Metabolic Pathway Correction

Computer Sciences Department
University of Wisconsin-Madison

CS 760 Course Project

Authors:
Chaitanya Gokhale
Hidayath Ansari

Supervisor:
Prof. David Page

1 Introduction

A metabolic pathway is a set of enzyme-catalyzed reactions through which an organism converts (e.g. synthesizes, degrades) one biochemical compound to another. This series of reactions may be incredibly complex in some cases, and considerable work has been done in experimentally determining these pathways for a number of reactions. Typically, a number of catalysts and metabolites are present during these reactions. The temporal profiles of concentration changes of each metabolite contain valuable information for determining the pathway structure. While measurement technologies are improving, they are still not at a point where they can provide accurate and reliable concentration measurements for each metabolite [**?**]. However, promising developments in mass spectrometry and magnetic resonance imaging may make such data available in the near future. The goal then, is to estimate metabolic networks given dense time series concentration measurements of the metabolites involved. In this project, we propose a method to correct a perturbed network given simulated time series data (for the correct network as experimentally determined).

2 Kinetics and Modelling Techniques

Conventionally, there have been two broad classes in which modelling and simulation fall into: Logical graph-based networks and differential equation models [**?**]. A graphical model such as the one we use qualitatively characterizes the reactions and their order. A differential equation model on the other hand seeks to quantitatively describe the parameters of the reactions. DE models require a large number of kinetic parameters to specify fully, and are not useful without them. King et al. have also presented an in-between model based on qualitative reasoning, chemical heuristics and inductive logic programming to recreate

metabolic networks [**?**]. Alternative ways of modelling these networks such as flux models, mechanistic models, and stoichiometric networks have been given a thorough treatment by Wiechert [**?**].

A promising tool for generating time series data for metabolic pathways is the Complex Pathway Simulator, or COPASI [**?**], which essentially models the reaction through differential equations and is very comprehensive and flexible.

3 Previous Work and Approaches

A large number of approaches have been tried in constructing metabolic networks purely from empirical data, aided with heuristics and gene regulatory network data. We provide here a short review of the salient features of some of them.

3.1 Optimization

Beasley and Planes [**?**] characterize the equations in a pathway through a set of purely mathematical constraints derived via stoichiometry and use optimization to develop solutions for them. Constraints are represented mathematically as equality and inequality relations, and are derived from considering the number of atoms and molecules on both sides of a chemical reaction and using conservation. A unique point of this work is that a distinction is made between high-presence compounds and low-presence ones, and concentration of the former is not quantified numerically. It is instead assumed to be available as needed. The objective function seeks to minimize the number of total reactions in the pathway, and also the number of excess ATP (adenosine tri-phosphate) molecules produced by following that particular pathway.

3.2 Nonlinear Searches

Velflingstad et al. [**?**] approach this problem in a two-front manner and frame it in terms of a regression task. The first part is inference of a model structure, and the second regression to fit parameters within that model. The authors compare various techniques such as S-systems, Lotka-Volterra models and other *nonlinear canonical forms* derived from biochemical systems and develop a nonlinear model which is to be linearized. A metabolic network is modeled as a biochemical network with time-derivative relations among the components of the system. These equations are then decoupled and linearized and four regression methods are analyzed to fit the data.

3.3 Pathway Holes

The method proposed by Green and Karp [**?**] is indicative of a broader approach: Using data from gene networks to strengthen assertions about metabolic networks. Using a partly constructed metabolic network, metabolites are found in known databases to fill in the "'holes."' Each unknown metabolite is known to have a function, and databases are searched to find all compounds with similar functions. The retrieved structures are then matched against the genome for that organism, and a probability measure found for each match. In this way incomplete networks are filled in.

3.4 Petri Nets

Nummela et al. [**?**] use bipartite graphs (petri nets) to model metabolic networks. They represent them using a bit-vector and use standard genetic algorithm techniques to come up with the most "'fit"' network. Their method is detailed and analyzed further in section 4.2.

3.5 Adapting reverse engineering algorithms

Nemenman et al [**?**] have adapted a reverse engineering algorithm from gene transcriptional networks to metabolic networks with encouraging results. Specifically, they generated synthetic data closely mirroring real profiles and conditions of red blood cell metabolism and applied an algorithm called ARACNE to it. Four sets of (publicly available) data were generated according to various parameter settings, three of which were steady-state and one dynamic. This is because a majority of transcriptional reverse engineering algorithms take steady-state data as input. The data was generated to account for noise, temporal properties, correlations, and other real-world phenomena. Like many other reverse engineering algorithms for that task, ARACNE works by probabilistically modelling dependencies among variables (metabolite concentrations in this case) and finding mutual information among those distributions. Several techniques are used to eliminate falsely correlated data and the algorithm then proceeds to build a network. The results obtained were close to par with benchmarks for transcriptional gene networks data.

