Inferring the function of genes from synthetic lethal mutations

Ray, O and Bryant, CH

http://dx.doi.org/10.1109/CISIS.2008.124

Inferring the function of genes from synthetic lethal mutations

O. Ray
University of Bristol
Department of Computer Science, Woodland Road
Bristol, BS8 1UB, United Kingdom
Email: oray@cs.bris.ac.uk

C.H. Bryant
The Robert Gordon University
School of Computing, St Andrew Street
Aberdeen, AB25 1HG, United Kingdom
Email: chb@comp.rgu.ac.uk

***Abstract*—Techniques for detecting synthetic lethal mutations in double gene deletion experiments are emerging as powerful tool for analysing genes in parallel or overlapping pathways with a shared function. This paper introduces a logic-based approach that uses synthetic lethal mutations for mapping genes of unknown function to enzymes in a known metabolic network. We show how such mappings can be automatically computed by a logical learning system called eXtended Hybrid Abductive Inductive Learning (XHAIL).**

I. Introduction

The advent of high-throughput laboratory techniques have contributed to a detailed knowledge of biological activity at the genomic and metabolomic levels. DNA sequencing has revealed the genetic code of many organisms, and metabolic studies have exposed the enzyme-catalysed biochemical pathways by which these organisms survive. Yet a major challenge still facing biologists is to achieve a more comprehensive understanding of cellular operation by considering the interplay between genomic and metabolomic phenomena. The most basic task in this enterprise is determining the function of genes by identifying which enzymes they encode [1].

Functional genomics has traditionally employed methods based on knocking out single genes from an organism and using auxotrophic growth experiments to study the resulting phenotype [1]. This involves comparing the viability of mutant strains on synthetic growth media to obtain clues about the enzyme encoded by the deleted gene. But recent studies have shown the limitations of single gene deletions and prompted new techniques for detecting epistatic interactions in double mutant strains [2]. These new approaches can uncover so-called synthetic lethal deletions — i.e., pairs of mutations that are lethal in combination, but viable in isolation.

This paper introduces a logic-based approach for inferring the function of genes from synthetic lethal mutations. The work builds upon a model of metabolism used in an earlier Robot Scientist project [3]. Like the Robot Scientist, our aim is to map genes of unknown function to enzymes in a known metabolic network. But, unlike the Robot Scientist, our method is able to represent and reason about double gene deletions. We show how this can be accomplished using a logical learning system called eXtended Hybrid Abductive Inductive Learning (XHAIL) [4].

II. Background

A. Metabolic Pathways

Living organisms import nutrients from their environment in order to synthesise the essential molecules necessary for their survival [1]. The conversion of nutrients into complex molecules is achieved by sequences of biochemical reactions known as metabolic pathways. Each step in a metabolic pathway usually represents a small change to the molecule and is mediated by a specific enzyme. Enzymes are biological catalysts that speed up reactions by many orders of magnitude so they can support the processes of life. A pathway can be identified by the sequence of enzymes it contains.

In general, enzymes catalyse reactions in which one set of molecules, called substrates, are converted into another set of molecules, called products. For simplicity, this paper will only consider reactions that transform a single substrate into a single product. All of the substrates and products which appear in a metabolic pathway are called metabolites. Metabolic pathways often intersect on shared metabolites to create complex graphs known as metabolic networks.

Fig. 1 illustrates a metabolic network in which two nutrients nut_1 and nut_2 are converted into one essential molecule ess_7 via four intermediate molecules mol_3-mol_6. Each arrow represents one reaction in which the metabolite at the tail is transformed into the metabolite at the head via the enzyme on the side. In all, there are seven reactions catalysed by seven enzymes enz_a-enz_g. This network shows that ess_7 can be synthesised from either nut_1 (via pathways $aceg$ and $adfg$) or from nut_2 (via pathway bfg).

