Gene Network Reconstruction Using a Distributed GA with a Backprop Local Search

Mark Cumiskey, John Levine and Douglas Armstrong

School of Informatics, University of Edinburgh,
5 Forest Hill, Edinburgh EH1 2QL, Scotland
M.A.Cumiskey@sms.ed.ac.uk
johnl@inf.ed.ac.uk
jda@inf.ed.ac.uk

Abstract. With the completion of the multiple genome sequencing programmes the emphasis is now moving toward a functional understanding of these genes and their network interactions. Microarray technology allows for large-scale gene experimentation. Using this technology it is possible to find the expression levels of genes across different conditions. Gene network inference is the discovery of the inter-dependent relationships between genes based on this data. This paper investigated the use of a genetic algorithm using a backpropagation local searching mechanism to reconstruct gene networks from test data. This study demonstrates that the distributed genetic algorithm approach shows promise in that the method can infer gene networks that fit test data closely. Evaluating the biological accuracy of predicted networks from currently available test data is not possible. The best that can be achieved is to produce a set of possible networks to pass to a biologist for experimental verification.

1 Introduction

The accurate reconstruction of gene networks has many possible benefits and currently is the focus of much active research. This interest was sparked off by the development of a new technology that allows the massive scale measuring of mRNA. The presence of mRNA is indicative of whether a gene is active or not. The major areas of research that will benefit from gene network inference include: cancer research [12] (compared the profiles of healthy and cancerous tissue and identified differences is expression profiles); drug discovery [6]; toxicology [5]; disease prevention [2]. This paper investigates the use of genetic algorithms (GAs) in analysing the expression levels of genes toward constructing gene networks. A backpropagation local search feature is also developed.

2 Background and Microarray Technology

It is possible to measure on a large scale the levels of mRNA within a biological sample at different time points, or across experimental events, producing an expression profile of gene activity across these stages. Such profiles can be used to infer gene networks. A gene network can be described as a group of interdependent genes, when expressed have a common biological function. Also, 'gene network' encompasses any unknown variables that effect the operation of the network. The goal is ultimately to decipher the precise connections of the genetic network, identifying for each gene which other genes it influences.

Microarray technology provides the researcher with a snapshot of the levels of the protein precursor; the mRNA, within a biological sample. While it always has to be remembered that mRNA is only a precursor, and post-processing events are functionally important, the mRNA profile does give us an indirect measurement of gene activity in the sample.

The data sets retrieved from microarray experiments are massive. It is possible to snapshot an entire genome at once across various condition or time points. The yeast cell for example has around 6000 genes. For the data garnered to be useful a large number of readings must be taken. The resultant data sets are far too large to infer anything meaningful by hand.

3 Approach

A genetic algorithm with backpropagation local search was applied to the problem of inducing valid gene expression networks from given microarray data. In the GA population, each individual is a possible gene network, consisting of a collection binary links between genes and weights on each link. A specialist crossover operator has been developed which combines randomly chosen subnetworks from two parents in order to create a new network. In the initial networks found by the GA, the weights are set to coarsely grained values, to find a broadly acceptable configuration of gene interactions; these are then fine tuned by a backpropagation local search algorithm to find the weights which give the best match to the microarray data.

Simple String Representation

An important part of the workings of a GA is the definition of the individuals' (chromosome) representation. Figure 1 shows a sample network. An arrow entering a gene signifies that the gene is regulated either positively of negatively by the genes originating the arrow. The weight value is a measure of the level of influence, with a minus value denoting a repressing effect. In this model a string is used to define the gene network topology; this configuration is a short hand version of the weight matrix method described below. It avoids having to store large sparse matrices. A topology is made up of a series of nodes with weights; representing a gene and the effect of other genes on it. Each node has an id, and a series of node id's with associated weights, identifying influencing nodes and the strength of their influence as a weight. Nodes that are effected by other nodes are separated using a comma; each of these nodes has its influencing connection information enclosed by angle brackets; within these brackets is the from node id with the weight value, which indicates the level of influence. This representation is shown in Figure 3.

