Automatic Analysis of Developmental Regulatory Networks

Jonathan Carter
NERSC, Lawrence Berkeley National Laboratory

Antonio Garcia-Martinez
Department of Electrical Engineering and Computer Science,
University of California, Berkeley

Bill Kramer
Department of Electrical Engineering and Computer Science,
University of California, Berkeley

May 23, 2003

Abstract

Over the past decade, the ability to accurate sequence genomes has improved to the point where today, it is possible to acquire complete genomic sequences of organisms within a day. This makes it feasible to start to study the interaction genes and other signal information have with the genome. The effort described in this paper is to use the manual analysis of a gene regulatory network, which took years of effort to produce, and use it to see if, given a set of data of what genes are influenced when other genes are active or inactive, it is possible to automatically determine the regulatory network with any degree of accuracy.

1 Introduction

Over the past decade, the ability to accurate sequence genomes has improved to the point where it is possible to acquire complete genomic sequences of or-

ganisms within a day. This makes it feasible to start to study the interaction genes and other signal information have with the genome. This process, the influence one gene has on others - either helping turn target genes on or off - is the regulatory network, and the act of turning a gene on or off is called *transcription regulation*.

Transcription is initiated when RNA polymerase begins to unravel the two strands of DNA and starts transcribing the DNA sequence into an RNA. For RNA polymerase to bind to the transcription start site (TSS), other pre-initiation protein complexes generally have to bind to the DNA just upstream (with respect to the direction of transcription) of the gene in question. The intergenic region close to the start of the gene, called the *promoter* region, contains small sets of sequences where *transcription factors*, can selectively bind. It is common that combinations of transcription factors are needed to influence the transcription. The result is that the relationships between genes can be expressed as a graph, where the genes are considered nodes or vertices.

Directed edges indicate whether or not one gene influences another. The influence of a gene can either cause the target gene to express (turn on) or repress (turn off) and hence the edge can have weight of 0, 1, or -1. Once in this form, graph algorithms, such as motif finding, can be used for further analysis.

Only a few regulatory networks have been explored to an unambiguous degree, in part because of the expense and time required to experimentally determine the complex relationships. However, recent advances in *microarray technology* make it possible to rapidly assess whether a specific gene is active, and to do it in a cost-effective way. Thus the genomics community is on the verge of an explosion of expression data which can only be analyzed by automating the ability to find transcription relationships between genes.

Microarrays are two-dimensional grids of expression data where time is along one axis and different genes (or mutant forms of the same gene) are along the other. Arrays give a broad, comprehensive view of the state in the cell or embryo, see Figure 1.

Fortunately, a few well-studied organisms like *Escherichia coli* and *Saccharomyces cerevisiae* have had their expression patterns studied for decades using techniques that are slower than micro-arrays. One of these studies by a large team lead by Eric Davidson recently published the proposed transcription network for the embryonic development of the sea urchin *Strongylocentrotus purpuratus*.

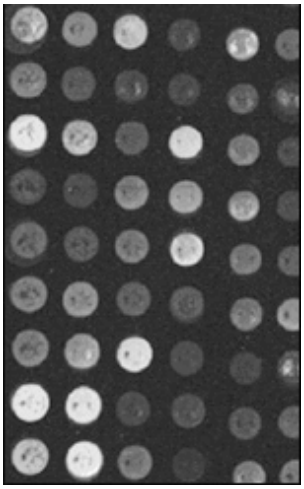

Figure 1: Image of micorarray

The aim of this paper is to automate the network inference process undertaken by Davidson and completed following years of painstaking peicing together of the regulatory puzzle. While no piece of software can seriously aspire to replace the bench experimentalist's hard-won intuition, we hope that by trading a little accuracy for a lot of time, we can expedite what will likely be a common procedure in the near future.

2 Regulatory Gene Networks in the Sea Urchin Embryo

The sea urchin embryo offers many advantages for study of developmental regulatory networks. The embryo shows considerable development in only 36 hours, and develops into a simple larva compared with insects or vertebrates. In addition, the molecular and developmental biology of the sea urchin has been extensively studied, and dozens of regulatory genes have been cloned and their expression pattern well described. Furthermore, the genome has been somewhat characterized.

