Linköping Electronic Articles in
Computer and Information Science
Vol. 7(2002): nr 12

Machine Learning Metabolic Pathway descriptions using a Probabilistic Relational Representation

N. Angelopoulos

S.H. Muggleton

Linköping University Electronic Press
Linköping, Sweden

http://www.ep.liu.se/ea/cis/2002/012/

Published on November, 2002 by
Linköping University Electronic Press
581 83 Linköping, Sweden

Linköping Electronic Articles in
Computer and Information Science
ISSN 1401-9841
Series editor: Erik Sandewall

Typeset by the authors using LaTeX
Formatted using étendu *style*

Recommended citation:
<Authors>. <Title>. Linköping Electronic Articles in Computer and Information Science, Vol. 7(2002): nr 12. http://www.ep.liu.se/ea/cis/2002/012/. *November, 2002.*

This URL will also contain a link to the authors' home pages.

Affiliations

N. Angelopoulos S. Muggleton
Department of Computing, Imperial College, London, UK.

Abstract

A variety of approaches aimed at machine learning probabilistic relational representations have been explored in recent years. These include learning algorithms for Probabilistic Relational Models, Bayesian Logic Programs and Stochastic Logic Programs. In this paper we investigate the question of whether there exist important applications which require the machine learning of such representations. In particular, applications of this kind must necessitate a) machine learning, b) probabilistic representation and c) relations. We argue in this paper that the automated extension of partial metabolic pathway descriptions fits all these criteria. As usual with such representations the learning task can be divided into sub-tasks involving model building and parameter estimation. In order to gain understanding of the sample size requirements for estimating the parameters of medium-sized metabolic networks we conducted experiments in which RMS error is estimated with increasing sample size. We separately investigated the effects of multiple pathway contributions to the concentrations of metabolites within the networks. For non-branching pathways the results indicate that parameter estimation is feasible from limited sources of data. However, the introduction of branches increases sample requirements considerably.

1 Introduction

A variety of approaches aimed at machine learning probabilistic relational representations have been explored in recent years. These include learning algorithms for Probabilistic Relational Models [7], Bayesian Logic Programs [6] and Stochastic Logic Programs [8]. In this paper we investigate the question of whether there exist important real-world applications which require the machine learning of such representations. In particular, applications of this kind must necessitate a) machine learning, b) probabilistic representation and c) relations. We argue in this paper that the automated extension of partial metabolic pathway descriptions fits all these criteria. As usual with such representations the learning task can be divided into sub-tasks involving model building and parameter estimation. In order to gain understanding of the sample size requirements for estimating the parameters of medium-sized metabolic networks we conducted experiments in which RMS error is estimated with increasing sample size. We separately investigated the effects of multiple pathway contributions to the concentrations of metabolites within the networks. For non-branching pathways the results indicate that parameter estimation is feasible from limited sources of data. However, the introduction of branches increases sample requirements considerably.

This paper is structured as follows. Section 2 covers the motivation for the paper in terms of the question to be investigated. It is argued here that accurate modelling of metabolic pathways for the purpose of functional genomics requires machine learning, a probabilistic representation and the use of relations. Section 3 introduces the representation of Stochastic Logic Programs (SLPs). An existing algorithm, FAM, for parametric estimation of the probability labels of SLPs is described in Section 4. Sections 5 and 6 describe experiments which test the feasibility of applying FAM to estimate probabilities for non-branching and branching metabolic networks respectively. Lastly a summary of the results is provided in Section 7 together with suggestions for further work.

2 Motivation

The overall objective of this paper is to critically investigate *Probabilistic Logic Learning* methods by answering the following question:

Question. Do there exist important applications which require the machine learning of Probabilistic Relation representations?

In the following section we will argue that the modelling of metabolic pathways represents such an application. In particular we argue the requirement for a) machine learning, b) relational representation of learned models and background knowledge and c) a probabilistic representation.

2.1 Metabolic Pathways

Metabolic networks represent the detailed function of biochemical reactions within organisms. Publicly available metabolic network databases such as KEGG[1] allow bioinformaticians to cross-reference knowledge concerning the location and sequence of known genes, protein products and ligands with

[1] (http://www.genome.ad.jp/kegg)

reported biochemical reaction pathways involved in metabolism. KEGG contains overlays for the metabolism of multiple organisms.

