Using Coarse-Grained, Discrete Systems for Data-Driven Inference of Regulatory Gene Networks: Perspectives and Limitations for Reverse Engineering

Dirk Repsilber*, Jan T. Kim†, Hans Liljenström*, and Thomas Martinetz†

* Institute for Biometry and Informatics
SLU, Johan-Brauners väg 3
S - 75733 Uppsala, Sweden
Dirk.Repsilber@ebc.uu.se
hans.liljenstrom@bi.slu.se

† Institut für Neuro- und Bioinformatik
Seelandstraße 1a
D - 23566 Lübeck, Germany
{kim,martinetz}@inb.mu-luebeck.de

Abstract

This contribution gives an initial report of a new project exploring the perspectives and limits of reversely engineering regulatory gene networks from gene expression data. The availability of such data is currently increasing dramatically due to the microarray technology. However, inferring the underlying network from expression data is difficult. We address the reverse engineering problem by simulating the regulatory network and the process of extracting expression data. The simulated expression data thus obtained are then subjected to an algorithm for inferring regulatory networks. With this concept, we intend to assess the effects of statistical properties of the data extraction process on the suitability of the expression data for reverse engineering.

1 Introduction

It is well known that the genetic information of an organism, stored in its genome, is used to determine the phenotypic traits of that organism. Consistent with the central dogma of molecular biology, the genome itself is not modified in this process. Rather, processes such as cell differentiation during development and phenotypic adaptation to varying environmental conditions are achieved by differential expression of genes. Recent advances in molecular biology have revealed that gene expression is controlled by genetic information through networks of regulatory genes. These findings have been reflected in many computer models of evolution and of development [5, 6, 13, 4, 8]. All these models are abstract and highly simplified in their respective ways, due to constraints in computational complexity as well as due to scarcity of data which could be used for parameterising more detailed models.

Since the last few years, an increasing number of genomes have been completely sequenced. Recent technological advances, in particular microarray technology, now allow direct measurements of gene expression on a genomic scale. This raises the challenge of using such expression data for inferring the architecture and dynamic rules of genetic networks. For an introductory review see [11]. This challenge is also referred to as the reverse engineering (RE) task [3]. At the same time, availability of expression data opens up new perspectives for modelling.

The size of the state space of regulatory networks i.e. the number of networks that can be constructed with a given number of nodes (genes) and links (regulatory interactions) is subject to extreme combinatorial growth as the number of genes increases. The complexity of these networks and the number of parameters that need to be fitted from data frustrate all attempts to reverse engineer any, but the smallest and simplest regulatory networks from data by brute force (e.g. by exhaustively sampling the state space).

Thus, reverse engineering requires considerable abstractions and simplifications in order to tackle biologically relevant regulatory networks. Even given the enormous amount of sequence and expression data, it is still not realistic to attempt reconstructions of regulatory networks in full detail. All we can try, at the moment, is obtaining coarse-grained models of regulatory networks. Therefore, it is necessary to investigate in which respects such coarse-grained network reconstructions provide insights into true biological regulatory gene networks.

In the current project, we intend to approach this issue by generating simulated expression data with a detailed computer model of a regulatory network, and using these data to assess the performance of a reverse engineering system. For simulating expression data, we use `transsys` [9], a detailed and powerful system for modelling regulatory networks. The reverse engineering system is `HypoNet` [10], a system which uses a genetic algorithm to evolve regulatory networks which generate expression patterns similar (or, ideally, identical) to the observed data.

Among the many approaches to reverse engineering regulatory networks, two assumptions are frequently made. First, it is often assumed that the networks are small, usually limited to no more than 10 to 20 interacting genes. This assumption can be motivated biologically from studies on the number of interacting genes for quantitative trait loci in plants and in *Drosophila* [2, 7]. Following this argument, we will (at least initially) investigate relatively small networks. Biologically relevant networks can thus be defined as groups of genes, which work together in controlling and regulating a trait by interacting and regulating the expression of the genes in the network (and possibly additional genes which then actually ”implement” the trait). Secondly, it is frequently assumed that genes can be modelled to have only a small number of discrete activity states. This assumption can be motivated by the observation that phenotypes resulting from mutations of a given locus can often be grouped into two or a few distinct classes. It is unclear, however, whether this argument is relevant to the level of gene regulatory networks.

