Automating development of metabolic network models

Robert Rozanski[1⋆], Stefano Bragaglia[2], Oliver Ray[2], Ross King[1]

[1] School of Computer Science, University of Manchester, M13 9PL, UK
[2] Department of Computer Science, University of Bristol, BS8 1TH, UK

Abstract. Although substantial progress has been made in the automation of many areas of systems biology, from data processing and model building to experimentation, comparatively little work has been done on more encompassing systems that combine all of these aspects. This paper presents an active learning system called Huginn that integrates experiment design and model revision in order to automate scientific reasoning about Metabolic Network Models. We validate our approach in a simulated environment using test cases derived from a state-of-the-art model of yeast metabolism. We show that Huginn can not only improve metabolic models but that it is able to solve a wider range of biochemical problems than previous methods and use a wider range of experiment types. Also, we show how design of extended crucial experiments can be automated using Abductive Logic Programming for the first time.

1 Introduction

Biological systems are extremely complicated. Even the model cellular systems of *Escherichia coli* and *Saccharomyces cerevisiae* consist of thousands of genes, proteins, small molecules, *etc.*, all interacting in complicated spatial-temporal ways. In addition, as biological systems have evolved through Darwinian evolution, Ockham's razor is not as effective as it is in the physical sciences.

Currently, although computational tools are used to build systems biology models, their curation is mostly done manually by humans, who identify conflicting results, suspicious or low-confidence elements of models, ask specific questions to test the models and run manual experiments. However, humans can only investigate small parts or aspects of models, because of their typical size and complexity. This bottleneck could be overcome by automating model development,*i.e.* the process of asking specific questions, running tailored experiments to answer them, and revising models if needed.

⋆ corresponding author: rozanskr@cs.man.ac.uk
Huginn is an open-source software, available at:
github.com/robaki/huginnCMSB2015
All figures and tables from this paper are in public domain; files can be downloaded from: github.com/robaki/huginnCMSB2015

1.1 Adam, a robot scientist

King *et al.* [10] created an automated system investigating a problem of orphan enzymes in metabolic models of yeast. The system, called Adam, was able to propose initial hypothetical models and then design two-factor growth experiments to test them. Experiments were run using automated laboratory equipment. Gathered data were then analysed to determine which models to refute. Adam, although successful, has multiple limitations. Its methods of proposing hypotheses are specific to the problem of orphan enzymes. Experiment design and hypothesis testing algorithms were limited to only one type of experiment and could not be easily extended. It also lacked general revision capabilities. These limitations make Adam unsuitable candidate for a general-purpose metabolic model development system.

1.2 Huginn

We have developed Huginn[1], to overcome limitations of Adam. In doing this we have drawn from Machamer's, Darden's and Craver's (MDC) theory of discovering mechanisms. We have adopted MDC concept of mechanism to represent Metabolic Network Models (MNM) in a way suitable for automated system. We have also used their characterisation of the final stage of model development process as a guide when designing Huginn. We have used Logic Programming, and Abductive Logic Programming (ALP) (Gringo [9], Clasp [8] and XHAIL [15]) tools to automate model construction and revision, as well as testing consistency of models with experiments. We have also used them to automate experiment design, which to the best of our knowledge had not been done before in any similar system.

2 Metabolic networks as biological mechanisms

A lot of research in biology is concerned with development of models of mechanisms (*e.g.* of DNA replication or signal transduction). By representing what is happening in biological systems these models provide a way to predict and explain their behaviour in a way easily understandable to humans. Recently the notion of mechanism in biology attracted attention of philosophers of science who try to specify what mechanisms are and how they are being discovered. [2, 5, 6]

In this study we have adopted the notion of mechanism proposed by MDC [14]. They have characterised mechanisms as collections of entities and activities organised in such ways that they can produce regular changes from setup to termination conditions. For example, a model of cellular respiration would show how cells produce ATP from glucose through a series of chemical reactions and transport processes.