For this reason, many research groups are actively performing experiments with sea urchin embyros. Recently, Davidson and co-workers published a provisional regulatory gene network [1, 2] for specification of endomesderm

in the sea urchin embryo. Based on many years of work, the network incorporates perturbation data; computational and experimental cis-regulatory data; the results of rescue experiments; and all the underlying information from experimental embryology.

Before commencing the perturbation experiments, a good deal of work was spent disovering and characterizing new genes [3, 4]. According to Davidson and coworkers most of the transcriptional regulatory genes that are involved specifically in endomesoderm specification up to 24 hours are probably known. After discovery, many genes were sequenced and direct cis-regulatory analysis carried out. In some cases cis-regulatory regions for genes were confirmed experimentally, while in other cases, the regions were inferred by comparison with the same region of the genome of a close relative of the sea urchin under study.

2.1 Perturbation Data

A variety of perturbation experiments were made by Davidson and coworkers to uncover the relationship between various genes in the network. We divide these into two categories, the first being perturbations of factors that act on the gene network from outside, the second being perturbations of factors that are entirely within the gene network. This distinction, of course, depends on the progress of experiments — future experiments will expand the network and encompass current 'external 'perturbations. However, separating the perturbations will be useful when comparing our model to Davidsons. Since, using the simple models in this project, we are not likely to be able to correctly generate a model of the external factors.

The first category is a set of perturbations consisting of mRNA injections that influence transcription factors that are known to be important in the development of the endomesoderm, but are not themselves expression products of genes in the network being studied. Much previous work has uncovered that nuclearisation of β-catenin is required in a certain spatially localized ring of cells in order that the sea-urchin embryo develops correctly, and that a cadherin mRNA (cad moe) injection binds free β-catenin in the cytoplasm and prevents nuclearisation. This type of knock out effects many genes downstream. In a similar manner, an injection of dominant negative Notch overexpression (dnn moe) acts to block the Notch receptor. This receptor is part of an important intercellular signaling mechanism that causes particular type of cells to form, and blocking it's action causes a large ripple

	Perturbation Type	Gene
External	MOE	cad dnn
Internal	MASO	alx1 bra dri ets1 eve foxa foxb gatac gatae gcm gsc hnf6 n pmar1 soxb1 hox11/13b krl tbr
Internal	EN	dri elk krox otx pmar1
Internal	MOE	pmar1 soxb1

Table 1: Perturbations used

effect.

The second set of experiments is a much larger set of perturbations, each designed to knock out or enhance the effect of a particular gene in the network. Within this set there are three techniques used to develop knock outs or enhancers:

- Antisense morpholino oligonucleotides (MASO) are molecules designed to bind to the mRNA produced by a particular gene. This shuts down the the translation of mRNA to transcription factor and acts to knock out the expression of a particular gene.

- Engrailed repressor domain fusions (EN) are mRNA molecules that are a fusion of the mRNA of a particular gene and a piece of mRNA the encodes a powerful repressor called Engrailed. When these molecules are translated, the transcription factor will likely bind to any site that the original genes transcription would bind to, but it's action will be to silence the gene it has bound to.

- Use of an mRNA of a gene directly. This causes an over expression of a particular gene (MOE).

For example, if the gene krox normally upregulates gene otx, then an injection of engrailed fusion of krox (krox EN) causes a transcription factor to be produced that will bind to otx and prevent any expression of otx. The full set of perturbations is shown in the table 2.1.

The effects of these perturbations were monitored in two ways. The first was whole-mount in situ hybridization, where mRNA expressed by a

gene of interest is fused with a special piece of sequence which can be made visible after some further processing. This technique allows information to be gathered about the time and location of gene expression, but is not very accurate.

The second was using qualitative polmerase chain reaction (QPCR). In this technique the expressed mRNA is converted to DNA via a reverse transcriptase, and with appropriate PCR primers the number of copies is exponentially increased via the usual PCR procedure. By running the PCR until the DNA is just detectable, the original amount of mRNA can be calculated based on how many PCR iterations were required. This process is very accurate, but does not afford any spatial information regarding expression to be gathered.