B. Auxotrophic Growth Experiments

The expression of each enzyme in a metabolic network is regulated by one or more genes which are said to code for that enzyme [5]. Modern experimental methods allow mutant strains to be created by selectively deleting individual genes from the genome of an organism. In some species, such as the yeast S. cerevisiae, whole libraries of single gene deletion mutants are readily available [6]. If a deleted gene encodes an enzyme necessary for the synthesis of some essential molecule, the corresponding mutant is said to be auxotrophic (i.e., unable to produce that molecule).

Auxotrophic growth experiments are a classical way to discover clues about the compromised enzyme by comparing the viability of mutant strains with the wild type on different growth media supplemented with or deprived of carefully chosen nutrients [1]. Previous work has shown such tests can be automated and interpreted by an intelligent Robot Scientist platform [3]. For example, referring to Fig. 1, if a single deletion mutant survives in a medium containing both nut_1 and nut_2, but does not survive in a medium containing just nut_1, then the deleted gene must encode enz_a.

More recent studies have begun to reveal the limitations of single mutant experiments [2]. The problem is that biological systems have evolved robust mechanisms to compensate for single points of failure. These include alternative pathways for synthesing the same molecule and multiple genes for coding the same enzyme. Thus 80% of deletions in the yeast genome are non-essential for survival [7] and have a measurable effect only under very specialised growth conditions [6] which are difficult to replicate and may have unwanted side-effects such as sensitivity to synthetic growth media [8].

C. Synthetic Lethal Mutations

Deeper insights into genes that operate in parallel or overlapping pathways with a shared function can be obtained from double deletion strains [2]. This is because double mutants can uncover synthetic lethal deletions that are lethal in combination, but viable in isolation. Referring again to Fig. 1, enzymes c and d would show a synthetic lethal relation in a medium containing nut_1 but not containing nut_2. However, disabling just one would have no effect in this medium.

Industrial strength methods — such as dSLAM (diploid-based Synthetic Lethality Analysis with Microarrays) and SGA (Synthetic Genetic Array) — are now becoming available which perform genome-wide screens for synthetic lethality. These use highly developed microarray techniques that exploit genetic bar codes inserted into mutant libraries and avoid many difficulties of standard auxotrophic experiments. These techniques are reviewed in [2].

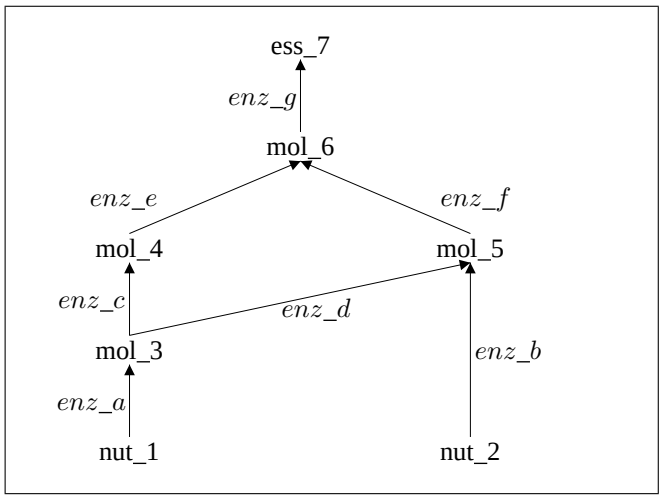

Fig. 1. A Highly Simplified Metabolic Network

D. Knowledge Representation by Logic Programs

The analysis of synthetic lethal screens can be seen as a combinatorial problem. But the vast number of possible gene-enzymes mappings means that additional knowledge and constraints must be used in practice. Logic programs provide an expressive and intuitive modelling language in which knowledge and constraints can be modularly added and refined. They also have the advantage of a formal semantics and sophisticated tool support which have proven useful in real applications of bioinformatics [3], [9].