The core qualitative information about metabolism are the chemical reactions and other processes that can occur in an organism, as well as chemical

[1] From the Norse mythology – one of two ravens scouting the world for Odin.

substances involved in them. MNM represents these processes in a form of hypergraph. MNM typically abstract away not only concentration and dynamics of the system, but also some of the conditions considerations, *e.g.* certain enzymes are expressed only under specific conditions. MNM can be understood as MDC-type mechanism description. MNM show how certain chemicals are produced from other chemicals by representing continuous chemical paths from the former to the latter. Initial and termination conditions would here be presence of specific species (*e.g.* metabolites) and genes in specific compartments (*e.g.* cytosol). Activities like chemical reactions, transport, gene expression and complex formation would join these conditions through intermediate steps.

3 Methods

3.1 Discovery of mechanisms

The MDC concept of biological mechanism was developed to better understand discovery of mechanisms. Discovery should not be understood here as an event, but as an extended iterative process of exploration, specification, building, testing and revision. According to MDC [4,6], the process starts with exploring and characterising phenomenon of interest, *i.e.* one that is to be explained by description of mechanism. Then, incomplete and often abstract sketches of mechanisms are formulated, taking into account clues like nature of the phenomenon, its context (*e.g.* evolutionary), its spatial and time characteristics. These sketches show how possibly the phenomenon could be produced. Through specification and initial evaluation sketches are turned into schemata: they still may be to some extend incomplete or abstract, but contain enough information to allow production of fully specified models. Then, through further instantiation (if required) and looking for direct experimental evidence, final descriptions of mechanisms are produced. Transition between each of these stages involves construction, evaluation and anomaly resolution (revision), which is guided by specific strategies.

In this paper we focus on the latter stages of the discovery process, where phenomenon is fully characterised and models of mechanisms are composed entirely of non-abstract elements, *i.e.* they are constructed from specific reactions, proteins, metabolites and not from place-holder elements. We implement a number of strategies proposed by MDC in design of Huginn, specifically:

- continuity and productivity are taken into account in construction, consistency testing and revision
- generation and elimination of rival hypotheses (using crucial experiments)
- looking for direct evidence for hypotheses by *in vivo* and *in vitro* experiments:
  - entity and activity detection
  - characterising entities *in vitro* (enzymes' properties and complex formation)
  - disrupting mechanisms and studying changes (gene deletions and changes in medium composition)

The model development process used in Huginn (see fig. 1) is initialised in a number of steps. First, initial models and experiment results are recorded. Then models are checked for consistency with the results, as well as other criteria, like ability to produce termination conditions (*i.e.* synthesize final compounds) and presence of disconnected activities (*e.g.* reactions which substrates are not present in the model). Models that failed are revised. If the pool of initial models is smaller than user-specified threshold, then additional models are produced to fill that gap and the system is ready to enter proper development cycle.

The first step in the development cycle is to design an experiment to test current working models. Then, the experiment is executed (simulated) and results used to test working models. Refuted models are revised. If there is no way to make the model logically consistent with the results, then one or more of them will be ignored. This ability to ignore results is important for dealing with limitations of the Knowledge Representation method, as well as factors such as experimental noise, and the open world problem. The quality scores of models are then recalculated based on the number of covered and ignored results.

Huginn stops development process if at least one of three conditions is true. The first condition is lack of progress. If there were any new models produced recently or if the best (highest quality score) model has recently changed, then development continues. The second condition is running out of experiments to do. It happens when working models become empirically equivalent. In this case Huginn tries to redesign models at random, but if it fails 10 times, it stops. The last condition is running out of time or exceeding maximum number of cycles: both values are specified by the user.