To be recorded a change of greater than 3-fold was required, i.e. less than 33% of the control level of transcripts or more than a 300% result from the perturbation. Usually more than one experiment was performed, using both the same embryo culture and different ones. The results of all the QPCR experiments are available at `http://www.its.caltech.edu/~mirsky/qpcr.htm` and are updated as new experiments are performed.

A schematic of the network in [1] is shown in Figure 2.1. Additional view of the network and of our intermediate results are available at [18, 17].

2.2 Direct and Indirect Effects

By itself, perturbation data cannot distinguish between direct or indirect effects on genes. A knock out of a gene that encodes a transcriptional activator may decrease expression of both the target genes it has an immediate effect on, and also on genes donwstream of this target gene. A direct effect occurs when a perturbation changes the expression or function of a transcription factor that changes the expression of another gene which has a target site for that factor. Obviously, detailed cis-regulatory analysis of the gene will resolve whether or not there is a target site, but this analysis was not available for many genes. In this case, rescue experiments were used. A rescue experiment consists of the attempted rescue (reversal) of a pertubation effect by injection of an appropriate amount of mRNA encoding a different perturbation. For example, the perturbation otx En depresses the expression of the eve gene in the 24-27 hour time period. However, introduction of krox MOE returns the expression of eve to normal levels, so it is likely that otx acts on eve indirectly through krox.