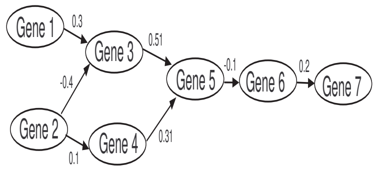

Fig. 1. A sample gene network with weights

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0.3 & -0.4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.51 & 0.31 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Fig. 2. A weight matrix representation of the gene network

Using a weight matrix as shown in Figure 2, first used by Weaver [1] is a well known method of representing the gene interactions of a regulatory network. The weight matrix consists of $n \times n$ weight values, each of which indicates the influence of one specific gene on another. Each column details the influencing genes for each gene identified by the row index. Figure 2 is a matrix representation of the network in Figure 1. With the row and column indices starting at 1 it can be seen that gene 3 (row 3) is influenced by gene 1 (column 1) with a positive regulation of 0.3 and is repressed by gene 2 (column 2) by -0.4.

The weight matrix can be further translated into a string as shown in Figure 3 representation for easy GA manipulation. This is the representation used by the GA.

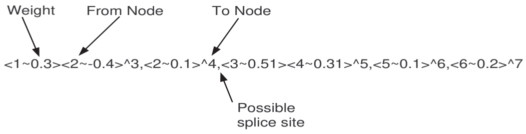

Fig. 3. A string representation of the gene network

Selection

Tournament selection is used to choose the individuals for crossover. In tournament selection a set number of individuals are chosen at random from the population and the best individual from this group is selected as a parent. A tour size of 10 is used. 10 individual are chosen randomly from the population; the most fit individual is chosen to reproduce.

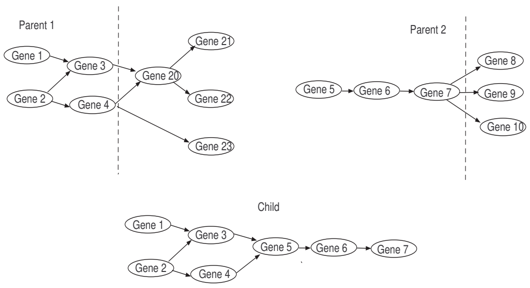

Fig. 4. Parent 1 and Parent 2 and a child network

Simple Crossover

One point crossover is used to inter-combine the subnetworks within individuals. One point crossover can occur at any gene position. Genes preceding and following the crossover point are considered separate groups; these groups are combined with the groups from another individual. Validity checks are made to ensure that the resultant configuration is a valid network, not just a gathering of random non-connected nodes. Figure 4 shows two networks of genes and their dependencies. After crossover the two parent networks may combine in a way that could possibly increases the individuals' fitness value. The resultant child network derived is shown below the parents. If a particular pattern has a higher fitness, it has a greater probability of surviving through future generations.

Mutation

A mutation setting of 0.001 is used to ensure population diversity. This mutation is present on the connections only. Weight mutation does not occur. After each generation an individual has a 0.001 probability of one of its nodes changing connection to another node covered in the data set.

Evaluating Network Fitness

Each network topology must be evaluated against the test data to identify if it is representative of this data. In a GA this is carried out by the fitness function. To evaluate a candidate network it must be translated into a form that can be measured against the test data. The idea is to find a network that explains the data. If a candidate network is correct than there is a relationship between the genes effecting a given gene, across all time steps. The procedure is generate a matrix of data in the same form as the test data, based on the candidate network and get a distance measure of how close this estimate is to the actual data. The steps are: for each gene in the candidate network get the estimated expression level based on the weights and connections; pass this value through a sigmoid function to impose a realistic biological limit; repeat for all genes in the candidate network, across all test data steps; get a distance measure between the estimate networks generated data and the test data.

$$s_i(t+1) = \sum_{j=0}^{n} w_{ji} x_j(t) \tag{1}$$

w - weights.
x - gene expression value.
s_i - gene expression estimate.