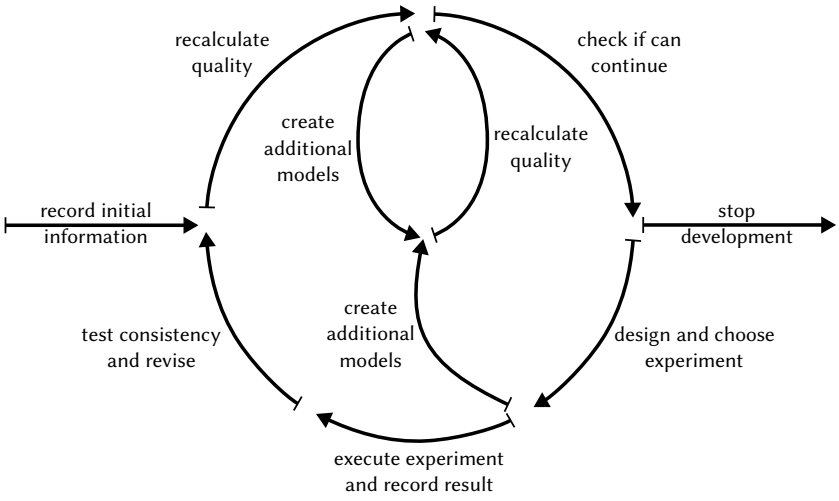

Fig. 1. Model development process

3.2 Abductive logic programming

The development process relies on four core operations: consistency checking, revision, production of additional models and experiment design. We use ALP for these operations. Abductive inference is typically understood as inference to the best explanation. In ALP abduction is defined as constructing a hypothesis H, that together with background knowledge B, entails a set of examples E:

$$B \cup H \models E$$

Unlike deduction, abduction is a defeasible form of inference, *i.e.* given true background knowledge and examples (observations), it may produce false hypotheses. However, it has the advantage of being able to produce novel knowledge. ALP tools have been used previously for completion [3, 11] and revision of metabolic networks [16]. Thanks to optimisation capabilities of existing tools one can generate theories that not only satisfy hard logical constraints, but are also optimal with respect to user-specified criteria.

3.3 Representing models using logic

MNM can be formalised and translated into datalog-style logic programs. Entities are defined by their type, identifier and version. Huginn currently supports four types of entities: metabolite, protein, complex or gene. Versions enable one to represent uncertainty regarding an entity's properties. Two currently supported properties are *catalyses* and *transports*.

Huginn supports five types of activities: chemical reaction, complex formation, expression, transport or growth. Substrate and product predicates are used for all types of activities and specifies not only what entities are required and produced, but also in what compartments. Apart from substrates, chemical reactions and transport may need catalyst or transporter respectively.

Models are defined by specifying which setup conditions and activities they contain.

All these facts describe the elements involved in the MNM. In order to determine which metabolites are synthesizable, simulation rules are added to this description. A group of rules marks as *active* activities which all substrates are either initially present or synthesizable (in appropriate compartment) and which catalyst/transporter requirements are met. Additional rule marks all products of active reactions as synthesizable.

3.4 Experiment types and predictions

Model descriptions need to be supplemented with prediction and consistency rules to support use of empirical information. Predictions describe what outcome models predict w.r.t. description of experiment. Outcome is binary: true or false. In addition, model can be indifferent w.r.t. experiment (it does not predict any outcome). Model is inconsistent with a result of experiment if outcome

of the experiment is different from the predicted one. Prediction rules determine predicted outcomes of experiments. There are seven types of experiments currently used in Huginn and each of them has its separate set of prediction rules:

Entity Detection: detection of metabolites, proteins or complexes.
Entity Localisation: as above, but in a specified compartment.
Activity Detection: used for detecting growth.
Activity Reconstruction: checks if activities can be reconstructed without enzymes or transporters.
Reconstruction Enzymatic Reaction: checks whether given entity can catalyse specific reaction.
Reconstruction Transporter Required: as above, but for transporters.
Two Factor Growth Experiment: used previously to test candidate parent genes of orphan enzymes [10]. It tests whether decreased growth rate after gene deletion can be offset by addition of a particular metabolite.

Some types of experiments can include interventions: addition or substraction of a specific entity from specific compartment. In our study we have restricted interventions to manipulation of the growth medium (addition/substraction of nutrients) and gene deletions. The way the interventions are handled differs depending on the nature of the task (revision, experiment design, *etc.*).