Figure 1 shows the aromatic amino acid pathway of yeast. In the study described in [2] this pathway description was extracted from KEGG and encoded in Prolog [12]. The aim of the study was to use the Inductive Logic Programming (ILP) system ASE-Progol to augment this pathway description on the basis of experimentally-derived data.

2.1.1 Requirement for machine learning

KEGG has become an invaluable tool for bioinformaticians in industry and academia world-wide. However, it contains information which is both incomplete and sometimes also incorrect. For instance, in many cases there are pathways with missing enzymes. Also it is well known that metabolic pathways often have many pathways associated with generation of a particular metabolite. Often researchers document only the primary pathway. Symbolic machine learning is arguably the only technology which has the potential to use observational data to revise and augment known pathway models. In the previously mentioned study [2] it was demonstrated that Inductive Logic Programming (ILP) used in this context could reduce expected laboratory costs involved in determining novel gene functions by two orders of magnitude.

2.1.2 Requirement for relational representations

Identification of the detailed chemistry of an unknown enzyme function is an under-constrained problem without the use of background knowledge. The required background knowledge needs to represent 1) the details of the substrates and products of individual reactions, 2) the existence of required co-factors and inhibitors associated with the enzyme reaction. This information is available from databases such as Ligand [13] in the form of relational tables. Additionally, hierarchical descriptions of general classes of enzyme function are described in the GO database [3]. The GO descriptions could potentially be used to help the automatic proposal of novel enzyme functions. However, the descriptions are only available in the form of natural language. The data is available from Ligand to learn rules which represent the GO classes. However, due to the relational aspect of background predicates such as co-factor-of and inhibitor-of, these class descriptions would require a first-order representation.

2.1.3 Requirement for relational representations

One approach to representing metabolic pathways is as a logic program with one predicate for every metabolite. The graph representing the calling diagram of the predicates in the logic program can then be equated with a metabolic network. This was the approach taken in [2]. One problem with this approach is that it fails to account for relative rates of reactions. Figure 2 illustrates this problem. In the figure the metabolite C00493 can be synthesised using enzymes A and B. Suppose that in a case like this Pathway A had a reaction rate which is 100 times that of B, ie $p_A = 100p_B$. It would be difficult to distinguish between results based on starvation versus plentiful supply of substrates needed for Pathway B. However, starvation of substrates for Pathway A would almost completely block the generation of C00493. This would disagree with the logic program-based representation which would simply state that C00493 could either be generated by

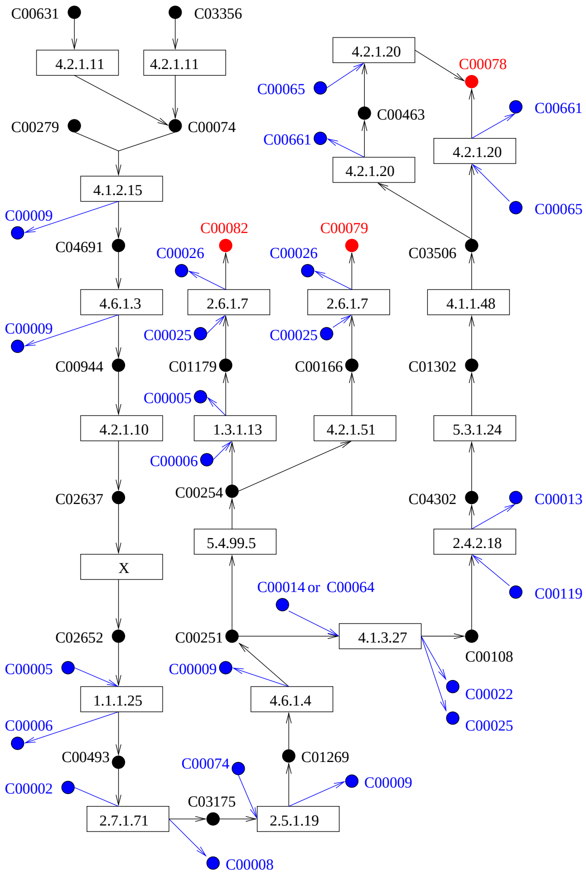


Figure 1: The aromatic amino acid pathway of yeast [2]. The circles represent metabolites, referred to by their KEGG [13] accession numbers. The pathway pertains to the biosynthesis of the aromatic amino acids phenylalanine, tyrosine and tryptophan which are shown in red. The main reaction paths are shown in black. Molecules which are involved in the reactions but which do not lie on the main pathway are shown in blue. Each rectangle, together with those metabolites which are linked to it, represents a chemical reaction. Reactants are shown entering on one side of each rectangle and products leaving on the other. Each rectangle is labelled by the EC-number [11] of the enzyme which catalyses the reaction. For clarity arrows are included pointing to intermediate and end products.