Regardless of biological motivation, discrete state modelling is convenient for using microarray data. As these data are often quite noisy, it is common practice to discretise the primary relative expression levels into ”fold-increase” values, i.e. to group transcript-levels into classes which are n-fold increased or decreased in expression, with respect to some reference condition. As a special case of this approach, genes are classified as ”up-regulated”, ”not regulated” or ”down-regulated”. Current technology also induces a strong discretisation on the temporal axis, as each temporal sample requires an individual microarray.

In the initial phase of the current project, we discretise the real-valued, simulated expression data from `transsys` with respect to (sampling) time and to expression levels. These discretised, simulated samples are then used to assess the effects of discretisation on the reconstructions obtained with `HypoNet`. The concept of our approach is illustrated in Fig. 1.

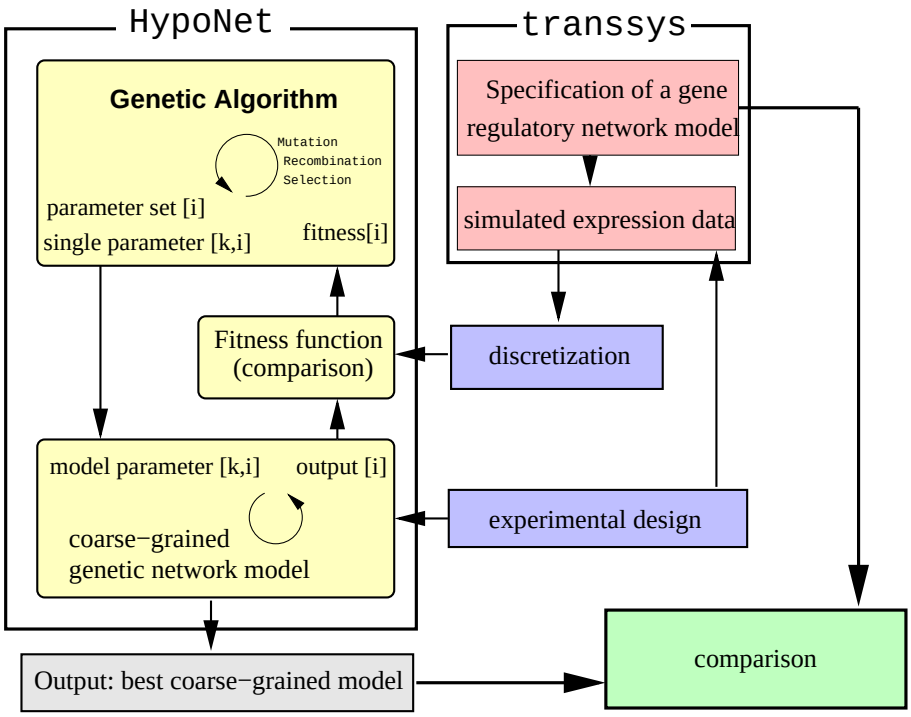


Figure 1: The general framework for using simulated regulatory networks for evaluating network inference by reverse engineering driven by simulated expression data.

2 Model and Methods

2.1 Reverse Engineering with `HypoNet`

For the reverse engineering, we use the `HypoNet` system [10] which employs a genetic algorithm (GA), as in [12], to generate a discrete, multi-state network which produces expression patterns that are as consistent with the observed data as possible. Observed data are input as a set of time series, measured upon exposing the biological system to different environmental conditions.

`HypoNet` can be used in different ways: First, it can generate a network without any information, in addition to the discretised expression data. Secondly, `HypoNet` allows the incorporation of *a priori* knowledge as hypotheses. A hypothesis specifies edges in the network, or parts of transition tables of the nodes. Such information may *e.g.* be known from additional experiments or, more generally, represent *a priori* biological knowledge. When given a hypothesis, `HypoNet` employs the GA only to evolve those network components which are not specified. Finally, `HypoNet` can be applied to different hypotheses. This mode allows checking which hypothesis is most compatible with the experimental data.

In this initial phase of our project, we use hypotheses only to specify N, the number of nodes, and K, the number of links per node. For the future, however, we intend to provide the system with more elaborate hypotheses. This will allow us to

- evaluate the power of the RE-algorithm to discriminate false hypotheses
- identify key elements which, when provided in hypotheses, improve the reverse engineering results

The latter item is of particular interest, as this may provide a basis for the rational design of additional experiments in the long run.