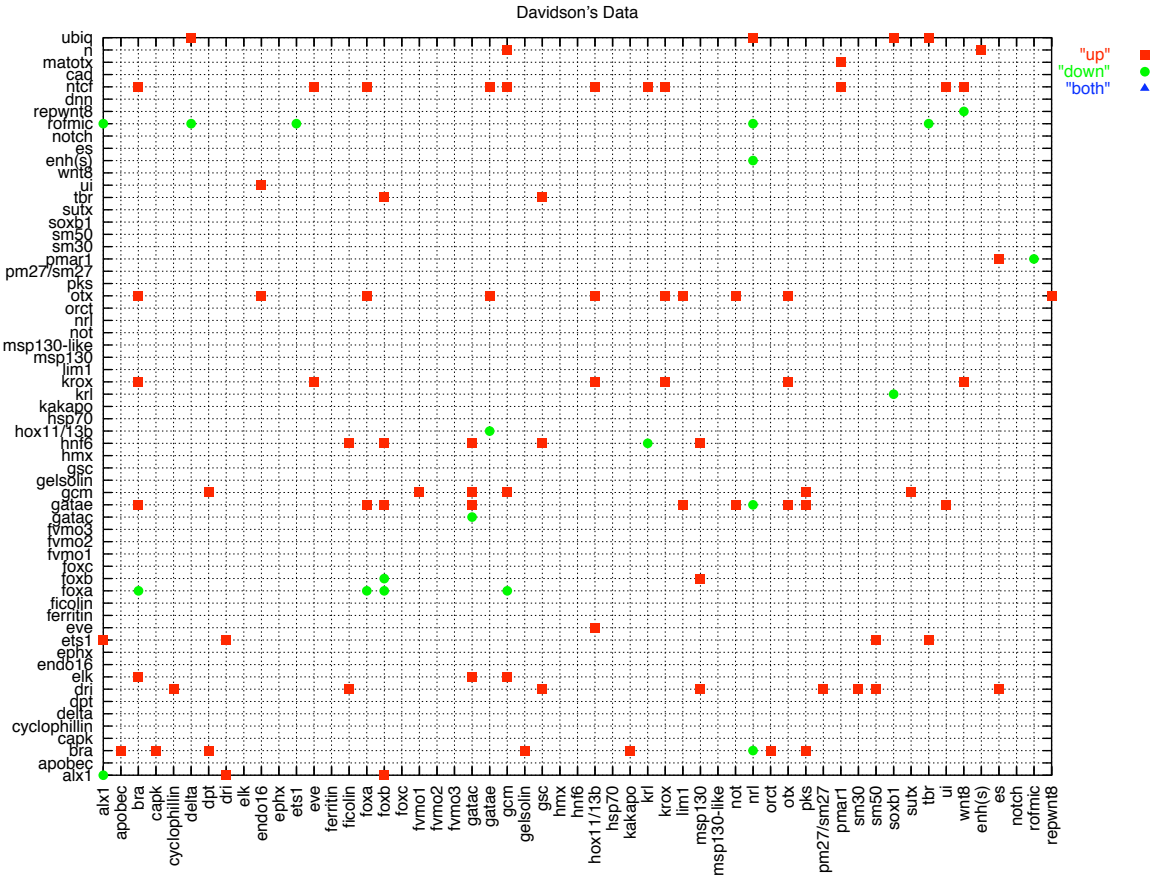

Figure 2: Davidson's Gene Network

Direct experimental and computational cis-regulatory analysis of genes is recorded somewhat cryptically in Table 3 of [2], using the acronyms ECRA to mean experimentally verified connection, and CCRA to mean a computationally verified connection. The details of rescue experiments are scattered in footnotes to Table 3 and Appendix I of [2]. They are not recorded as direct QPCR measurements.

3 Modeling the Sea Urchin cis-Regulatory Network

We chose to model the cis-Regulatory network using a directed graph (digraph) where the nodes in the graph are genes, and the arcs between the nodes are perturbations. The arcs carry a sign to indicate whether the perturbation is an enhancer, $+1$, or a repressor, -1.

3.1 Working with the QPCR Data

The original format of the QPCR data file was:

`GENE` *gene name*
`PERTURB` *perturbation name*
time spec. {*value*, `NS`} *separator* {*value*, `NS`} *separator* ...
...

where *time spec.* indicates the time period during which the experiment was run, for example, 12-16h; *value* is a real number showing the increase or decrease of expression of the specified gene in response to the sepcified perturbation, `NS` is used to indicate no signal; *separator* may be one of ',' used to indicate QPCR experiments using the same batch of cDNA, or '/'used to separate different embryo cultures.

To this file we added two more annotations. The first was to facilitate a direct comparison of the network shown in Davidson's paper with our own networks. The annotation consisted of the keyword `NET` followed by a ± 1 indicating an up or down regulation respectively. The second annotation was made to account for the various rescue experiments and additional information that had been accumulated to indicate that an interaction was actually indirect. This annotation consisted of the keyword `NET` followed by 0.

3.2 Software and Algorithms

Two parallel software suites were written to manipulate this data. The first consisted of a set of Perl scripts to parse the QPCR data file and generate connectivity matrices, to compare connectivity matrices, and to generate input data for Gnuplot so a network could be visualized using a scatter plot; a Fortran 90 program to carry out other matrix manipulations such as transitive reduction. For more details, see [18] The second was a Java framework to parse the data, generate connectivity matrices perform other matrix manipulations, and to implement a motif finding algorithm. For more details, see [17]

One of the most obvious experiments to try on the QPCR data is a transitive reduction.

The *transitive closure* of a directed graph (digraph) G, denoted by G^+, is obtained by adding an arc (i, j) if there is a path from i to j in G [8]. The transitive closure of a graph indicates the reachability of all pairs of vertices in a graph. Two graphs G and G' are said to be *transitively equivalent* if they have the same transitive closure. The *transitive reduction* of G, denoted by G^-, is obtained by removing as many arcs as possible, such that the resulting graph is transitively equivalent to G.

For an arbitrary digraph, finding its transitive reduction in NP-complete; however, if the digraph is acyclic, then its transitive reduction can be determined in polynomial time. Warshall's transitive closure algorithm [15] finds a transitive closure of a matrix in $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ space.

We modify Warshall's algorithm in a simple way to produce a transitive reduction of the QPCR digraph, taking into account that: the edges in our digraph have signs, the meaning of which must be preserved; and that the matrix representing the digraph, D, is not an adjacency matrix, it is likely to have transitive edges. The first modification is to use the product of the edges when generating a new transitive edge. This ensures that, for example, when A enhances B and B enhances C, then A must enhance C. The second modification is that algorithm must record edges that are present in D but discovered to be transitive, and must record newly generated edges as transitive. The algorithm is shown as pseudocode below. After the algorithm has completed, and element of R that has a value greater than 2 corresponds to a transitive edge.

```
R := D
for k = 1 to n do
  for i = 1 to n do
    for j = 1 to n do
      if R(j,i) > 0 and R(k,i)*R(j,k) > 0 or
         R(j,i) < 0 and R(k,i)*R(j,k) < 0 then
         R(j,i) := 2*R(j,i)
      else if R(j,i) == 0 and R(k,i)*R(j,k) != 0 then
         R(j,i) := 2*R(k,i)*R(j,k)
      else if R(j,i) != 0 and R(k,i)*R(j,k) != 0 then
         record inconsistent edge
```

The final branch of the `if` clause records edges that are inconsistent with one another.