A logical language is built from terms (which denote objects or individuals) and predicates (which denote properties or relations). The most basic logical expression is called an atom and comprises a predicate followed by a tuple of terms. For example, if nut_1, enz_a, mol_3 and $gene\_a$ are terms, and if $reaction$ and $codes$ are predicates, then the atoms $reaction(nut\_1, enz\_a, mol\_3)$ and $codes(gene\_a, enz\_a)$ can express the facts that enz_a catalyses a reaction from nut_1 to mol_3 and that $gene\_a$ codes for enz_a.

In brief, a logic program is a set of clauses of the form

$$a_0 : -l_1, \ldots, l_n.$$

where a_0 is an atom and each l_i is either an atom a_i or its negation $not\ a_i$. The atom a_0 is called the head of the clause and each l_i is called a body literal. Logically, a clause is an implication which says that, if all the body literals are true, then head atom must also be true. Clauses can contain variables which may be replaced by any terms in the language. A clause with an empty body is called a fact and denoted a_0. A clause with the head atom $a_0 = false$ is called a constraint and states the body literals $l_1, \ldots, l_n$ must not all be true. If p is a predicate and $t_1 \ldots t_n$ are terms, then $p(t_1; \ldots; t_n)$ is used to abbreviate the facts $p(t_1) \ldots p(t_n)$.

E. Hypothetical Reasoning with XHAIL

XHAIL is a hybrid reasoning system for logic programs that integrates abductive (explanation-based) and inductive (generalisation-based) inference within a common learning framework [4]. Given a logic program $\mathcal{T}$ (theory or background knowledge) and a set of ground literals $\mathcal{G}$ (goals or examples), XHAIL will return a logic program $\mathcal{H}$ (hypothesis) that, together with $\mathcal{T}$, explains $\mathcal{G}$.

The XHAIL hypothesis space is constrained by a form of language bias called mode declarations As explained in [10] mode declarations are either head declarations $modeh(r, s)$ or body declarations $modeb(r, s)$, where r is an integer (recall) and s is a ground atom (scheme). Schemes contain special placemarker terms $\#$, $+$ and $-$, which must be replaced by constants, input and output variables of a specified type.

In this way, a set of head and body declarations allow certain literals to appear in the head and body of a hypothesis clause, and their recalls limit how many literals each may contribute. By default, XHAIL will return maximally compressive hypotheses containing the fewest number of literals. This is a popular heuristic motivated by Occam's Razor, which prefers the simplest hypothesis explaining the data.

III. Model

This section introduces our logical model for analysing synthetic lethal mutations. The approach is illustrated by means of a hypothetical organism with the highly simplified metabolic network given in the previous section. In all, the model contains 31 clauses which, for convenience, are split into four groups shown in Figs. 2-5 and described below.

(1) $reaction(enz\_a, nut\_1, mol\_3).$
(2) $reaction(enz\_b, nut\_2, mol\_5).$
(3) $reaction(enz\_c, mol\_3, mol\_4).$
(4) $reaction(enz\_d, mol\_3, mol\_5).$
(5) $reaction(enz\_e, mol\_4, mol\_6).$
(6) $reaction(enz\_f, mol\_5, mol\_6).$
(7) $reaction(enz\_g, mol\_6, ess\_7).$

(8) $nutrient\_in(nut\_1, med\_1).$
(9) $nutrient\_in(nut\_1, med\_2).$
(10) $nutrient\_in(nut\_2, med\_2).$

(11) $essential\_molecule(ess\_7).$

(12) $gene(gene\_a; gene\_b; gene\_c; gene\_d;$
$gene\_e; gene\_f; gene\_g; gene\_h).$

Fig. 2. Experimental Setup

(13) $experimental\_observations :-$
$synth\_lethal\_muts(gene\_c, gene\_d, med\_1),$
$not\ synth\_lethal\_muts(gene\_c, gene\_d, med\_2),$

$synth\_lethal\_muts(gene\_d, gene\_e, med\_1),$
$not\ synth\_lethal\_muts(gene\_d, gene\_e, med\_2),$