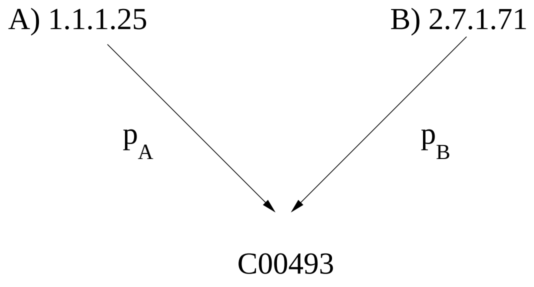


Figure 2: Alternative synthesis paths for metabolite C00493 involving enzymes 1.1.1.25 (A) and 2.7.1.71 (B). p_A and p_B represent the proportions of C00493 which are due to reactions A and B respectively.

Pathway A or by Pathway B. This example demonstrates that representations are needed which allow incorporation of rate information. The most obvious way to do so is by way of probabilities which capture these rates as proportions. The Brenda database provides rate constants for some reactions which can be used in conjunction with the Michaelis-Menten equation to derive these probabilities. Unfortunately this information is very incomplete. However, given metabonomic data [10][2] these probabilities could be derived directly using algorithms such as FAM [5] for estimating labels on SLPs. The feasibility of this approach is investigated in the experiments described in Sections 5 and 6. First we introduce the representation of SLPs and the FAM algorithm in the following sections.

3 Stochastic Logic Programs

3.1 Syntax of SLPs

A pure SLP S is a set of labelled clauses $p{:}C$ where p is a probability (ie. a number in the range $[0, 1]$) and C is a first-order range-restricted definite clause[3]. The subset S_p of clauses in S with predicate symbol p in the head is called the definition of p. For each definition S_p the sum of probability labels π_p must be at most 1. S is said to be normalised if $\pi_p = 1$ for each p and unnormalised otherwise. $P(S)$ represents the definite program consisting of all the clauses in S, with labels removed.

Example 1 Unbiased coin. *The following SLP is normalised and represents a coin which comes up either heads or tails with probability 0.5.*

$$S_1 = \left\{ \begin{array}{l} 0.5 : coin(head) \leftarrow \\ 0.5 : coin(tail) \leftarrow \end{array} \right\}$$

S_1 is a simple example of a sampling distribution

3.2 Proof for SLPs

A Stochastic SLD (SSLD) derivation is a sequence $D_{S,G} = \langle 1{:}G, p_1{:}C_1, .., p_n{:}C_n \rangle$ in which G is a goal, each $p_i{:}C_i \in S$ and $D_{P(S),G} = \langle G, C_1, .., C_n \rangle$ is an SLD

[2]Time varying measurements of metabolic concentrations in living organisms
[3]Cussens [4] considers a less restricted definition of SLPs.

derivation from $P(S)$. A SSLD derivation $D_{S,G}$ is a refutation in the case in which $D_{P(S),G}$ is a refutation from $P(S)$. SSLD refutation represents the repeated application of the SSLD inference rule. This takes a goal $p{:}G$ and a labelled clause $q{:}C$ and produces the labelled goal $pq{:}R$, where R is the SLD resolvent of G and C. The answer probability of $D_{S,G}$ is $Q(D_{S,G}) = \prod_{i=1}^{n} p_i$. The incomplete probability of any ground atom a with respect to S is $Q(a|S) = \sum_{D_{S,(\leftarrow a)}} Q(D_{S,(\leftarrow a)})$. We can state this as $S \vdash_{SSLD} Q(a|S) \leq Pr(a|S) \leq 1$, where $Pr(a|S)$ represents the conditional probability of a given S.

4 Parameter estimation algorithm

For parameter estimation over Stochastic Logic Programs we use the Failure Adjusted Maximisation (FAM) algorithm [5]. This is an EM algorithm which takes into account failure branches in the (stochastic) proof tree. Our implementation of the algorithm is written in Prolog and follows [5]. The only notable difference is that we treat all refutations, ambiguous or otherwise, uniformly.