2.2 Simulating Gene Expression Data with `transsys`

`transsys` is a formal language for describing and "programming" regulatory gene networks. Using a given `transsys` network, time series of expression data are generated, starting with random expression levels as initial conditions. To enable reverse engineering, regardless which algorithm to be used, it is proven crucial to supply time series of expression data for a variety of different experimental (environmental) conditions [1]. In our case, multiple environmental conditions are simulated by generating series with multiple, different initial conditions.

For our initial tests, we generate `transsys` networks with N genes and K regulatory elements per promoter at random. Here, regulatory elements are conceptualised descriptions of transcription factor binding sites and the effect (activation or repression) induced upon binding. `transsys` networks contain many real-valued quantities:

- A decay rate for each factor
- A maximum level of activation or repression for each regulatory element
- A value describing the affinity of each regulatory element to its binding factor
- A constitutive expression level for each gene (*i.e.* the expression level that is adopted for absence of regulatory factors)

These parameters have straightforward biological motivations, but no correspondence in the discrete, multi-state networks generated by `HypoNet`. For our initial tests, we use `transsys` networks in which these values are identical for all factors or regulatory elements, respectively. This allows us to focus on reconstruction of the network topology rather than the details of the dynamic rules. It should be noted, however, that `transsys` is suited for investigating more complex and realistic networks in the future; of course, each factor and each *cis*-regulatory element can be parameterised individually, and parameters can even change dynamically. It will be interesting to test `HypoNet` with `transsys` networks with identical topologies but different dynamic parameters.

2.3 Discretisation

Expression levels for all genes in the `transsys` network are normalised to $[0, 1]$ and subsequently discretised by dividing $[0, 1]$ into S intervals of equal width.

The normalisation aims at generating input-values comparable to microarray-data, where only log-ratios are available and absolute levels of gene expression are essentially incomparable between genes. The discretisation step, on the one hand, aims at modelling the high error level for experimental global expression data, which makes these data suitable for representation as qualitative values. On the other hand, the discretisation is a step towards a simplified model and results in a finite search space. Finally, sampling (temporal) discretisation is simulated by keeping only a small set of samples from equidistant time points. An explaining example for both discretisation procedures can be found in Figure 2.

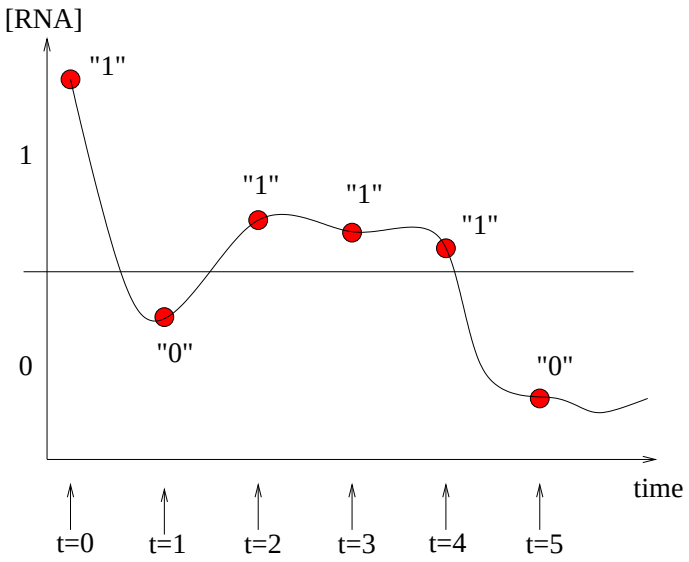


Figure 2: Discretisation of (simulated) expression data. Discretisation is in both sampling and expression level: In this case 6 equidistant time-points were chosen to sample. The discretisation cutoff is indicated. Sampled values above this cutoff are discretised as "1", samples with values below as "0" (for this Boolean example).