$not\ synth\_lethal\_muts(gene\_a, gene\_b, med\_1),$
$synth\_lethal\_muts(gene\_a, gene\_b, med\_2),$

$synth\_lethal\_muts(gene\_e, gene\_f, med\_1),$
$synth\_lethal\_muts(gene\_e, gene\_f, med\_2),$

$singl\_lethal\_mut(gene\_a, med\_1),$
$singl\_lethal\_mut(gene\_g, med\_2).$

Fig. 3. Experimental Observations

(14) $goal(experimental\_observations).$

(15) $modeh(10, codes(\text{"}\#gene\text{"}, \text{"}\#enzyme\text{"})).$

Fig. 4. XHAIL Directives

A. Experimental Setup

Fig. 2 gives the background knowledge for the relevant organism and the experimental conditions. Clauses (1-7) encode the metabolic reactions from Fig. 1, while Clauses (8-10) specify the growth media used to culture the mutants. Here, there are two growth media: med_1, which contains the nutrient nut_1; and med_2, which contains nut_1 and nut_2.

(16) $synth\_lethal\_muts(Gi, Gj, D) :-$
$genes(Gi, Gj), medium(D),$
$inviable\_without\_genes(Gi, Gj, D),$
$not\ singl\_lethal\_mut(Gi, D),$
$not\ singl\_lethal\_mut(Gj, D).$

(17) $singl\_lethal\_mut(G, D) :-$
$gene(G), medium(D),$
$inviable\_without\_genes(G, G, D).$

(18) $inviable\_without\_genes(Gi, Gj, D) :-$
$genes(Gi, Gj), medium(D),$
$codes(Gi, Ei), codes(Gj, Ej),$
$essential\_molecule(M), enzymes(Ei, Ej),$
$not\ make\_without\_enzymes(D, M, Ei, Ej).$

(19) $make\_without\_enzymes(D, M, Ei, Ej) :-$
$medium(D), metabolite(M), enzymes(Ei, Ej),$
$essential\_molecule(M), nutrient\_in(N, D),$
$path\_without\_enzymes(N, M, Ei, Ej).$

(20) $path\_without\_enzymes(M, M, Ei, Ej) :-$
$metabolite(M), enzymes(Ei, Ej).$

(21) $path\_without\_enzymes(Mi, Mj, Ei, Ej) :-$
$metabolites(Mi, Mj), enzymes(Ei, Ej),$
$reaction(E, Ni, Nj), E \neq Ei, E \neq Ej,$
$path\_without\_enzymes(Mi, Ni, Ei, Ej),$
$path\_without\_enzymes(Nj, Mj, Ei, Ej).$

(22) $metabolites(Mi, Mj) :-$
$metabolite(Mi), metabolite(Mj).$

(23) $enzymes(Ei, Ej) : -enzyme(Ei), enzyme(Ej).$

(24) $genes(Gi, Gj) : -gene(Gi), gene(Gj).$

(25) $metabolite(Mi) : -reaction(E, Mi, Mj).$

(26) $metabolite(Mj) : -reaction(E, Mi, Mj).$

(27) $enzyme(E) : -reaction(E, Mi, Mj).$

(28) $medium(M) : -nutrient\_in(N, M).$

(29) $coded(E) : -enzyme(E), gene(G), codes(G, E).$

(30) $false : -enzyme(E), not\ coded(E).$

(31) $false : -gene(G), enzymes(Ei, Ej),$
$codes(G, Ei), codes(G, Ej), Ei \neq Ej.$

Fig. 5. Metabolic Theory

Clause (11) states there is one essential molecule ess_7, and Clause (12) asserts there are eight genes $gene\_a$ - $gene\_h$ in the genome of the organism (which must be mapped onto the enzymes in the metabolic network).