The above equation gets the estimated gene expression level for gene s at time $t+1$ by multiplying the expression level of each gene at time t by the proportional influence level of each dependent gene (from the gene dependency matrix). So, for each node in the network being evaluated that has dependency arrows arriving, the corresponding weight of these arrows is multiplied by the actual expression level (from test data) of the nodes (genes) originating the dependency arrows, this gets an expression level estimate for the gene receiving the dependency arrows. If the dependency weight values and the network connections are correct then the estimated expression value will be the same (or close) as the actual test data expression value.

$$x_i(t+1) = \frac{n_j}{1 + e^{-n_j(s_i(t+1))}} \tag{2}$$

The estimate gene expression value for each gene is passed through a sigmoid function for biological realism [1]. In weight matrix models the direct result of the step-wise calculation is often of an arbitrary magnitude, this being the result of the additive nature by which the input signals from all nodes are combined into the node of interest. To overcome this problem, it is necessary to scale this value to reflect what occurs in biological systems.

$$fitness = \sum_{i=0}^{n} \sum_{t=0}^{T} |y_i(t) - x_i(t)| + num/b \tag{3}$$

$i = 0..n$ - number nodes in the possible network.
$t = 0..T$ - across time steps/events.
y - actual expression levels.
x - estimated expression levels.
Num - number of nodes (genes in the topology).
b - network size bias.

Each sample network topology is evaluated against the target expression level, using equation 3. The estimated value x is found for each node in the network being evaluated using equation 1 and equation 2. Equation 3 gives a distance measure of how closely the estimated expression levels of the network being evaluated match the actual expression levels from the test data. The smaller the error distance the higher the fitness. There is an imposed bias toward smaller networks as known biological networks are relatively small.

Local Search

When the GA has completed it passes the discovered gene interconnections with the coarse weight settings (-1,0,5, 1 etc) to the backpropagation component. Its task is to finely tune the weight settings. In standard BP each weight is adjusted in proportion to its contribution to the overall error. Here, since the expression levels are unrolled across time steps (or events) it is necessary to get each weights contribution across all steps. During the update stage, the weights across all layers need to be updated by the same amount. The sum of the weight error is used, and the weights are updated equally, that is, the sum of adjustments over all layers for the weight is used for each weight update. With this restriction in place the neural network will minimise the error by adjusting the same weight for each gene across all steps. The end result being the weight settings that most closely explain the data, based on the candidate network presented. To compensate for this larger than usual update a low learning rate is set. This is a common approach in bptt [8] (backpropagation through time). For a good description of bptt see [9].

Training

Training is the process of getting the weights on the connections between nodes to adapt to a training data set by applying a set of training data sequentially to the input layer of the net and comparing the output to the (known) correct result. Following this, the weights are updated to reduce the error. In backpropagation [11] the error value at each output node is passed back to the preceding layer. The backpropagation algorithm causes each preceding layer to adjust its weights to neutralise the error; this procedure is repeated causing each preceding layer to adjust its weights in a similar manner. The bptt restriction on this is that the same node connection weight must be updated by the same amount across all the steps.

Island Model

Recent studies have shown that parallelisation can significantly improve the performance of genetic algorithms [3]. The use of the island model also reduces the likelihood of premature convergence, as each island has its own separately sampled initial population.

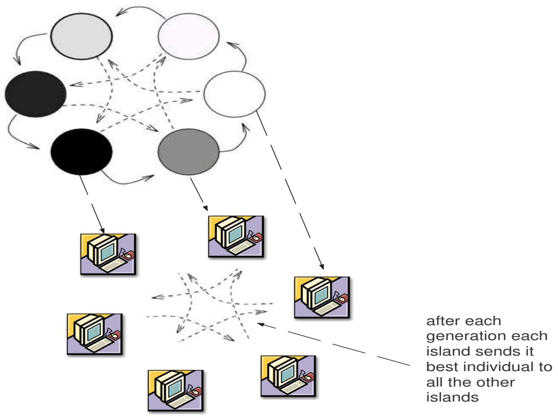

Fig. 5. In the island model several islands of sub-populations evolve simultaneously, with each island mapped to one processor. Each processor simultaneously evolves it own private population; migrating fit individuals at certain periods.

GA's are highly suitable for a parallel computation approach. Figure 5 shows the island model. Each island is run on a separate processor dealing with its own section of randomly generated individuals. At the end of each generation migrations occurs. Each island sends its best individual to all the other islands.