3.5 Automating crucial tasks

As mentioned above, four essential tasks in the model development cycle are: consistency check, revision, construction of additional models and experiment design. All of these tasks were automated using Logic Programming (LP) techniques.

consistency check: This step consist in checking whether models are consistent with all known results as well as additional structural criteria. Specifically, models must synthesize all compounds specified in the termination conditions, they must not contain any activities missing substrates and they cannot contain two versions of the same entity (that situation would be equivalent to having inconsistent beliefs about the entity's properties).

revision: Models are revised by supplementing requirements from consistency check with mode declarations specifying what activities can be added and removed. XHAIL then tries to minimise a number of changes to the model. In cases where more than one optimal solution is found, one is chosen at random to keep the population of working models at a constant size.

For example, metabolite met_8 was detected in cells with deleted gene $g26$. This outcome is in conflict with predictions of model (a) (fig. 2). In that model met_8 can be synthesized from input metabolites met_7 and met_11 (marked

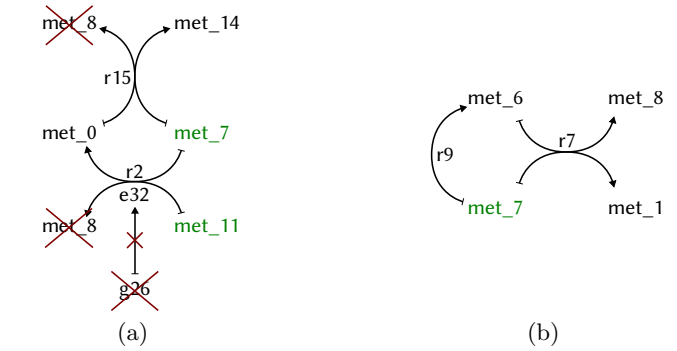

Fig. 2. Revision example: (a) deletion of $g26$ disrupts reactions $r2$ (lack of enzyme) and $r15$ (lack of substrate: met_0) and thus prevents the model from producing met_8, contrary to experimental results. Consistency with the results can be restored by adding two additional reactions (b) which can produce met_8 independently from $g26$.

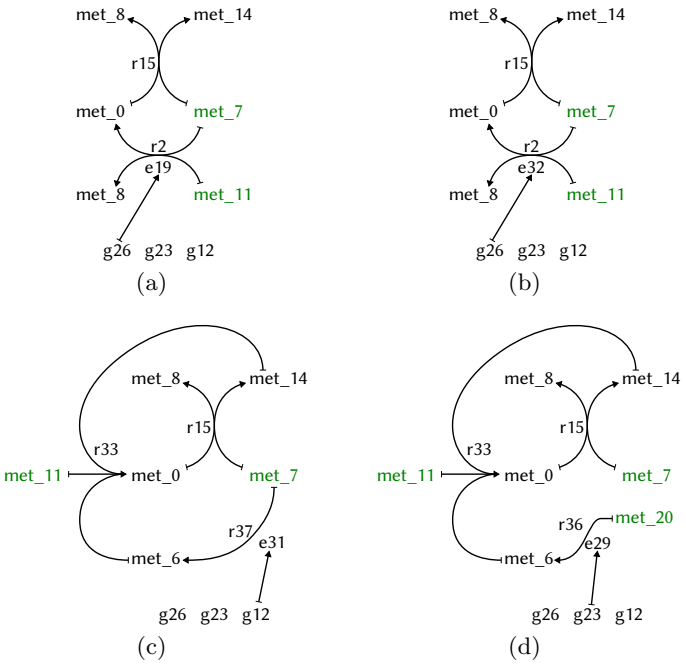

Fig. 3. Experiment design example: models (a) and (b) rely on gene $g26$ to produce met_0 and met_14, while models (c) and (d) rely on genes $g12$ and $g23$ respectively. Thus experiment consisting in deleting $g26$ and detecting either met_0 or met_14 will split these models into two groups: one predicting that the metabolite will be synthesised despite deletion, the other that it will not be.

green) in reaction $r2$, which requires enzyme coded by $g26$. Alternatively, it can be synthesised in $r15$, but that requires some source of substrate met_0. Since the only source of met_0 is $r2$, deletion of $g26$ disrupts both reactions and met_8 is not produced. Consistency with the experimental result can be restored by adding reaction(s) that can synthesise met_8 independently from $g26$, *e.g.* reactions $r9$ and $r7$ (fig. 2(b)).

construction of additional models: Additional models are constructed using almost the same approach as revision, but adding a requirement that resulting models must be different (contain different set of activities) from any of the working models.

experiment design: The idea behind our approach to experiment design is to design such experiment that it will split the working models into two groups of equal size: one predicting that outcome of experiment is true, the other that it is false. This can be understood as an extension of the concept of crucial experiment.

