* is bi-directional. Sets of nodes selected via the graphical interface can be interactively accessed via Prolog.

4 Conclusions

We have argued in this paper that Prolog is a power platform for data analysis and computational research in biofinformatics. Biological knowledge can be succinctly represented and reasoned about within logic programming which has traditional strengths in artificial intelligence research and provides a high-level at which one can interact with biological datasets.

In addition, we present practical steps towards the promotion of logic programming in the manipulation and visualisation of biochemical networks and associated gene lists. Graph operation on such lists are crucial to communicating results of analysis to experimentalists but can also provide the basis for further analysis. For example, in network based regression algorithms [5]. Our software is a useful addition to existing Prolog code in the bioinformatics domain [6].

References

1. Angelopoulos, N., Costa, V.S., Azevedo, J., Camacho, R., Wessels, L.: Integrative statistics for logical reasoning. In preparation, `http://bioinformatics.nki.nl/~nicos/sware/real` (2012)
2. Fendt, S.M., Oliveira, A.P., Christen, S., Picotti, P., Dechant, R.C., Sauer, U.: Unraveling condition-dependent networks of transcription factors that control metabolic pathway activity in yeast. Mol Syst Biol **6** (2010)

3. Kanehisa, M., Goto, S., Sato, Y., Furumichi, M., Tanabe, M.: Kegg for integration and interpretation of large-scale molecular datasets. Nucleic Acids Res. (2012) D109–D114
4. Keshava Prasad, T.S., Goel, R., Kandasamy, K., Keerthikumar, S., Kumar, S., Mathivanan, S., Telikicherla, D., Raju, R., Shafreen, B., Venugopal, A., Balakrishnan, L., Marimuthu, A., Banerjee, S., Somanathan, D.S., Sebastian, A., Rani, S., Ray, S., Harrys Kishore, C.J., Kanth, S., Ahmed, M., Kashyap, M.K., Mohmood, R., Ramachandra, Y.L., Krishna, V., Rahiman, B.A., Mohan, S., Ranganathan, P., Ramabadran, S., Chaerkady, R., Pandey, A.: Human protein reference database2009 update. Nucleic Acids Research **37**(suppl 1) (2009) D767–D772
5. Maathuis, M.H., Colombo, D., Kalisch, M., Bhlmann, P.: Predicting causal effect in large-scale systems from observational data. Nature Methods **7**(4) (2010) 247–248
6. Mungall, C.: Experiences using logic programming in bioinformatics. In Hill, P., Warren, D., eds.: Logic Programming, 25th International Conference, ICLP 2009. Volume 5649 of Lecture Notes in Computer Science. Springer (2009) 1–21
7. R Development Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. (2011)
8. Shannon, P.: RCytoscape: Display and manipulate graphs in Cytoscape. (2012) R package version 1.6.3.
9. Szklarczyk, D., Franceschini, A., Kuhn, M., Simonovic, M., Roth, A., Minguez, P., Doerks, T., Stark, M., Muller, J., Bork, P., Jensen, L.J., Mering, C.v.: The string database in 2011: functional interaction networks of proteins, globally integrated and scored. Nucleic Acids Research **39**(suppl 1) (2011) D561–D568
10. van Roosmalen, W.: Motility involved genes (2012) Personal communication.
11. Zaidel-Bar, R., Itzkovitz, S., Ma'ayan, A., Iyengar, R., Geiger, B.: Functional atlas of the integrin adhesome. Nature Cell Biology **9**(8) (08 2007) 858–867