B. Experimental Observations

Fig. 3 represents a set of experimental observations that could be revealed by synthetic lethal screening of the test organism and which provide the learning examples for our approach. In particular, the positive and negative literals in the body of Clause (13) are the positive and negative examples, respectively. These examples contain two sorts of atoms:

- $synth\_lethal\_muts(g_1, g_2, m)$: asserts that the genes g_1 and g_2 correspond to 'synthetic lethal mutations' in m (i.e., a mutant lacking both these genes cannot survive in the medium, but a mutant lacking just one can).
- $singl\_lethal\_mut(g, m)$: says gene g corresponds to a 'single lethal mutation' in m (i.e., a mutant lacking this gene cannot survive in the medium).

The first pair of body literals state that $gene\_c$ and $gene\_d$ exhibit a synthetic lethal relationship in med_1 but not in med_2. The final pair of body literals state that genes $gene\_a$ and $gene\_g$ are single lethal mutations in growth media med_1 and med_2, respectively.

C. XHAIL Directives

Fig. 4 contains the XHAIL directives that specify the learning problem. Clause (14) states that the goal of the system is to return a hypothesis that correctly explains the experimental observations above. Clause (15) is a head declaration which states that the returned hypotheses may contain (up to 10) ground atoms of the form $codes(gene\_i, enz\_j)$ where $gene\_i$ is a gene and enz_j is an enzyme.

D. Metabolic Theory

The core component of our model is a theory which links the observable predicates $synth\_lethal\_muts$ and $singl\_lethal\_mut$ to the abducible predicate $codes$. This theory is shown in Fig. 5 and formalises the notion of synthetic lethality in terms of metabolic paths from growth nutrients to essential molecules.

The key predicate, $inviable\_without\_genes$, is defined in Clause (18) and states that a mutant lacking two genes Gi and Gj is inviable (i.e., unable to survive) in a growth medium D if those genes code for two enzymes Ei and Ej and there is some essential molecule M which cannot be made from D without using enzymes Ei and Ej.

As formalised in Clause (19), it is possible to make a metabolite M from D without the help of Ei and Ej if and only if there is a metabolic path from one of the nutrients N in D to the metabolite M which does not use either of the enzymes Ei or Ej. The collection of all such paths are defined inductively in Clauses (20-21).

Clauses (22-24) are convenient abbreviations that make the preceding clauses easier to read. Clauses (25-27) simply define enzymes and metabolites as the objects appearing in the underlying reactions. Similarly, Clause (28) defines growth media as entities that contain nutrients. As stated in Clause (29), an enzyme E is said to be $coded$ if there exists (at least) one corresponding gene G that $codes$ for E. Clauses (30-31) are constraints which state that every enzyme must be coded and that each gene may code at most one enzyme.

Given all of these definitions, the notions of single and synthetic lethal mutations can now be formalised in terms of the predicate $inviable\_without\_genes$ with respect to a medium D. As shown in Clause (17), a gene G corresponds to a single lethal mutation if the organism is inviable without that gene. As shown in Clause (16), genes Gi and Gj correspond to synthetic lethal mutations if the organism is inviable without those genes but neither deletion constitutes, by itself, a single lethal mutation.

E. XHAIL Results

Given the input clauses (1-31) detailed above, the XHAIL system computes two solutions, each containing seven $codes$ atoms (the minimum needed to ensure all seven enzymes are covered). The enzyme-gene codings returned by these solutions are summarised in Fig. 6 below.[1]

enzyme	solution 1	solution 2
enz_a	$gene\_a$	$gene\_a$
enz_b	$gene\_b$	$gene\_b$
enz_c	$gene\_e$	$gene\_c$
enz_d	$gene\_d$	$gene\_d$
enz_e	$gene\_c$	$gene\_e$
enz_f	$gene\_f$	$gene\_f$
enz_g	$gene\_g$	$gene\_g$

Fig. 6. XHAIL Results

Both solutions correctly explain all the experimental observations. They differ only in the mapping between genes and enzymes c and e. This is because, as can be seen from Fig. 1, there is no possible way to distinguish enzymes c and e on the basis of media 1 and 2.[2]

But if we knew a-priori that $gene\_c$ codes for enz_c, then we could rule out the former solution by adding the fact

$$(32)\ codes(gene\_c, enz\_c).$$

And if we wanted XHAIL to compute all non-minimal solutions too, we could add the directive

$$(33)\ find(all).$$

In this case XHAIL returns twenty solutions, obtained from various permutations involving the redundant $gene\_h$.