For example, lets consider four models from fig. 3. The input metabolites are met_7, met_11 and met_20 (marked green, only shown where relevant), and the output metabolite is met_14. All models synthesize met_14 in $r15$, but differ in ways they produce required substrate for this reaction: met_0. Models (a) and (b) rely on $r2$ and gene $g26$, while models (c) and (d) use $r37$ (needs gene $g12$) and $r36$ (needs $g23$) respectively. Therefore, if $g26$ is deleted models (a) and (b) will predict that met_14 is not produced, while (c) and (d) that it is produced. One of plausible experiments for this group of models is then a *detection entity* experiment, detecting met_14 and involving one gene deletion (of $g26$).

Since some models may be considered to be better and therefore more probably correct in a subjective sense, we split not raw numbers of models, but rather their total quality score. Since designing experiment that will split scores into equal groups is not always possible, this task was implemented as optimisation problem. The system tries to minimise total penalty, which is calculated as follows:

$$P = |0.5 * \sum_{m} q(m) - \sum_{m \in T} q(m)| + |0.5 * \sum_{m} q(m) - \sum_{m \in F} q(m)| + \sum_{m \in I} q(m)$$

where m is model, $q(m)$ is model's quality, T, F and I are sets of models predicting that outcome is true, false or indifferent respectively. Due to complicated nature of this task it was implemented using Gringo/Clasp directly, not through XHAIL.

4 Results and Conclusions

The goal of our study is to check whether the proposed system can be used in model development. To answer this question we supplied Huginn with initial

models containing errors and run the development process to see whether the models would be improved. At this initial stage of evaluation using real biochemical experiments is not necessary and would not be cost effective. Instead we have run simulations using models of reference, which are fragments of the yeast consensus metabolic model 7.11 [1]. Knowledge bases containing activities and entities for model development were created by mixing elements from a given model of reference with additional, erroneous elements which role is to make the development process harder. Initial models were created by randomly selecting a set of activities from these knowledge bases. Improvement of models

Table 1. Model improvement during simulations: decrease in average error, expressed as a fraction of initial error, for system configuration that uses all types of experiments (columns 3 and 4) and configuration using only two-factor growth experiments (columns 5 and 6). Three simulations were run per each test-configuration combination. Column 2 shows number of all activities (metabolic reaction, transport, *etc.*) involved in each test-case.

		all experiments		two-factor only	
test-case	size	mean	std. dev.	mean	std. dev.
1	25	0.52	0.02	0.26	0.06
2	31	0.42	0.06	0.27	0.08
3	35	0.22	0.05	0.14	0.05
4	39	0.46	0.19	0.40	0.15
5	42	-0.01	0.08	0.02	0.00
6	43	0.30	0.01	0.22	0.01
7	46	0.28	0.09	0.20	0.07
8	60	0.57	0.08	0.40	0.03
9	71	0.45	0.07	0.28	0.09
10	86	0.47	0.13	0.37	0.09
11	86	0.42	0.02	0.32	0.03

consists in removing and adding activities so that working models resemble the model of reference. To quantify the difference between a model and the model of reference we use a symmetric difference between the sets of activities involved in the models. To test if performance differs significantly between different configurations of the system we have used pair-wise comparison of improvement and then a binomial test.