The variant of the Warshall algorithm implemented has limitations when the input contains cycles, and the perturbation data actually contains a small number of cycles. For example, otx increases expression of krox and krox increases the expression of bra, however krox increases the expression of otx which also increases the expression of bra. Under these circumstances, edges otx $\rightarrow$ krox and krox $\rightarrow$ otx are both labelled transitive and both are deleted. Because of this behaviour, some nodes have no inputs after running the algorithm. In reality, at least one of these edges needs to be kept. We implemented a simple fix for this problem. If the node has no inputs after transitive reduction, an edge is kept for a value of R equal to 2. These correspond to edges between nodes that are connected via one other path. In the example above both edges will be kept.

4 Motifs in genetic regulatory networks

It is by now well-established that many artifical and naturally-ocurring networks have striking and recurrent topological properties. For instance, networks as disparate as the World Wide Web, the human social network, and power grids posses a degree-distribution among their nodes that varies as an inverse power. The so-called 'power-law' networks possess nodes whose outdegree goes as $1/k^{\alpha}$, where k is the node rank, and α is some scaling parameter typically ranging from 2 to 4. Such a degree distribution also leads to other interesting macroscopic properties among such networks, such as strikingly short search times and small average diameter.

Many networks also possess certain striking microscopic properties, specifically, the existence of over-represented motifs. A motif is a combination of three, four, or five (or arbitrarily many) node subgraphs that appear more often than random chance would allow. In the case of transcription networks, such motifs are thought to represent the biochemical constraints of how transcription factors bind to each other and to DNA, and thus initiate transcription. Hence, finding such motifs, if they exist, would elucidate not only how transcription factors collaborate to control transcription, but would also facilitate building the large-scale network describing the interaction of transcription factors among many genes (thousands in the case of the whole genome).

Recently, the Alon [10] group enumerated all three and four-node subgraphs appearing in the transcription network of a well-studied organism *S. cerevisiae*, or brewer's yeast. Then, they compared the rate of appearance of a given motif to the same motif in random networks of similar degree distribution. Motifs appearing in the transcription network more than their random counterparts were flagged as significant. In the case of *S. cerevisiae*, there were three such motifs: the feed-forward loop, the single-input module, and the dense overlapping region (DOR).

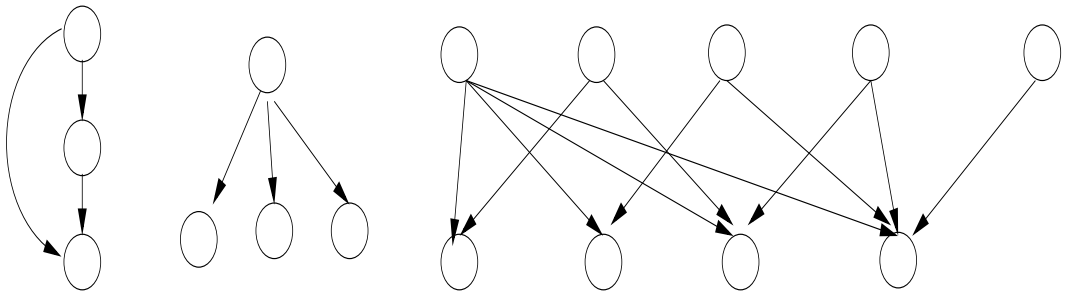

Figure 3: Recurring motifs in transcription networks. Feed-forward loop (*left*), single-input module(*center*), and dense-overlapping region (*right*). These latter two are directional bipartite graphs.