2.4 Comparing Reverse Engineering Results with the Original Model

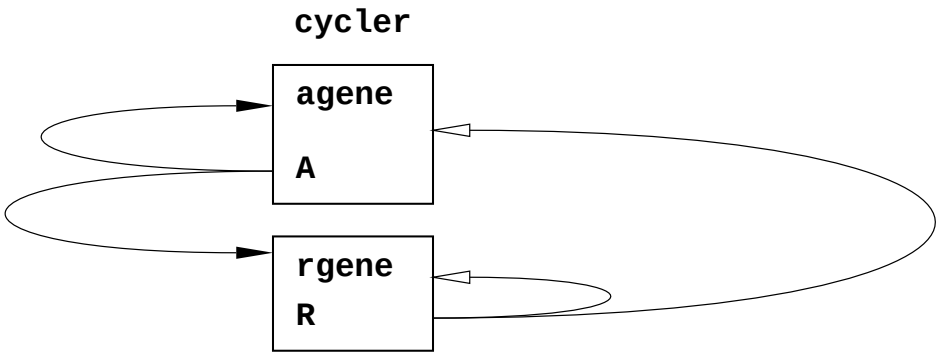
The `transsys` software package includes tools for graphically rendering `transsys` networks, and tools for systematic comparison of networks are under development. These will be employed for our study as they become available. Currently, we have a tool for translating the topology of a discrete, multi-state network generated by `HypoNet` into a `transsys` network, and we use this tool for visually comparing the networks obtained by reverse engineering with the original `transsys` network.

3 Results

In our first approach we tested two `transsys`-networks (see Figures 3 and 6 respectively), a most simple case, the `cycler`-network, and a more complicated random example for $N = 5$, the `xrnd`-network.

Figures 3, 4 and 5 show data illustrating the process of generating simulated expression data using `transsys` and running `HypoNet` to obtain reconstructed networks for a most simple example. Results of reverse engineering with `HypoNet` are shown in Fig. 5. The genetic algorithm was run for 100 generations with a population size of 50 individuals. Further parameters of the RE-GA were a mutation rate of $r_{\mathrm{mut}} = 0.5$ per individual and reproduction event, a cross-over probability of $r_{\mathrm{cross}} = 0.5$ per reproduction event and a generation gap of $\Delta g = 0.1$. Selection was carried out with the roulette wheel selection procedure, with keeping the individual with the highest fitness value in the GA population.

A more complex network is shown in Fig. 6. The topology of this `transsys` network has been generated randomly. The corresponding reverse engineering results are shown in Fig. 7. No network obtained by reverse engineering reflects the topology of the original `xrnd` network correctly. The examples provide an impression of the variability of the results. A more detailed




```
transssys cycler
{
  factor A { decay: 0.1; }
  factor R { decay: 0.1; }

  gene agene
  {
    promoter
    {
      constitutive: 0.01;
      A: activate(0.01, 1.0);
      R: repress(0.1, 1.0);
    }
    product
    {
      default: A;
    }
  }
```

```
  gene rgene
  {
    promoter
    {
      A: activate(1.0, 10.0);
      R: repress(1.0, 1.0);
    }
    product
    {
      default: R;
    }
  }
}
```

Figure 3: The `transssys` program `cycler`, shown as a network graph (top) and as `transssys` program code (bottom). In the graph, activating interactions are shown as arrows on the left side, ending in a closed tip, arrows on the right side and ending with an open tip depict inhibitory interactions.

analysis of these results will be conducted when tools for graph based similarity analyses for `transssys` networks are ready.

4 Discussion

The main problem we address here concerns how the discretisation is done for both expression levels and data sampling, and which consequences this has for reverse engineering possibilities. Here, the advantage of using simulated expression data is that the topology and dynamical rules of the underlying system are known, whereas in the case of molecular data, this information is largely unknown.

The discretized, simulated expression data generated with `transssys` are indistinguishable from empirical data while the underlying dynamical system, *i.e.* the `transssys` program, is known. This allows us to systematically investigate the effects of data sampling and discretisation on the results of reverse engineering with `HypoNet`, and to elucidate the perspectives and limitations of reverse engineering in general.

The `transssys` network `cycler`, shown in Fig. 3, consists of two transcription factors and two genes. Its dynamics is oscillatory over longer periods. However, this is not strongly reflected in the

condition 1	condition 2	condition 3	condition 4	condition 5
1 1	0 0	0 1	1 0	0 1
0 1	0 1	0 0	0 1	0 1
0 1	0 0	0 0	0 1	0 1
0 0	0 0	0 0	0 0	0 0
0 0	0 0	0 0	0 0	0 0

Figure 4: Five example time series of expression levels obtained with `cycler`, started with different random initial conditions representing five different experimental conditions, and discretised to $S = 2$ states. Each column gives the time series for one of the two genes in this network, while each row gives the discretised expression values for a single time-point. Note that, for conditions 3 and 5 the initial values ($t = 0$) are equal, but the single time series are different. This is due to information loss in discretisation and disables full reverse engineering (see text).