3.6 Dynamic Bayesian Networks

Kim et al. [**?**] provide an overview of how discrete and continuous dynamic Bayes net models have been used to crunch high-throughput data to infer gene regulatory networks. They proceed to define a Bayesian "'goodness"' function for a network maximizing the posterior probability of a network and work out what it means in terms of time series data.

4 Possible Approaches

4.1 The DBN approach

Given that we have time series data for metabolite concentrations and a partially correct metabolic reaction pathway, one would naturally attempt to model is using Dynamic Bayesian Networks to . Ong et al [**?**], use DBNs for modelling relationships among genes using time series gene expression data. However, we think applying similar techniques for the pathway correction problem would not be optimal. For constructing the gene regulatory networks, the expression data could be easily modelled in binary form, since it is easy to assign a binary representation for activators and repressors. However, if we try something similar for our concentration time series data, it could result in a gross over-approximation, as explained in the following example.

One way to assign binary representation in our case would be to consider the change in concentration of the metabolite in successive time steps. We can assign 1, if the concentration increases while -1 if there is a fall in concentration. Let

A and B be two metabolites involved in the pathway. Now, suppose our Bayes Network has an edge from A to B, and B has an edge to C, i.e. A is a parent of B, and B is a parent of C. The algorithm will try to find probabilistic values for B being assigned value 1 or -1 given that A and C take some particular values. In our case, this structure would imply that reactant A gives intermediate product B which in turn is giving C. Now during the course of the reaction, there might be a number of data points at which $A = -1$, $C = 1$, and $B = 1$, and at the same time there could be many other points at which, for the same values of A and C, B takes value -1. This is because, the rate constants of these reactions will determine which reaction will be dominant in which phase. Thus, the probabilities computed by the DBN would be very fickle and wouldn't help much in learning the structure of the pathway.

The above is true for discrete DBNs. Continuous DBNs may be able to model this precisely; however they increase the complexity beyond what we propose. For this reason we don't analyze them in depth like the others.

4.2 Genetic Algorithms

Another approach is on the lines of Petri nets that uses Genetic algorithms. We can define a predicate `gives(A,B)`, that is true if A gives B in a reaction. Using such predicates one can model any pathway as a set of clauses and thus we can represent each clause as a bit-vector. Each such bitvector then forms our hypothesis. Starting with the bitvector for the given partially correct pathway, we can use operations such as mutation and crossover effectively to generate a number of different possible hypotheses. We will need an appropriate *fitness function* for scoring each hypothesis. The basic approach taken by Nummela et al in Petri nets [**?**] is similar. However, the fitness function that they use, and what one would have to use with this approach, involves numerically integrating the Kinetics equation, to estimate concentration values. We believe that such an approach would have some drawbacks.

For each time step, we need to numerically integrate the equation in order to arrive at our concentration estimate, which would involve a lot of computation. In addition, the estimates obtained in this manner could be prone to errors in data values, since the numerical integration method is aimed towards immunity from noise. The underlying assumption here is that the first-order kinetic equation holds. So slight deviations from the equation in the actual reaction could lead to significant errors in computaton. Also, this approach simply scores individual reactions, without any reward for two reactions, both having a good score, connected in the pathway.

The above potential pitfalls that we observed led us to come up with the proposed algorithm, which we believe should show improved performance against some such drawbacks.

5 Proposed Method

We detail in this section the components of the algorithm, such as the chosen data representation, a sampling scheme, learning parameters, a method to identify features and scoring them, and finally a method to combine all of these to identify the best candidate network.

5.1 Data representation

We choose a graph-based representation encoded as a vector to represent candidate hypotheses. This is done in order to closely capture the graphical relations from the network and facilitate manipulating hypotheses and incrementally modifying them.

The problem is to encode a given metabolic pathway. A natural graph representation is used, with nodes representing reactants and edges reactions. We use the notion of "'depth of a network,"' which is essentially the length of the shortest path from a root of the network graph. If there are several possible roots (nodes with in-degree zero), we pick one arbitrarily. At each depth we have all given metabolites as nodes. If the maximum depth of a network is d, and the total number of metabolites m, there are md nodes in the encoding graph. If we think of this graph as a matrix of m rows and d columns where higher column numbers correspond to deeper nodes, a cell in one column may have an edge to every cell in the next column as well as the previous column (reversible reactions), i.e. a theoretical possibility of $2m^2d$ edges. We assign a weight to each of these possible edges and serialize them within a single weight vector. This vector succinctly captures all the information in the metabolic graph and in addition also indicates our degree of confidence in each connection in the graph. We treat metabolites and catalysts as reactants, to completely capture all information available in the pathway. Zero-weight edges denote no connection between the compounds at that depth.