The motifs were found through exhaustive and computationally slow search of three and four node subgraphs, as well as a clustering approach that located areas of dense connectivity, and hence high probability of finding a DOR. However, such a search may well be slow in a transcription network larger than the ~ 500 nodes of *E. coli*. Additionally, computer science theorists have delighted in finding polynomial-time algorithms for the

motif-finding problem (or more correctly, the subgraph isomorphism problem), which is known to be NP-complete[6]. To that end, two of these authors formulated just such algorithms for the three motifs cited above, and implemented them on the *E. coli* network used in [10]. Since *this* paper focuses only on sea urchin and the Davidson work, we only summarize these results here.

The feed-forward loop can easily be found via the Floyd-Warshall algorithm [9] in $O(n^3)$ time. The single input module and dense-overlapping region can be seen to both be examples of one type of subgraph, what we call a *directional bipartite* graph. Unlike the conventional bipartite graph well-known to graph theorists, our definition includes the additional constraint that the graph be directed, and furthermore all edges traversing the partition be pointing in the same direction. Thus a DB graph is a directed graph that can be divided into two sets of nodes V_1 and V_2 such that $(u, v) \in E$ implies that $u \in V_1$ and $v \in V_2$, and that $u \notin V_2$, $v \notin V_1$. This is exactly the dense overlapping region sketchily defined by Alon[10] and shown in Fig. 3.

The details and full implementation of the algorithm can be found here[16]. Very generally, the algorithm crawls the network in a depth-first search-like manner, growing a series of directional bipartite graphs as it trawls, and only outputting the set of maximal such subgraphs. The running time is $O(n)$ for arbitrary motif size, a dramatic improvement over the exponential cost of the Alon group's exhaustive search. While additional heuristics were added to assure accuracy in the face of possible strange topologies, the authors are confident of the algorithm's fundamental soundness.

A graph of the *E.coli* transcription network with discovered motifs is available[16], and seems to concur broadly with the motifs found in [10]. As indicated in that work, the motifs correlate strongly with gene function, with a given dense-overlapping region comprising most or all of the genes involved in a cell's basic functions, such as osmotic stress reaction or carbon utilization.

It is possible, if not likely, that these motifs are also prevalent in higher organisms, like sea urchin. If this is true, then finding such motifs in a network like the one considered in this paper would make the task of network inference considerably easier. In the edge-removal process described in this paper, edges belonging to a known motif might be spared, as such edges are more likely to be correct. In addition, examining nodes as grouped in a motif might allow us to extract information that, currently, resides only the experimentalist's head. In other words since, as indicated by [10], topology

probably reflects function, then knowing such microscopic topological detail may elucidate function. As we have discovered in this work, armed with nothing other than the Davidson perturbation data and rudimentary knowledge of sea urchin development, inferring the transcription network is difficult indeed. If, however, we can clearly divide the relevant genes by function, it would be one more factor to consider, in addition to time and location of expression within the embryo, in understanding gene expression.

This leaves for the moment the equally challenging question of motif discovery. If indeed there are recurrent motifs in transcription networks, then they are likely to be more complex in multicellular eukaryotes like sea urchin, than those found in single-celled prokaryotes like *E. coli*. Alternatively, while the basic building blocks could be limited to the simple motif vocabulary found in *E.coli*, there may well be a higher-order structure that organizes such blocks into a complex network. Hence, the obvious next step in the study of transcription networks is to probe larger and more complex networks, in larger and more complex organisms, where such phenomenon would be apparent.

5 Analysis

The procedures described above iteratively progresses from all the possible transcription factor networks indicated in the QPCR data toward the network presented in Davidson's paper [2]. There are 55 different genes in the original QPRC data, which can be viewed as nodes on the network. Six more nodes are added to the network in the paper to accommodate maternal influences and other affects. In a directed network of 61 nodes, there are 7,442 possible edges.

Davidson's actual data tables indicates a total of 238 edges, yet the final transcription factor network in the paper, which we will call the 'paper network', only has 104 edges defined. There are multiple ways the edges are reduced as described below. Using the footnote information in the Davidson paper removes 66 edges.