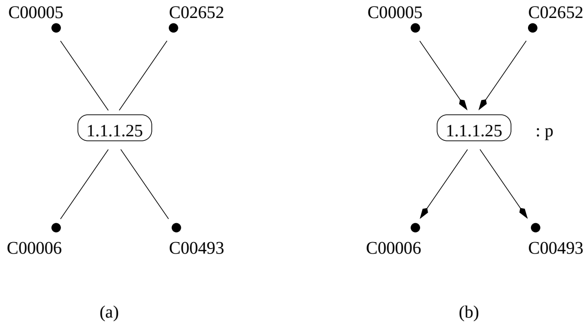
The expected value FAM maximises, is the contribution of a particular clause to deriving the observed/training data. The two main values in calculating this at each iteration are, the contribution of the clause to deriving an individual datum weighted by the frequency of the datum, and the contribution, adjustment, to the failed derivations, which is weighted by the overall probability of failure. The expected values for the clauses of a single stochastic predicate are normalised and become the new stochastic labels. The algorithm terminates when the loglikelihood has converged.

FAM only operates correctly over *pure*, *normalised* SLPs. A normalised SLP means that a single optimum exists. Our programs are normalised since we are interested in the probabilistic dimension of SLP, and thus the labels of each stochastic predicate add-up to 1.

In pure SLPs each derivation is uniquely identified by the sequence of the clauses visited, since choice between matching clauses is stochastic rather than in the SLD style. Instead of pure SLPs we propose a less restrictive class of programs. *Singular* SLPs are mixed/impure programs for which all derivations of some predefined goal map to distinct stochastic paths. A stochastic path is the sequence of all stochastic clauses used in a derivation. In effect this limits the amount of non-determinacy, i.e. backtracking at execution time, permissible. In our programs this was achieved by using a top level deterministic predicate, which shielded non-determinacy from the stochastic part of the computation.

5 Experiment 1 - non-branching

A part of the pathway introduced in Section 2.1, containing a single enzyme, is detailed in Fig 3(a). Metabolites $C00005$ with $C02652$ and $C00006$ with $C00493$ can be, irrespectively, the inputs and outputs of the reaction facilitated by enzyme 1.1.1.2. In principle, enzymes enable reactions in both directions. The probabilistic version, Fig 3(b), requires that reactions occur in one direction. This is because the semantics of the attached probability are "Given the inputs are present, the reaction will happen with probability p". Note that the probability is attached to the reaction, Fig 3(c), not to the enzyme. Thus, when we need both directions we have to explicitly attach each direction to the relevant enzyme. In Machine Learning one is




```
enzyme( '1.1.1.25', rea_1_1_1_25, [c00005,c02652], [c00006,c00493] ).
0.80 :: rea_1_1_1_25( yes, yes, yes, yes ).
0.20 :: rea_1_1_1_25( yes, yes, no, no ).
```

(c)

Figure 3: At top, single node of two pathways. Top left, the non-probabilistic case. Top right, reaction has direction and probability. Bottom, the SLP code for the probabilistic case.

not very often interested in both directions of enzyme reactions, because a set of input metabolites is usually given and the focus is in whether certain outputs can be produced.

In general attaching a probability to a reaction can account for a multitude of reasons for which a reaction might not always occur. For example, rates of reactions in relation to availability of metabolites, competing reactions, secondary paths and variations of physical chemistry assumptions.

5.1 Materials

Our program contains 21 stochastic clauses each corresponding to a single enzyme (full code can be found in [1]). This is effectively a node-edge representation of a chain network/graph. Each edge has an attached probability, p_j, which signifies that the reaction will happen with probability p_j while it will not occur with probability $1 - p_j$. Traversing this chain is done by a non stochastic part which is deterministic at the top-level. We assume, (a) reactions deplete their inputs, which are assumed to be in unit quantities, and (b) each reaction is considered at most once, provided its input metabolites are present.

The core parameter estimation (PE) scenario is as follows. An SLP with its N true parameters $p = \langle p_1, p_2, \ldots, p_N \rangle$ is used to sample S samples from a single top-goal. The parameters are then replaced by uniformly distributed ones. Using FAM we obtain $\bar{p} = \langle \bar{p}_1, \bar{p}_2, \ldots, \bar{p}_N \rangle$. The top goal for querying both the discussed programs is $?can\_produce(Metabolites, Products)$. . In this respect the data carry detailed information about which reactions failed to fire and where the chain(s) of reactions stopped.