It can be argued that this data representation is somewhat overkill for the current method and use, but we chose this because it is extensible to many other possible algorithms involving iterations and probabilistic representation of edges in such networks.

5.2 Sub-sampling

Time-series data is unlike other datasets in the sense that an individual data point is not a complete and independent training example by itself. One data point in isolation cannot be used to incrementally improve or tune the parameters of a fitted model. For example, in other algorithms like k-means clustering, or k-nearest-neighbor, or Bayesian methods, every available data point can play a part in improving the accuracy of the model. In support vector machines, of course, a data point contributes to the model only if it is near the boundary, i.e. a support vector.

Additionally, if just one giant time-series dataset is available, there needs to be a way to split it in order to train, tune, and test the learned model. While multiple datasets may be generated if one is using a simulator, such luxuries may not be affordable in real data collection. To this end, we propose a simple sampling scheme, the utility of which will become clearer in a subsequent section on features.

We divide the original dataset into a number k of contiguous smaller subsets, the divisions either being arbitrary or according to a natural division by state or condition of the metabolic process. The sampling is now done by choosing n data points from each dataset at random (without replacement). Care must be taken to preserve the chronological order of the measurements. The subset division is done in order to prevent the random bunching of data points and

to have them spread out relatively evenly over most time periods. This set of nk distinct vectors forms a complete time series which may be used for training purposes.

5.3 Learning

Given a partially correct metabolic pathway network, we begin with its representation as a graph as described in 5.1. We wish to learn the pathway model that is closest to the actual using the time series data for metabolite concentrations. We begin by noting that for any reaction, the concentrations are governed by the rate law for that reaction. The order of the reaction and rate constants are the parameters that determine the relation between the rate of change of reactants/products and the concentrations of the reactants. As the rate law suggests, for an isolated reaction, the rate of change of concentration of the product depends on concentration of one or more reactants.

In our case we have a network of reactions. Thus, the rate of change of concentration any particular compound (which is represented by a node in our graph), should solely depend on the rate laws for each of the reactions in which the compound is involved. Now for a first order reaction, this relationship is linear. However, for higher order reactions it could get complex. As done in Petri nets [**?**], we make the assumption that the reactions involved follow no higher than first order kinetics. There are a number of ways in which such a relationship can be learned.

Under the first order kinetics assumption, we have a linear regression problem at hand. We choose the Support Vector regression technique for dealing with this problem. The SVMs are extensively used for linear regression. We use one SVM for each of the compounds involved in the reaction. The target function is the rate of change of concentration of that particular compound. The features we use for learning this are the concentration values of the reactants that produce the particular compound and the concentration of that compound itself. To learn an SVM regression model, we compute the rate of change of concentrations at each time step. Assuming that the time steps are sufficiently small, this rate of change will be just the change in concentration divided by the difference in time values.

Another option to consider however are artificial neural networks. We preferred the use of support vector regression because of two things: we can accommodate nonlinear dependencies as well by choosing an appropriate *kernel function*, and secondly because we can grow regression trees for each reaction and come up with different models for different phases of the reaction. This we think is particularly important because concentration of compounds may change differently during different times of the reaction and it can be easily modelled in this way.

One interpretation of the weights learned by an SVM for a particular reactant-compound edge is that they are an estimate of the rate constant for the reaction. However, in some cases where the nonlinear component of a relation was also captured while learning the coefficients for it, it is to treated just like a mathematical fitting of data. We score the performance of each SVM by comparing the predicted concentrations and actual values, as explained later. Our aim is to construct the correct pathway from the given network. The SVMs that seem to be performing badly on the data set are an indication of presence of wrong

edges or absence of correct edges. Using such indicators, we explore the space of networks obtained from the existing network by adding/removing an edge, and search for the the one that scores best. This is done by identifying features.

5.4 Identifying Features

Features are composed of two consecutive edges, i.e. a triplet of $\{R_i\} \rightarrow \{R_j\} \rightarrow \{R_k\}$. This is done in order to balance the notions of developing connectivity in the graph while keeping in mind that the most effective learning happens for localized areas in the graph. Potential features are generated from a given candidate hypothesis network by extracting all two-edge subgraphs spanning 3 consecutive depth positions, and by perturbing the candidate graph in all possible ways. Using a scoring mechanism outlined below we identify high-confidence features by observing the number of times each edge pair appears in a learned network and use this to update the weights in corresponding hypotheses.