Results of our simulations show that Huginn can successfully improve initial models, with an average reduction in initial error of 37% (table 1, columns 3 and 4). For the smaller test-cases (1-7) working models tended to quickly become empirically equivalent and attempts to recover from it through generation of

randomised models would fail. For the bigger test-cases(8-11), Huginn tended to continue development until a time-out. That last unsuccessful attempt to construct new models was associated with decrease in average model quality (p=0.024). However, since in many cases constructing random models allowed Huginn to recover and continue development process, including this ability was beneficial (p=0.003).

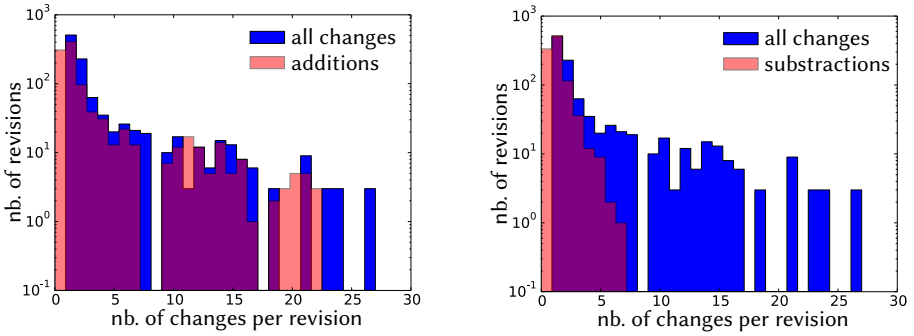

Fig. 4. Size of model revisions: each model revision may consist of multiple changes (additions or substractions of activities). The histograms compare distributions of additions (a) and substractions (b) with all changes. Note the log scale on y axis.

One of the crucial differences between Adam and Huginn is ability to use more types of experiments. To test whether this change is beneficial we have run additional simulations while limiting available experiment types to only *two-factor growth* experiments (table 1, columns 5 and 6). The results show that using more experiments is associated with bigger improvements (p=0.014). The main experiment types used by Huginn were *two-factor growth* (52% of experiments) and *entity detection* (45%). The latter often involved multiple interventions: gene deletions and medium manipulations (41%). In the most extreme case experiment would use 5 gene deletions on top of manipulating medium composition. While execution of such experiment in practice would be challenging at best, it shows that ALP techniques used in Huginn can cope with complex experiment design problems. The rest of experiment types used by Huginn were *reconstruction of enzymatic reactions* and *entity localisation*.

Another crucial difference between Adam and Huginn are their revision abilities. Adam can only add individual missing expression activities. Thanks to XHAIL, Huginn handles a wider range of activities, can also remove them, and can introduce multiple changes in one revision. Therefore, it should be able to make more substantial changes to MNM structures. Our simulations show that it is indeed the case: 51% of revisions involved more than one change (addition/substraction), while the biggest involved as much as 27 changes (fig. 4). Many revisions combined addition and substraction of activities (38%). Majority

of revisions (56%) involved changing elements other than expression activities. These results show that Huginn takes advantage of its enhanced revision abilities, introduces bigger changes to the models and is therefore capable of solving wider range of biochemical problems than Adam – not only the problem of orphan enzymes, but also other structural problems in the metabolic networks.

To sum up, we can conclude that Huginn improves on Adam, in a qualitative sense, by using more types of experiments and a more versatile revision method, and that these improvements translate into an increased ability to correct models. More extensive *in silico* tests are still needed to test Huginn's performance in different configurations and under different circumstances. For example, we did not test Huginn's ability to handle inconsistencies in results (*e.g.* introduced by experimental errors). We can also conclude that the presented experiment design solution can not only design useful experiments, but also handle complicated tasks that require multiple interventions.

5 Related work

King *et al.* [11] investigated performance of experimental strategies in the context of biochemical model completion. Our experiment design solutions follows the same principle as their ASE strategy for choosing experiments.