5.2 Method

There are two measures of interest, (a) how good an estimate of $p, \bar{p}$ is, and (b) how long did it take us to find $\bar{p}$. All calls to FAM were made with a termination condition of $1.0e-09$ for the log likelihood. In all cases the corresponding $\bar{p}$ was found within a single iteration.

x	t	I_x^t	$S_x^{l^t}$	$S_x^{u^t}$	$S_x^{i^t}$
a	1	10	100	1000	100
	2	20	110	1010	100
b	1	5	100	3300	400
	2	10	110	3310	400

Table 1: Dimensions of all experiments.

Each experiment E_x is identified by a single lower case letter, x. There are two sets of numbers ($t \in \{1, 2\}$) associated with each E_x. Number of iterations (I_x^t) a samples lower limit ($S_x^{l^t}$), a samples upper limit ($S_x^{u^t}$) and a samples step interval ($S_x^{i^t}$). See Table 5.2 for the dimensions of experiments. The intuition is that $S_a^{i^1} = S_a^{i^2}$ while $I_a^1 \neq I_a^2$ so to compare the effect of multiple iterations on the mean errors, and $S_a^{l^1}$ very near but not equal to $S_a^{l^2}$ so both sets can be viewed on a single graph without the error bars being cluttered. Thus, for experiment x and index t, FAM is run I_x^t times for each S in S^*

$$S^* = \{S \cdot 0 \leq k \wedge S = S_x^{l^t} + kS_x^{i^t} \wedge S \leq S_x^{u^t}\}$$

5.2.1 Accuracy

To estimate fitness of $\bar{p}$ to p we use *root mean square error* (Rms, $R_x^{t_i}$) for the means of squares for each $\bar{p}_k$ to p_k. Let $\bar{p}_k$ be a shorthand for $\bar{p}_{\langle x,t,i,k \rangle}$ be the k learned parameter for experiment x set t and iteration i. Then

$$R_x^{t_i} = \sqrt{\left(\frac{\sum_{j=1}^{N} (p_j - \bar{p}_{(x,t,i,j)})^2}{N} \right)}$$

For the complete set of iterations for t, the mean and the sample standard deviation are :

$$\bar{y}_x^t = \frac{\sum_{j=1}^{I_x^t} R_x^{t_j}}{I_x^t}$$

$$s_x^t = \sqrt{\frac{(I_x^t \sum_{j=1}^{I_x^t} (R_x^{t_j})^2) - (\sum_{j=1}^{I_x^t} R_x^{t_j})^2}{I_x^t (I_x^t - 1)}}$$

5.2.2 Runtimes

Runtimes are measured in milliseconds and the raw values are used. $T_x^{t_i}$ is the runtime of experiment x and set of dimensions t at ith repetition. Thus,

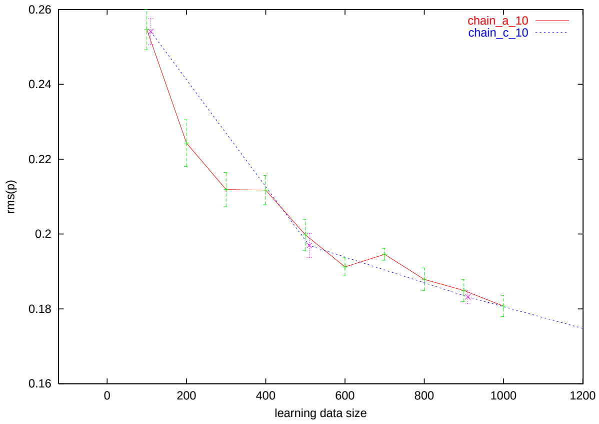
$$\bar{y}_x^t = \frac{\sum_{j=1}^{I_x^t} T_x^{t_j}}{I_x^t}$$

$$s_x^t = \sqrt{\frac{(I_x^t \sum_{j=1}^{I_x^t} (T_x^{t_j})^2) - (\sum_{j=1}^{I_x^t} T_x^{t_j})^2}{I_x^t (I_x^t - 1)}}$$