[1]Total execution time is less than one second on a 1.7 GHz Centrino Duo laptop PC running Windows Vista with 1 Gb of RAM.

[2]Clearly, enzymes enz_c and enz_e could only be differentiated by a medium containing the metabolite mol_4 (in which case the organism could survive without enz_c but not without enz_e).

IV. Related Work

The XHAIL system used in our work integrates techniques from the fields of Abductive Logic Programming (ALP) [11] and Inductive Logic Programming (ILP) [12]. In brief, ALP and ILP are methods for hypothetical reasoning that compute hypotheses which generalise or explain a set of examples or goals with respect to a prior background theory. The two approaches differ primarily in the syntactic restrictions they impose on their outputs and inputs.

- ILP systems are designed to return hypotheses with non-ground clauses, but most are unable to reason correctly with negation. Moreover, many ILP systems can only infer one clause in response to each example and they typically assume the predicate in the head of the hypothesis is the same as the example — a restriction called Observation Predicate Learning (OPL) [13].
- ALP systems are only designed to return hypotheses with ground facts, but most are able to reason correctly with negation. Furthermore, most ALP systems can infer more than one fact in response to each example and they usually do not impose any restrictions on the predicates in the hypotheses and examples.

The reasoning task studied in this paper is a typical ALP problem because: (i) the hypothesis is a set of ground facts; (ii) negative literals are used in the model; (iv) two $codes$ facts must be assumed in order to explain one $synth\_lethal\_muts$ example; and (iii) the predicates in the examples are different from the hypothesis. In this sense our approach is related to previous work on applying ALP to genetic (as opposed to metabolic) regulatory networks [14], [15].

The logical theory used in our work builds upon a model of metabolism [16] used in a Robot Scientist application which 'originates hypotheses to explain observations, devises experiments to test these hypotheses, physically runs the experiments using a laboratory robot ... and then repeats the cycle' [3]. A prototype system was shown to reconstruct part of known biosynthesis pathway by running auxotrophic growth experiments on mutant strains of S. cerevisiae [3].

The brain of the Robot Scientist is an ILP system called Progol5 [13] which uses a contrapositive reasoning method known as Theory Completion by Inverse Entailment (TCIE) to realise a restricted form of abductive inference [17]. While this allows Progol5 to overcome the OPL restriction, it can still only infer one fact to explain each example, and it cannot perform abduction through negation. For these reasons, the Progol5 work is limited to single gene deletions

V. Conclusion and future work

This paper showed how logic programs can be used to infer the function of genes from synthetic lethal mutations. Our approach exploits the facts that (a) logic programs provide an expressive and intuitive language for representing background knowledge, examples and hypotheses and that (b) they benefit from sophisticated reasoning support.

However, the example used in this case study was highly simplified and the model must be refined before it can be usefully applied to real data. Firstly, the model must be extended to deal with reactions that have more than one substrate and/or product. Secondly, the model could be extended to represent more subtle interactions between metabolic pathways. For example, a metabolite in one pathway may inhibit an enzyme in another: either by blocking its catalytic effect or by blocking the expression of its regulating gene(s).

To do this, we will generalise existing logic programming models of metabolic inhibition [9]. Then we will validate our approach on real biological data and compare it with other methods for refining biological networks. Finally, we note that the declarative nature of our logical model means it can easily be extended to interactions between more than two genes.

References

[1] A. Lehninger, *Biochemistry: The Molecular Basis of Cell Structure and Function*, 2nd ed. Worth Publishers, 1979.