Substantial advancements have been done in the field of equation discovery. Džeroski and Todorovski [7] described QMN and LAGRANGE – systems for discovering quantitative and qualitative laws governing dynamical systems. Schmidt and Lipson [17] developed a system for discovering non-trivial conservation laws from experimental data. Lovell [13] described an artificial experimenter, whose role was to discover equations describing behaviour of biochemical entities like enzymes or co-enzymes. Todorovski *et al.* [18] developed HIPM, a system for developing complex hierarchical models of dynamical systems using induction, while taking advantage of expert knowledge. Compared to these studies we focus on qualitative aspects of scientific discovery, which can provide necessary insight into functioning of biological systems in terms of mechanistic explanations. However methods for developing quantitative models are likely to be useful in further steps of building biological models.

Langley [12] summarised lessons learned from their experience with developing computational tools for scientific discovery. They advise to use scientists' representations and their knowledge; tools should not just summarise, but provide explanations. Our approach follows these lessons. Representation of metabolism used by Huginn is taken from biochemistry, ensuring that it is easily understandable by biologists. Huginn records all produced models and results so checking why particular models were produced is possible.

Acknowledgment

This work is supported by an EPSRC-EU Doctoral Training Award and the Faculty Engineering and Physical Sciences of the University of Manchester.

References

[1] H. W. Aung, S. A. Henry, and L. P. Walker. Revising the representation of fatty acid, glycerolipid, and glycerophospholipid metabolism in the consensus model of yeast metabolism. *Industrial Biotechnology*, 9(4):215–228, 2013.

[2] W. Bechtel and R. C. Richardson. *Discovering complexity: Decomposition and localization as strategies in scientific research*. MIT Press, 2010.

[3] G. Collet, D. Eveillard, M. Gebser, S. Prigent, T. Schaub, A. Siegel, and S. Thiele. Extending the metabolic network of ectocarpus siliculosus using answer set programming. In *Logic Programming and Nonmonotonic Reasoning*, pages 245–256. Springer, 2013.

[4] C. Craver and L. Darden. Discovering mechanisms in neurobiology. *Theory and method in the neurosciences*, pages 112–137, 2001.

[5] C. F. Craver and L. Darden. *In search of mechanisms: Discoveries across the life sciences*. University of Chicago Press, 2013.

[6] L. Darden. *Reasoning in biological discoveries*. Cambridge University Press, 2006.

[7] S. Džeroski and L. Todorovski. Discovering dynamics: from inductive logic programming to machine discovery. *Journal of Intelligent Information Systems*, 4(1):89–108, 1995.

[8] M. Gebser, B. Kaufmann, A. Neumann, and T. Schaub. clasp: A conflict-driven answer set solver. In *Logic Programming and Nonmonotonic Reasoning*, pages 260–265. Springer, 2007.

[9] M. Gebser, T. Schaub, and S. Thiele. Gringo: A new grounder for answer set programming. In *Logic Programming and Nonmonotonic Reasoning*, pages 266–271. Springer, 2007.

[10] R. King, J. Rowland, S. Oliver, M. Young, W. Aubrey, E. Byrne, M. Liakata, M. Markham, P. Pir, L. Soldatova, et al. The automation of science. *Science*, 324(5923):85–89, 2009.

[11] R. D. King, K. E. Whelan, F. M. Jones, P. G. Reiser, C. H. Bryant, S. H. Muggleton, D. B. Kell, and S. G. Oliver. Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature*, 427(6971):247–252, 2004.

[12] P. Langley. Lessons for the computational discovery of scientific knowledge. 2002.

[13] C. J. Lovell. *An artificial experimenter for automated response characterisation*. PhD thesis, University of Southampton, 2011.

[14] P. Machamer, L. Darden, and C. F. Craver. Thinking about mechanisms. *Philosophy of science*, pages 1–25, 2000.

[15] O. Ray. Nonmonotonic abductive inductive learning. *Journal of Applied Logic*, 7(3):329–340, 2009.

[16] O. Ray, K. Whelan, and R. King. Automatic revision of metabolic networks through logical analysis of experimental data. In *Inductive Logic Programming*, pages 194–201. Springer, 2010.

[17] M. Schmidt and H. Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.

[18] L. Todorovski, W. Bridewell, O. Shiran, and P. Langley. Inducing hierarchical process models in dynamic domains. In *Proceedings of The National Conference on Artificial Intelligence*, volume 20, page 892. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.