5.5 Scoring

The regression formulae learnt as described above are used to score features. We take one sampling of the time series data, and for every point within that, predict the value for that variable in the next time step *in the original complete dataset*. The difference between these values is what is of concern, and these differences are summed up over all time points in the sample for each variable. The sums are then normalized for each variable by dividing it by the pre-computed standard deviation (over the entire dataset, again), and added for all variables (reactants) occurring in that feature. To account for noise and the minor mistakes that will invariably occur when comparing a real-world real number and a computed one, one could offset each individual score as follows: if the differece d is less than some pre-specified ϵ, we take it to be 0, and if it is larger, we consider its score contribution to be $d - \epsilon$. Of course, lower scores imply a higher confidence.

5.6 Putting it all together

After having described the parts above, we now outline how they may be combined in a coherent manner to infer metabolic networks. The dense time series data is taken, and a table Δ of changes in concentration between each pair of consecutive time steps is calculated. The guess network is taken and perturbed in all possible manners, each seeking to rectify known or predicted errors (such as reversal, addition and deletion of edges) in the network. This forms our candidate hypothesis set. From this set, we identify features as outlined above. Data from the Δ table is then used as per the learning procedure described above. Samples of the complete data set are taken, and scoring performed as outlined above. Each candidate network is then scored by summing the scores for all features associated with it. As an aside, this may be likened to a two-stage electoral process: The raw data is used to score features (general public voting for representatives), and the features are then used to either build or eliminate entire networks (representatives choosing a head). The difference in this analogy is that there is nothing being chosen; features and networks are being given a score. It is not a binary process and each score counts and contributes.

Note that since the networks at this stage are quite similar to each other, the scores will all be in the same range, but the differences are important. Now, a set of "'good"' candidate networks can be isolated, either by setting a threshold on the score or by picking the networks with k lowest scores. The process above can be iteratively repeated by perturbing each of these networks and evaluating the resultants, eliminating more networks and so on. Once a rising score trend is observed, we can stop this process and return the network with lowest score. The procedure as described above does not guarantee that the correct or even most optimal pathway will be found (largely because of a lack of formalism), but progress in this direction may be made by turning the iterations into a branch-and-bound algorithm with a concrete function for determining the lower bound of a network.

6 Conclusion and Future Work

A broad survey was conducted in the area of network construction from high-throughput profiling data for metabolic and gene networks. We learned the different methods that have been used to tackle this problem and others similar to it. Not much work has been done thus far in constructing or correcting metabolic networks from a purely computational standpoint involving only concentration data. Among the two approaches that we surveyed which are closest to this task [**?**, **?**], one is an attempt to transfer existing codebases to this task and the other a rather general treatment. Most other approaches to this problem try to include input from other experiments such as protein-protein interactions and binding site information and use the combined data to come up with a solution.

6.1 Theoretical comparison

We briefly compare our algorithm with the one for petri nets [**?**] and see how it may perform better.

- The fitness function that Nummela and Julstrom use in their genetic algorithmic solution is very general and has scope for being more specific to this task. For example, they assume hard linearity for differential equations representing the reactions and take differences. They also have to account for size of the network and penalize larger networks. Our approach on the other hand uses support vector regression (possibly nonlinear in a pseudolinear framework) to fit the relations. Also, we don't have to include a penalty term because we begin with a reasonably accurate guess network and merely modify a small part of it. Finally, our scoring function takes into account "'features"', i.e. a sequence of two reactions. We believe this would perform better than a naïve scoring method for single reactions as this would introduce a semblance of structure.

- The mutation and crossover functions in the above approach are also quite ad-hoc and can benefit from a more structured and directed search. For example, the changes caused due to mutation could be restricted to those that are changes over entire reactions, and high-scoring parts of networks could be crossed-over with each other.

6.2 Future Work

The most crucial work remaining is of course the implementation and determination of validity/accuracy of the above method. Aside from that, more work needs to be done on formalizing the ideas above and tuning function forms and minor variations of each step for best results. For example, the growing of regression trees and pruning techniques in the Learning subsection has to be further explored. The overall algorithm can also be generalized and cast as a branch-and-bound problem with a more concrete network scoring function that is robust and can determine lower bounds for a given network subject to a specified number of changes. The stopping condition can also be refined. The data representation is sufficiently general, and can be part of an extended algorithm based on this to reconstruct a metabolic network from scratch without a guess network.