All graphs presented here are drawn with y-axis errorbars. These, for particular x and t are calculated by

$$e_x^t = \frac{s_x^t}{\sqrt{I_x^t}}$$

5.3 Results



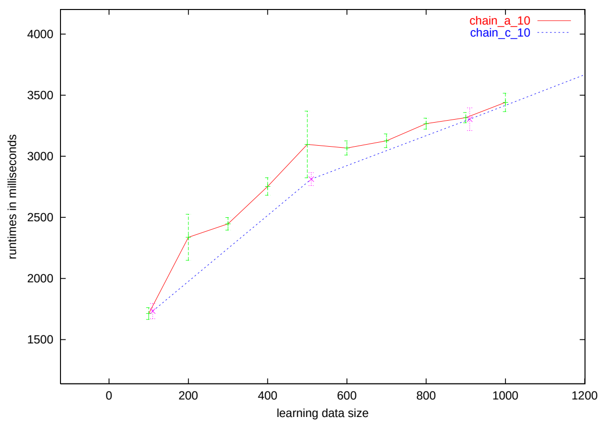


Figure 4: Left, (a) RMS for E_a and E_b at 10 iterations. Right, (b) runtimes for the same experiments.

In Fig. 4 we show the graphs for (a) the RMS of the parameters error and (b) runtimes. In both cases data size is shown on x-axis. The RMS graph shows that the algorithm behaves as expected in as far as that the error is decreasing as the data size increases. Thus, it provides insight about the number of biological experiments one would need to perform depending on desired accuracy. However, we should note that since the underlying graph has the form of a chain the presented graph is conditional to the real parameters we provided. Significantly different values for these, might shift the curve considerably. Graph (b) of Fig. 4 is evidence of a linear dependency between runtimes and size of data.

6 Experiment 2 - branching

In this Section we detail some experiments investigating the influence alternatives routes have on the performance of FAM.

6.1 Materials and method

We added an alternative path, of length 5, to the last Section's pathway program (see Fig. 5. This was added at the beginning of the chain-formed graph, and provided an alternative route from node 1 to node 5. We are interested firstly in the behaviour of the new program and secondly in comparing with the non-branching one.

We run the same experiments as before for the new SLP. Again, RMS for $\bar{p}$ and runtimes are used to measure accuracy and performance.

6.2 Results

From Fig. 6 we see that RMS and runtimes continue in the same pattern as in the previous case. That is, RMS is improving as population of learning set increases, and runtimes are linear to the same population.

However, Fig. 7 shows a significant quantitative difference when we compare the branching program to the non-branching one. Both RMS and runtimes show slack in performance.

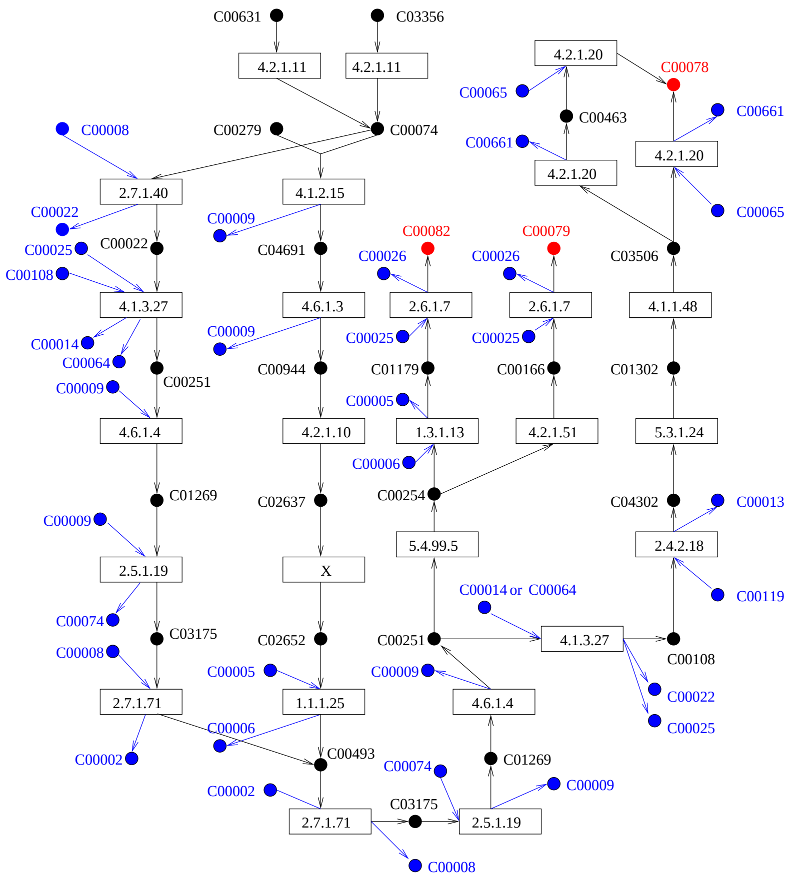


Figure 5: An alternative path added to the pathway of Fig.1. The addition is an artificial branch that in has no biological signifigance.