Performing transitive reduction on the data eliminates more edges. There are two ways to do transitive reduction. The first is to do the transitive reduction once for all the data. The other is to do it at each of the four time periods, and merge the results keeping all non-transitive edges at each time step. These two approaches differ in their impact. For example, using

Data	Operation	Number of Edges
QPCR		238
QPCR with footnotes		172
QPCR	TR	43
QPCR with footnotes	TR	49
QPCR	MTR	159
QPCR with footnotes	MTR	130

Table 2: Number of Edges in Network as a Function of Data and Algorithm

the original QPCR data, which starts with 238 edges the one time reduction eliminates 195 edges, far too many. Using transitive reduction on the data per each time period eliminates 79 edges and leaves a total of 159. Similar results are shown for other combinations. Essentially, doing transitive reduction once eliminates too many edges, but for transitive reduction taken at each time step the number of edges starts to approach the number of edges in the paper network close to the final diagram.

Table 2 summarizes the number of edges for each type of operation. The two basic data sources are the original, undisturbed QPCR data, and the QPCR data with the adjustments indicated in the Davidson paper. The two operations are transitive reduction done on edges from all time periods simultaneously (TR), and transitive reduction performed at each time of the four time periods (MTR). This results in 6 combinations.

There is a significant difference in doing transitive reduction once or multiple times - once per time interval. If the reduction is done just once, the number of edges is well under what is reported in the paper network. Applying transitive reduction at each phase of development, yields a number of edges higher than the paper network.

It is interesting to look at the number of edges in the original data at each of the four time periods. These are shown in Table 3.

Agreement on the total number of edges is not sufficient, the edges have to be the correct edges. Table 4 shows the comparison of the number of edges in each derived Transcription Factor network and that of the paper network. For each combination of operations, there are three categories of edges:

- Edges that exist in both derived network and the paper network
- Edges that exist in the derived network but do not exist in the paper

Time Period (hours)	QPCR	QPCR with footnotes
12-16	58	45
17-21	116	189
23-28	173	126
30-36	16	16

Table 3: Number of Edges as a function of Time

Comparing Paper with:	QPCR	QPCR TR	QPCR MTR	QPCR with foot-notes	QPCR with foot-notes TR	QPCR with foot-notes MTR
In Common	69	17	60	69	23	61
Edges in data but not in paper	166	26	97	100	26	67
Edges in paper but not data	32	87	42	32	81	41
Edges that disagree	3	0	2	3	0	2
Edges displayed	235	43	157	169	49	128
Edges in the Paper Network	104	104	104	104	104	104

Table 4: Comparison of edge counts and types

network.

- Edges that are in paper network, and not in the derived network.

Also in some cases, there is disagreement for a small number of edges that exist in both the derived network and the paper network. This is where the edges exists in both, but are opposite in function (e.g. an activator rather than a repressors, of vis a vis).

Some interesting points can be made about Table 4. First, the transformation with the closest match to the paper network is QPCR data, adjusted with the footnote comments in the paper footnotes and then with transitive reductions performed at each phase. This agrees to within 23% on the number of total edges. Despite that, the number of edges in the derived data and the paper network, are not consistent. There are 41 edges in the paper but not in the derived graph.

Comparing Paper with:	QPCR	QPCR TR	QPCR MTR	QPCR with foot-notes	QPCR with foot-notes TR	QPCR with foot-notes MTR
In Common	69	17	60	69	23	61
Edges in data but not in paper	121	22	64	70	19	41
Edges in paper but not data	4	59	14	4	53	13
Edges that disagree	3	0	2	3	0	2
Edges displayed	190	39	124	139	42	102

Table 5: Comparison of edge counts and types

Another way to evaluate the degree of agreement is to reduce the QPCR data to only the genes and perturbations in the paper network. Then the same transformations are applied to the data. In this case, the data shows improved agreement when all information is used. Indeed, as Table 5 shows, there is now 98% agreement between the number of edges in the derived network and the network in the paper. However, these edges are still inconsistent, as is also shown, with 13% of the edges indicated in the paper network not present in the derived network.

Figures showing the results are available in a separate PDF file, or at [17].