In this model, the population is divided into multiple sub-populations that evolve isolated from each other and exchange individuals occasionally. This methods keeps the network load light as bandwidth is only used to exchange fit individuals at the end of each generation. Also, less memory is used than a shared global population approach.

4 Results

For the experiments reported here, an island GA [13] running on up to 8 machines is compared with a GA running on a single machine. The microarray data used is the Rosetta Compendium [4] which is a data set consisting of 300 expression profiles of yeast (saccharomyces cerevisiae) mutants and chemical treatments. The results generated by the GA were evaluated in two ways: firstly, by how well they matched the microarray data, as judged by the fitness function, and secondly, by comparison with the Markov relations derived by Friedman [10] using his Bayesian network approach.

Testing whether a constructed network generated does in fact reflect what is happening biologically is difficult without submitting potential networks to a biologist for knock out experiments. There is not yet a framework available for comparing different networks generated by researchers on the same or different data sets. An effort here is made to compare the networks identified by the GA with the relations inferred by Friedman on the rosetta data set. These markov relations detail the effect of one gene on another, showing a confidence measure signifying the confidence in the relationship. A GA generated network is translated into a dependency pair listing and compared. For the single machine case, the best member of a population of 5000 randomly generated initial networks with a maximum size of 50 binary links has a fitness of well over 15000, showing it to be a very poor match to the microarray data (the perfect match would have a fitness score of 0). After running the GA for 19 generations, the best network has a fitness of 180.43. Backpropagation local search is then applied to fine tune the weight settings. The resulting gene expression network was found to contain 8 of the Markov relations found by Friedman. The data tables 1 and 2 show the sample test results. The headings are: Initial Pop, the starting population size; Net Size, the maximum size of the generated network; GA Fitness, the lowest (best) fitness values; BP Error, the lowest (best) error value; Markov matches, the number of corresponding pairwise connections found by Friedman; Time, execution time.

Table 1. GA plus backprop local search results on a single machine

Initial Pop	Net Size	GA Fitness	BP Error	Markov matches	Time
2000	50	186.60	2.032	6	30 min
2000	20	257.16	1.200	0	2 mins
5000	50	180.43	1.200	8	35 min
5000	20	221.40	0.8323	6	5 min

For the island model, the results are better: using 8 machines with a population of 2000 on each machine, the best coarse-valued network found has a fitness of 122.60. This network was found to contain 9 of the Friedman's Markov relations. Altering the number of islands, the number candidate networks in each population and the maximum number of binary links allowed in the gene expression network. The results broadly show that the technique works best with more islands, large populations and larger network sizes. The method also is computationally feasible: it takes between 6 and 42 minutes to converge on a solution.

Table 2. GA plus backprop local search results on a single machine

Num Machines	Initial Pop	Net Size	GA Fitness	BP Error	Markov matches	Time
4	2000	50	225.60	2.214	6	30 min
4	2000	20	257.16	1.334	2	7 mins
4	5000	50	223.43	2.200	4	42 min
4	5000	20	227.30	1.542	7	9 min
8	2000	50	122.60	2.635	9	30 min
8	2000	20	117.16	1.986	5	6 mins
8	5000	50	116.22	1.256	7	40 min
8	5000	20	132.30	0.623	5	11 min

A problem became apparent in that testing the validity of any sub-networks inferred was difficult. Given that biological testing of these predicted networks is a long-term problem we simulated network and ran the same algorithm on simulated microarray data. Removing all noise sources demonstrated that the GA approach was predicting valid relationships between the simulated gene nodes [7].

5 Current Work

Multi-network Representation

The approach described previously was good at finding small fit networks, containing two or three genes. The model currently under development uses this to good advantage by allowing these fragment networks to combine and grow. In the new representation an individual is a network consisting of a series of interconnected sub-networks. For computational operations each candidate is represented as a list of matrices, with each matrix representing a sub-network. These matrices are encoded as strings as described previously.

Figure 6 shows a network consisting of interconnected sub-networks, this representation is used to allow a local search crossover operator to find the sub-network interconnection configuration that achieve the highest fitness.