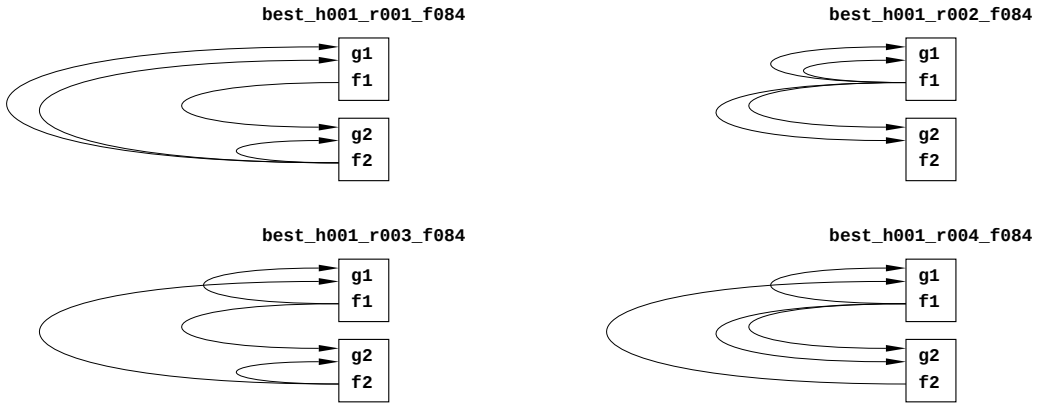


Figure 5: Four multi-state networks obtained from independent runs of `HypoNet` on discretised expression data generated with `cycler` starting with 20 different initial conditions. Networks with maximal fitness after terminating the genetic algorithm are shown. The networks are rendered as `transsys` networks; all links are shown as activation links in this process. Thus, the network `best_h001_r003_f084` (bottom left) reflects the link structure correctly. However, this structure is not favoured by fitness, as all networks shown here have the same fitness (0.84).

discretised data generated with `cycler`, as shown in Fig. 4. The five time series, consisting of five samples each, provide an impression of the scarcity and the coarse-grainedness of the input data with respect to the full network, especially when comparing these data to the full detailed structure shown in the listing in Fig. 3. However, it is just this scarcity which is typical for expression-level experiments, carried out using the microarray technology. In the reverse engineering results, the correct network topology does appear, as exemplified by `best_h001_r003_f084` in Fig. 5. However, this topology is not favoured by particularly large fitness value. As a tentative interpretation of this finding, we assume this to be an indication that the input data is insufficient for inferring the original network topology, due to a too coarse-grained discretisation either in the expression level or in the temporal dimension, or both. For example, as in the case for conditions 3 and 5, discretisation in the expression levels led to two time series which are incompatible at the level of the discrete deterministic network models: It does not exist a model which could represent both time series correctly. Hence, a finer representation of possible expression states, *i.e.* increasing S, might improve the discrete representation of the dynamic data and the subsequent reverse engineering. However, preliminary explorations of this hypothesis indicate that increasing S, the number of states, results in reduction of variability in reverse engineering results with `cycler`. It appears that the correct topology is obtained more rarely, not more frequently. Increasing temporal