6 Conclusions

Several conclusions are clear from our analysis. First, it is not possible to automatically or even manually derive a network from the data presented in the QCPR tables that is consistent with the paper network. There are two aspects to this conclusion. First, we are unable to derive a network that has approximately the same number of edges as the paper network. The second aspect is whether the data can be consistently derived - so that even if the total edges do agree, there is consistency in whether edges are overly specified or under specified. The closest agreement, as one would expect is when all the information is taken into account for the derived graph that adjusts the QPCR with footnoted information and with Transitive Reduction at each Time phase. Yet while there is agreement in total edges, it only had 48% of

the edges in agreement between the derived edges and the paper.

There may be several reasons for this discrepancy. First, there may be errors in our methods we have not detected, or that the algorithms used by Davidson had addition function we did not include. This is mitigated by the fact we approached the reductions in two algorithmic ways and the annotations from the QPCR data with footnote information is straightforward. The results of the two independent algorithms agree to less than 1% difference. Another possibility is that there is addition information Davidson had that is not included in the paper, which led to the transcription factor network in the paper. This information was not found by us in other publications.

When the data set was reduced to just the genes and perturbations in the paper network, there was very close agreement in the number of edges in the derived network, but still 13% inconsistency in which edges were actually found.

7 Acknowledgments

This work was supported by the Director, Office of Science, Office of Advanced Scientific Computing Research, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.

References

[1] E.H. Davidson *et al*, A Genomic Regulatory Network for Development, Science **295**, 1669, (2002)

[2] E.H. Davidson, *et. al.* A Provisional Regulatory Gene Networks for Specification of Endomesoderm in the Sea Urchin Embryo, Developmental Biology, 246, 162-190 (2002).

[3] P. Oliveri *et al*, A Regulatory Gene Network That Directs Micromere Specification in the Sea Urchin Embryo, Dev. Biol. **246**, 209, (2002)

[4] A. Ransick *et al*, New Early Zygotic Regulators Expressed in Endomesoderm of Sea Urchin Embryos Discovered by Differential Array Hybridization, Dev. Biol. **246**, 132, (2002)

[5] W. Wang, J. Michael Cherry, Hao Li A Systematic Approach to Reconstructing Transcription Networks in *Saccharomyces cerevisiae*, *Proc. Natl. Acad. Sci.* vol 99, 16893 (2002).

[6] B.T. Messmer, and H. Bunke, Subgraph Isomorphism in Polynomial Time, Technical Report IAM 95-003, University of Bern, Institute of Computer Science and Applied Mathematics, Bern, Switzerland (2002).

[7] T.I. Lee, N. J. Rinaldi, F. Robert, D. T. Odom, Z. Bar-Joseph, G. K. Gerber, ... D. K. Gifford and R. A. Young Transcriptional Regulatory Networks in *Saccharomyces cerevisiae Science*, **798**, 799-804, 2002.

[8] C. Papadimitriou, K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity, Dover Publications, 1982.

[9] T. Cormen, C. Leiserson, R. Rivest and C. Stein, Introduction to Algorithms, MIT Press, 2001.

[10] S. Shen-Orr, R Milo, S Mangan, U Alon, Network motifs in the transcriptional regulation network of Escherichia coli, Nature Genetics, **31**:64-68 (2002).

[11] T. Ideker, Thorsson and Karp, Discovery of Regularity Interactions throughy Perturbation: Inference and Eperimental Design.

[12] U. Ohler, G. Liao, H. Niemann and G. M. Rubin *Computational analysis of core promoters in the Drosophila genome*, Genome Biol. **12**, 2002.

[13] T. L. Bailey and Charles Elkan, Fitting a mixture model by expectation maximization to discover motifs in biopolymers, *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, 28-36, AAAI Press, Menlo Park, California, 1994.

[14] J. W. Fickett and Artemis G. Hatzigeorgiou, Genome Research **7**, 861-878, September 1997.

[15] A.V. Aho, J.D. Ullman and J.E. Hopcroft, The Design and Analysis of Computer Algorithms. Addison-Wesley Publishing Company, San Francisco, CA, 1976.

[16] http://cryptologia.com/GeneReg

[17] http://cryptologia.com/cs294

[18] http://www.nersc.gov/~jcarter/uni