Multi-level Crossover

Crossover occurs at two levels, the sub-network level and the individual level. The sub-network crossover involves a one point crossover between two randomly

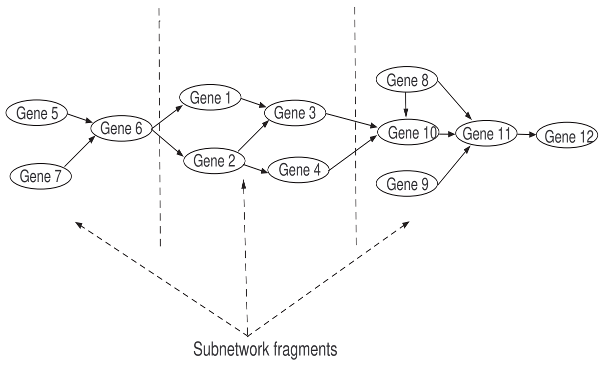

Fig. 6. A gene network consisting of three sub-networks (fragments)

chosen sub-networks from each of the individuals involved in the crossover. All of the sub-networks of the two individuals are split and then inter-combined. The higher level crossover method uses a form of local search; this operates by breaking up and recombining the individuals. An individual is a combination of interconnected sub-networks (fragments). Following the sub-network (level 1) crossover the newly formed fragments are separated and then recombined. This occurs for each possible placement of a sub-network in the individual. Picture a shuffling of sub-components until the overall system is as fit as it can be. These configurations are then measured for fitness; the two configurations with the highest fitness values are chosen to proceed. The advantage of this approach is that it stops the GA from converging on a small very fit sub-network.

6 Conclusion

The techniques for the reconstruction of gene networks are still in their infancy and are lacking a framework for the comparison and reuse of research results and data. What is currently achievable is to produce a set of possible networks to pass to a biologist for verification. The immediate goal is to reduce this set. Highly fit networks generated by the GA closely explain the test data; this reduces the set of networks that need further investigation. The island model markedly improves the fitness level achieved.

Bibliography

[1] C. Weaver et. al. Modeling regulatory networks with weight matrices, pacific symposium on biocomputing. *Journal of Computational Biology.*, 4:112–123, 1999.

[2] Carol A. et. al. Extraordinary bones functional and genetic analysis of the ext gene family. *Current Genomics*, 2, 2001.

[3] Goldberg et. al. The design of innovation, lessons from genetic algorithms,. *Technological Forecasting and Social Change.*, 1995.

[4] Hughes et. al. Functional discovery via a compendium of expression profiles. *Cell*, 102:109–126, 2000.

[5] Kim K et al. Identification pf gene sets from cdna microarrays using classification tress. Technical report, Biostatistics and Medical Informatics, University of WisconsinMadison, Madison, USA, 2001.

[6] Lawrence J. Lesko et al. Pharmacogenomic-guided drug development regulatory perspective drug evaluation and research . Technical report, Center for Drug Evaluation and Research, 5600 Fishers Lane Rockville, Maryland 20852 USA, 2001.

[7] Shin Ando.and Hitoshi Iba. Inference of gene regulatory model by genetic algorithms. *Proceedings of the 2001 IEEE Congress on Evolutionary Computation. Seoul Korea.*, 2001.

[8] Werbos P. J. Backpropagation through time, what it does and how to do it. *Proc. IEEE.*, 78:1550–1560, 1990.

[9] J. Hertz A. Krogh and R.G. Palmer. *Introduction to the Theory of Neural Computation.* Addison-Wesley, Redwood City CA., 1991.

[10] I. Nachman. Nir Firedman., with M. Linial. and D. Pe'er. Using bayesian networks to analyze expression data. *Journal of Computational Biology.*, 7:601–620, 2000.

[11] D. E. Rumelhart and J. L. McClelland. *Explorations in Parallel Distributed Processing.* MIT Press. Cambridge, Massachusetts., 1988.

[12] Richardson S. Mixture model for the identifying gene expression factors in survival analysis using microarray experiments. Technical report, Department of Epidemiology and Public Health, Imperial College, London, UK., 2001.

[13] Darrell Whitley. A genetic algorithm tutorial. Technical report, Computer Science Department, Colorado State University, Colorado State University Fort Collins, CO 80523.