For RMS we need larger learning sets to achieve the same accuracy. We have identified two factors which probably contribute to this. Firstly, that there are more reactions, thus parameters, to learn. Secondly that since the choice between competing enzymes, at the top of the chain where the branch was attached, is a non-stochastic clause this will substantially favour one of the branches, most likely leading to an atrophic other-branch, for which the estimated values contribute a greater error.

Similarly, the deterioration of runtimes may be attributed to: (a) the existence of more nodes (b) higher set-up costs and (c) longer chains, since there are two ways to reach the 5th node. Both differences, in RMS and runtimes, and our suggested explanations, warrant further investigation.

7 Conclusions and further work

In this paper we have investigated the question of whether there exist important applications which require the machine learning of probabilistic representations. As described in Section 2 one example of this task involves the identification of novel enzyme functions given partial descriptions of metabolic pathways. The task, by its nature requires machine learning, probabilistic representations and relations. As usual such tasks can be divided into sub-tasks involving model building and parameter estimation. In

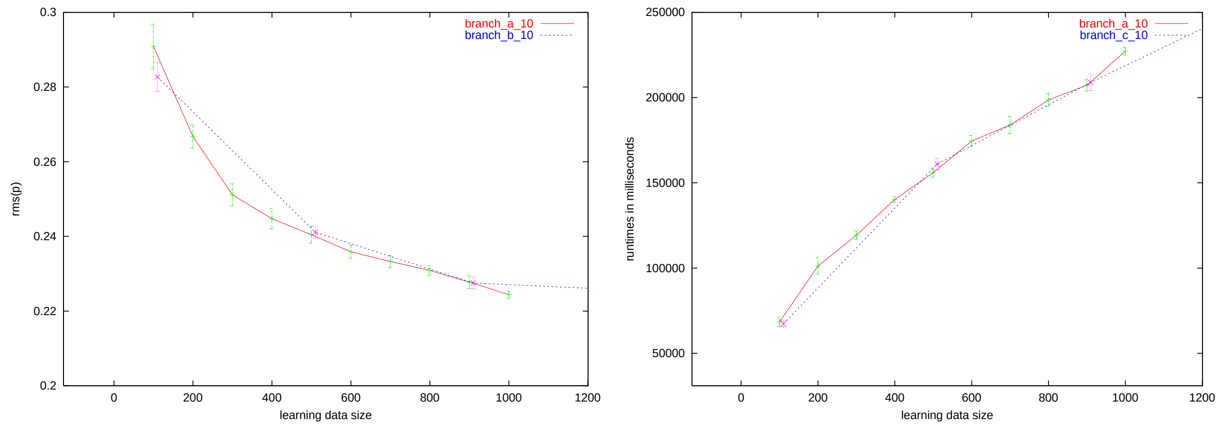


Figure 6: Left, (a) RMS for E_a and E_b at 10 iterations. Right, (b) runtimes for the same experiments.