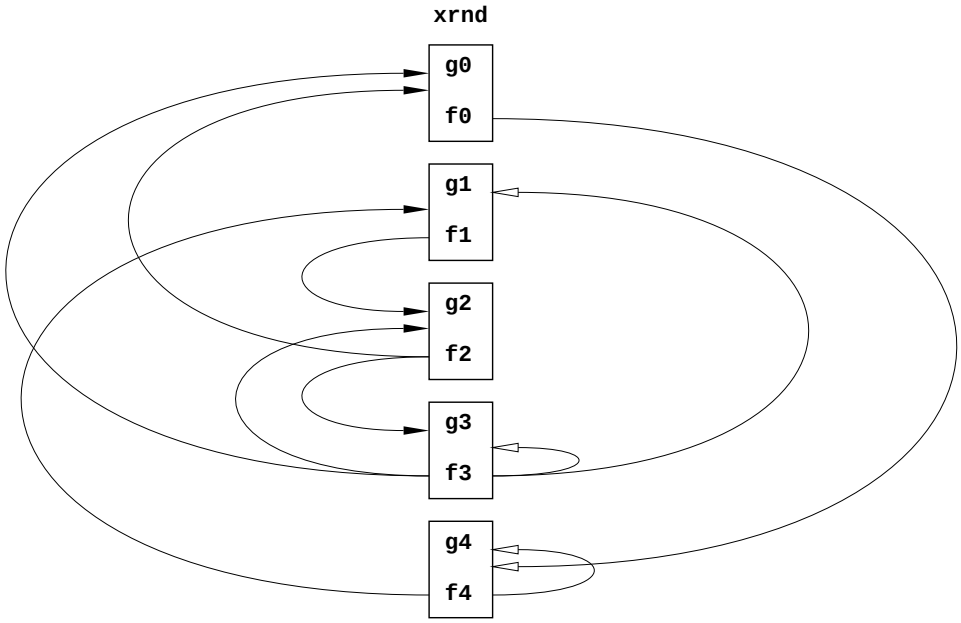


Figure 6: `xrnd` is a randomly constructed `transsys` network with $N = 5$ genes and $K = 2$ links per gene. All remaining parameters are the same for all genes.

resolution has not led to conclusive results.

Both studies have suffered from the fact that increasing resolution does not necessarily improve the amount of information in the simulated expression data. Discretising larger data sets frequently results in a strong over-representation of one particular state; in many cases, a pronounced predominance of the state 0 or the state $S - 1$ (i.e. the state corresponding to maximal activity) is observed. These observations may indicate that a more sophisticated method for discretisation is called for. As an example we plan to try a classification resulting in a uniform distribution of the discretised data.

A denser temporal sampling additionally often results in longer repetitions of rows in the input data. Insertion of additional zeros...

5 Summary and Outlook

Phenotypic processes, such as cell differentiation, development and genetically informed adaptation to changing environments are all mediated through dynamics in networks of regulatory genes. Currently, micro-arrays allow genome-wide access to gene expression data. Yet, the size of the space of gene regulatory networks, and the complex structure of this space, preclude straightforward inference of the topology and the dynamic parameters of regulatory networks from current gene expression data. Noise and limited resolution render expression data coarse-grained. These coarse-grained data can be used for obtaining correspondingly coarse-grained models of regulatory networks. It is, however, an open question how much information about the real, underlying regulatory network is conveyed by such coarse-grained models.

In this contribution, we introduce a project to address this issue by a modelling approach. We present the concept of using `transsys`, a software system for detailed modelling of regulatory networks, to generate simulated expression data. These data are then subjected to a normalisation

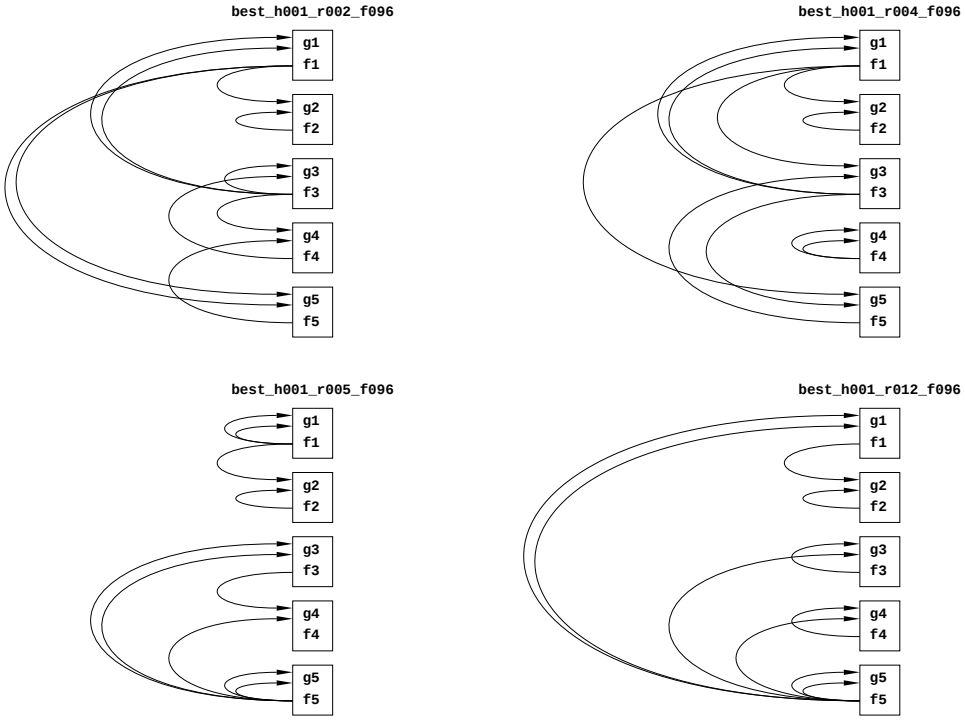


Figure 7: Four multi-state networks obtained from independent runs of `HypoNet` on discretised expression data generated with `xrnd`. 20 different initial conditions were used to compute expression levels at 80 time steps. The simulated expression data were then discretised to $S = 2$ states, and 5 samples drawn equidistantly were stored in the input set for `HypoNet`. As in Fig. 5, all links are shown as activation links. All networks shown here have the same fitness value, 0.96, which was the highest value seen in 50 independent runs.