[2] C. Boone, H. Bussey, and B. Andrews, "Exploring genetic interactions and networks with yeast," *Nature Reviews Genetics*, vol. 8, no. 6, pp. 437–449, 2007.

[3] R. King, K. Whelan, F. Jones, P. Reiser, C. Bryant, S. Muggleton, D. Kell, and S. Oliver, "Functional Genomic Hypothesis Generation and Experimentation by a Robot Scientist," *Nature*, vol. 427, pp. 247–252, 2004.

[4] O. Ray, "Nonmonotonic Abductive Inductive Learning," in *Proc. of the International Workshop on Abduction and Induction in Artificial Intelligence and Bioinformatics*. To appear, 2008.

[5] F. Jacob and J. Monod, "Genetic regulatory mechanisms in the synthesis of proteins," *Journal of Molecular Biology*, vol. 3, pp. 318–356, 1961.

[6] G. Giaever *et al.*, "Functional profiling of the Saccharomyces cerevisiae genome," *Nature*, vol. 418, pp. 387–391, 2002.

[7] E. Winzeler *et al.*, "Functional Characterization of the S. cerevisiae Genome by Gene Deletion and Parallel Analysis," *Science*, vol. 285, no. 5429, pp. 901 – 906, 1999.

[8] R. Cohen and D. Engelberg, "Commonly used saccharomyces cerevisiae strains (e.g. by4741, w303) are growth sensitive on synthetic complete medium due to poor leucine uptake," *FEMS Microbiology Letters*, vol. 273, no. 2, pp. 239–243, 2007.

[9] A. Tamaddoni-Nezhad, R. Chaleil, A. Kakas, and S. Muggleton, "Application of abductive ilp to learning metabolic network inhibition from temporal data," *Machine Learning*, vol. 64, no. 1-3, pp. 209–230, 2006.

[10] S. Muggleton, "Inverse entailment and Progol," *New Generation Computing*, vol. 13, pp. 245–286, 1995.

[11] A. Kakas, R. Kowalski, and F. Toni, "Abductive Logic Programming," *Journal of Logic and Computation*, vol. 2, no. 6, pp. 719–770, 1992.

[12] S. Muggleton and L. De Raedt, "Inductive Logic Programming: Theory and Methods," *Journal of Logic Programming*, vol. 19,20, pp. 629–679, 1994.

[13] S. Muggleton and C. Bryant, "Theory Completion Using Inverse Entailment," in *Proc. of the 10th International Conf. on Inductive Logic Programming*, ser. Lecture Notes in Computer Science. Springer Verlag, 2000, vol. 1866, pp. 130–146.

[14] I. Papatheodorou, A. Kakas, and M. Sergot, "Inference of Gene Relations from Microarray Data by Abduction," in *Proc. of the 8th International Conf. on Logic Programming and Nonmonotonic Reasoning*, ser. Lecture Notes in Computer Science, vol. 3662. Springer, 2005, pp. 389–393.

[15] B. Zupan, I. Bratko, J. Demsar, J. Beck, A. Kuspa, and G. Shaulsky, "Abductive inference of genetic networks," in *Proc. of the 8th European Conf. on Artificial Intelligence in Medicine*, ser. Lecture Notes in Artificial Intelligence, no. 2101. Springer, 2001, pp. 304–313.

[16] P. Reiser, R. King, D. Kell, S. Muggleton, C. Bryant, and S. Oliver, "Developing a logical model of Yeast metabolism," *Electronic Transactions on Artificial Intelligence*, vol. 5, no. B, pp. 223–244, 2001. [Online]. Available: http://www.ida.liu.se/ext/etaij/

[17] O. Ray, "Automated Abduction in Scientific Discovery," in *Model-Based Reasoning in Science and Medicine*, ser. Studies in Computational Intelligence, vol. 64. Springer, 2007, pp. 103–116.