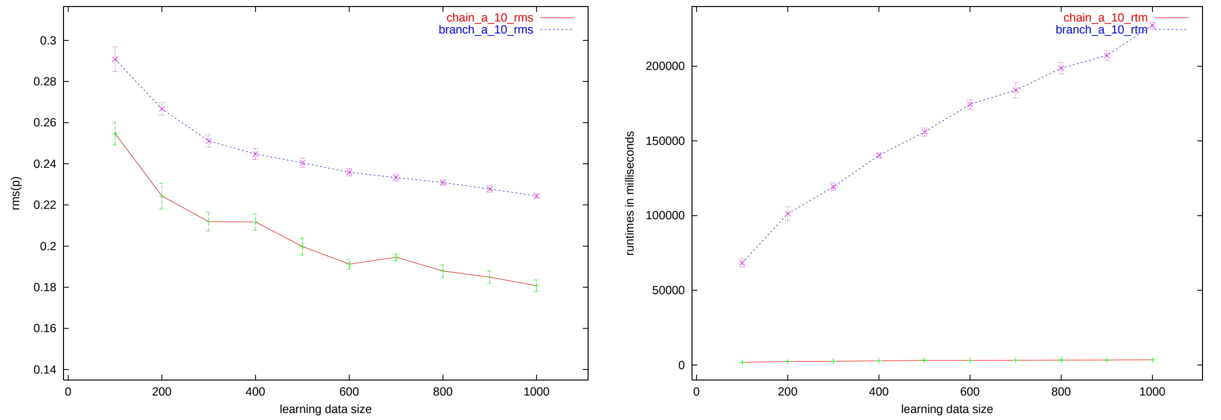


Figure 7: Comparison of RMS and runtimes for chain versus branched graphs

order to gain understanding of the sample size requirements for estimating the parameters of medium–sized metabolic networks we conducted experiments in which RMS error was estimated with increasing sample size. We separately investigated the effects of multiple pathway contributions to the concentrations of metabolites within the networks. For non-branching pathways the results indicate that parameter estimation is feasible from limited sources of data. However, the introduction of branches increases sample requirements considerably.

7.1 Further work

The work in this paper has addressed issues related to parameter estimation but not model building. In further work we would hope to apply techniques described in [8, 9] to this task. In addition we also intend to develop a theoretical characterisation of sample complexity for SLPs. The aim here would be to explain the sample complexity effects found in the experiments in this paper.

Acknowledgements

This work was supported partly by the ESPRIT IST project "Application of Probabilistic Inductive Logic Programming (APRIL)", (Grant reference IST-2001-33053) and the BBSRC/EPSRC Bio-informatics and E-Science Programme, "Studying Biochemical networks using probabilistic knowledge discovery" (Grant reference 28/BEP17011).

References

[1] Nicos Angelopouloos and Stephen Muggleton. SLPs for probabilistic pathways: Modelling and parameter estimation. Technical Report 2002/12, Department of Computing, Imperial College, London.

[2] C.H. Bryant, S.H. Muggleton, S.G. Oliver, D.B. Kell, P. Reiser, and R.D. King. Combining inductive logic programming, active learning and robotics to discover the function of genes. *Electronic Transactions in Artificial Intelligence*, 6-B1(012):1–36, November 2001.

[3] The Gene Ontology Consortium. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[4] J. Cussens. Loglinear models for first-order probabilistic reasoning. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence*, pages 126–133, San Francisco, 1999. Kaufmann.

[5] J. Cussens. Parameter estimation in stochastic logic programs. *Machine Learning*, 44(3):245–271, 2001.

[6] K. Kersting and L. De Raedt. Towards combining inductive logic programming with bayesian networks. In *Proceedings of the Eleventh International Conference on Inductive Logic Programming*, LNAI 2157, pages 118–131, Berlin, 2001. Springer-Verlag.

[7] D. Koller and A. Pfeffer. Probabilistic frame-based systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence, AAAI98*, pages 157–164. AAAI Press / The MIT Press, 1998.

[8] S. Muggleton. Learning stochastic logic programs. *Electronic Transactions in Artificial Intelligence*, 5(041), 2000.

[9] S. Muggleton. Learning structure and parameters of stochastic logic programs. In *Proceedings of the 10th International Conference on Inductive Logic Programming*, Berlin, 2002. Spriner-Verlag.

[10] J.K. Nicholson, J. Connelly, J.C. Lindon, and E. Holmes. Metabonomics: a platform for studying drug toxicity and gene function. *Nature Drug Discovery*, 1:153–161, 2002.

[11] International Union of Biochemistry and Molecular Biology. *Enzyme Nomenclature: Recommendations (1992) of the Nomenclature Committee of the International Union of Biochemistry and Molecular Biology.* Academic Press, New York, 1992.

[12] P.G.K. Reiser, R.D. King, D.B. Kell, S.H. Muggleton, C.H. Bryant, and S.G. Oliver. Developing a logical model of yeast metabolism. *Electronic Transactions in Artificial Intelligence*, 6-B2(024):223–244, November 2001.

[13] M. Kanehisa S. Goto, T. Nishioka. Ligand: chemical database of enzyme reactions. *Nucleic Acids Research*, 28:380–382, 2000.