and discretisation procedure which simulates expression measurement using current technology. The resulting data are then used for reverse engineering with `HypoNet`.

Currently, the software framework for implementing this concept has been established, and it has been used for preparing preliminary results. The correspondence between the detailed dynamic system described by a `transsys` network, and the coarse grained models obtained with `HypoNet` is indeed not trivial, indicating that our system has captured some aspects of the reverse engineering problem. In some initial observations, we report that improving data density does not necessarily improve reverse engineering performance.

In a longer perspective, we intend to develop our system to be able to generically model the statistical properties of gene expression measurement techniques. Even our initial explorations have illustrated that unfavourable statistical properties, such as the tendency to generate repetitive samples, can limit the amount of knowledge about the regulatory network which can be retrieved from the expression data. Such limitations cannot be overcome by simply increasing the data sampling density. Rather, the choice of data acquisition methods will have to be guided by the questions about the regulatory structure which shall be answered by the data. Computer models of regulatory systems can be expected to be a nice source for investigating the principles which can provide such guidance.

References

[1] Tatsuya Akutsu, Satoru Miyano, and Satoru Kuhara. Inferring qualitative relations in genetic networks and metabolic pathways. *Bioinformatics*, 16(8):727–734, 2000.

[2] C. Damerval, A. Maurice, J. M. Josse, and D. deVienne. Quantitative Trait Loci Underlying Gene Product Variation - A Novel Perspective for Analyzing Regulation of Genome Expression. *Genet.*, 137(1):289–301, 1994.

[3] Patrik D'haeseleer, Shoudan Liang, and Roland Somogyi. Genetic network inference: From co-expression clustering to reverse engineering. *Bioinformatics*, 16(8):707–726, 2000.

[4] Kurt Fleischer and Alan Barr. The multiple mechanisms of morphogenesis: A simulation testbed for the study of multicellular development. In Christopher G. Langton, editor, *Artificial Life III*, Santa Fe Institute Studies in the Sciences of Complexity, pages 379–416, Redwood City, CA, 1994. Addison Wesley Longman.

[5] Stuart A. Kauffman. Developmental logic and its evolution. *BioEssays*, 6:82–87, 1987.

[6] Stuart A. Kauffman and E. W. Weinberger. The NK model of rugged fitness landscapes and its application to maturation of the immune response. *J. Theor. Biol.*, 141:211–245, 1989.

[7] M. J. Kearsey and A. G. L. Farquhar. QTL analysis in plants; where are we now? *Hered.*, 80:137–142, 1998.

[8] Jan T. Kim. Lindevol: Artificial models for natural plant evolution. *Künstliche Intelligenz*, 1/2000:26–32, 2000.

[9] Jan T. Kim. `transsys`: A generic formalism for modelling regulatory networks in morphogenesis. In Jozef Kelemen and Petr Sosík, editors, *Advances in Artificial Life (Proceedings of the 6th European Conference on Artificial Life)*, volume 2159 of *Lecture Notes in Artificial Intelligence*, pages 242–251, Berlin Heidelberg, 2001. Springer Verlag.

[10] Dirk Repsilber, Hans Liljenström, and Siv G. E. Andersson. Reverse engineering of regulatory networks: Simulation studies on a genetic algorithm approach for ranking hypotheses. eingereicht bei Biosystems, 2001.

[11] R. Somogyi and C. A. Sniegoski. Modeling the complexity of genetic networks: Understanding multigenic and pleiotropic regulation. *Complexity*, 1(6):45–63, 1996.

[12] Mattias Wahde and John Hertz. Coarse-grained reverse engineering of genetic regulatory networks. *BioSystems*, 55:129–136, 2000.

[13] Stewart W. Wilson. The genetic algorithm and simulated evolution. In Christopher G. Langton, editor, *Artificial Life I*, pages 156–166, Redwood City, CA, 1989. Addison-Wesley.