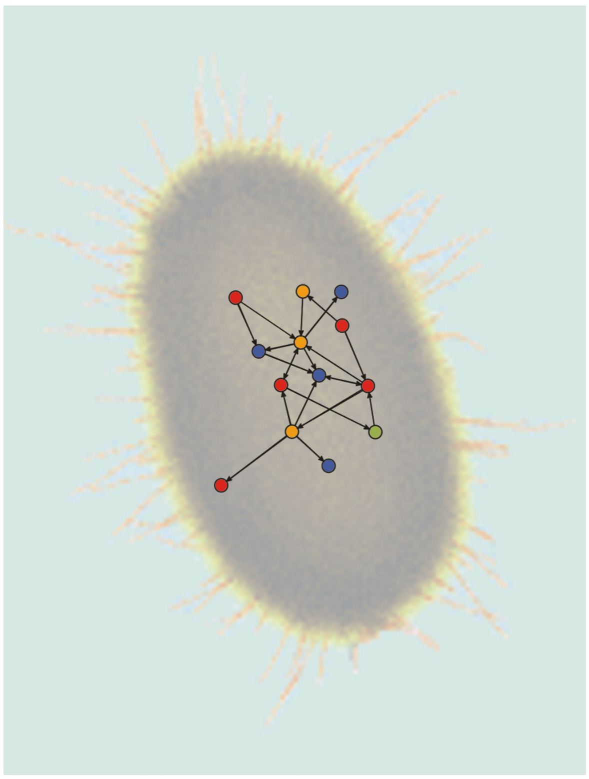
Reconstructing Gene Function and Gene Regulatory Networks in Prokaryotes

by

Dion Whitehead



A dissertation submitted in partial fulfullment of the requirements for the degree of

Doctor of Natural Philosophy (Doctor rerum naturae)

at the

University of Bielefeld (Germany)
2005

Summary

The ability of a bacterial cell to respond to a complex environment lies with the network of genetic interactions that process information regarding the state of the cell and its local environment. The object of this study has been to better understand bacterial genetic regulatory networks, which conceptually consist of nodes (the genes) and edges (the interactions between the genes). Two problems are addressed, one focusing on the nodes in the network, the other focusing on the edges (see Figure 1.1).

The problem concerning the nodes in the network is that a large fraction of genes from bacteria have an unknown function. Even the most studied microbe, *E. coli*, still has around 30% of its genes uncharacterised. Using phenotype data and genomic data, a method for gene-function prediction is improved and implemented in a tool called GeneTrawler. The gene-function prediction tool is validated against a number of well studied phenotypes, and successfully predicts the majority of genes known to be involved in each phenotype when the phenotype is well defined. In addition, it can suggest genes that are dependent on a phenotype, but not directly involved.

The second problem addressed in this thesis is reconstructing gene networks from DNA microarray data. DNA microarrays can measure the expression levels of thousands of genes simultaneously. Elucidating the topology of biochemical or genetic regulatory networks is a common problem in molecular biology (Arkin *et al.*, 1997), but piecing together gene regulatory networks with standard molecular methods not only requires significant time and effort, but may also miss the global or modular structure of the network. In addition, different stimuli can substantially change the ‘wiring’ of a network (Luscombe *et al.*, 2004), meaning that understanding how a network functions in time requires a global analysis of its dynamics combined with a detailed fine-grained approach.

There has been much interest in the possibility of reconstructing genetic regulatory interactions from microarray data. We investigate the limitations of such reconstruction procedures, in terms of what kinds of networks can (and cannot) be reconstructed. A method of reconstruction is investigated that uses evolutionary computation for ’evolving’ a population of networks to resemble a target network. The results demonstrate that the topology of gene regulatory network structures can be reconstructed from time series data if the network displays dynamics of a particular category where negative feedback loops determine the dynamical behaviour of the network.

Declaration

This dissertation is the result of my own work and includes nothing that is the outcome of work done in collaboration, except where explicitly noted in the text.

Structure

This thesis addresses two problems. Chapter 1 provides the background literature and concepts for both problem areas. Prokaryotes are introduced and placed in context with the two other domains of life, followed by a discussion on the impact of genome sequencing technology in Section 1.4. Some important concepts regarding genome evolution are outlined in Section 1.3 which are a basis for the gene-function prediction methods in Section 1.6. Methods of modeling gene regulatory networks are discussed in Section 1.8, followed by a discussion of previous attempts at reconstructing gene regulatory networks from microarray data in Section 1.9. A discussion about evolutionary computation concludes in Section 1.10.

The next two chapters (2 and 3) detail work on GeneTrawler, a tool for gene-function prediction. GeneTrawler uses phenotype and orthologue information to predict gene function. The tool was implemented as a web tool for working biologists and the instruction manual is located in the Appendix. Chapter 2 introduces the algorithm and described the implementation. Chapter 3 discusses the results and outlines future work.

Chapter 4 introduces the second problem addressed in this thesis. The motivation for reconstructing gene regulatory networks is discussed and the reconstruction method is described. The results determined by applying this method to random networks are presented and discussed in Chapter 5.

Chapter 6 concludes the thesis with a general discussion of the results of both Genetrawler and the gene regulatory network reconstruction procedure.

Acknowledgements

I would firstly like to thank Prof. Andreas Dress, for giving me the opportunity to study at the University of Bielefeld, and for the advice and support during this thesis. I would also like to thank Prof. Ralf Hofestädt and his group for additional support and advice.

Special thanks to Jacob Köhler and Stephan Philippi, with whom many discussions, both scientific and not, provided a valuable perspective during my studies in Bielefeld. The second part of this thesis was done in collaboration with Andre Skusa and Paul Kennedy, and I thank them both for working with me on such an interesting topic. I have learnt a lot from them both and hope to continue to do so in the future.

I greatly appreciate the support team at the Technischen Fakultät Bielefeld, especially Alexander Sczyrba for his help with a complex database query and Jan Krueger for his help in setting up the GeneTrawler web service. Also the Cebitec support team for their quick and excellent help on numerous technical questions.

I also thank Klaus Prank and the senior staff of the NRW International Graduate School in Bioinformatics and Genome Research, for making this oppurtunity possible and for their committment to bioinformatics teaching and research, and all my friends and collegues from the graduate school for making my time in Bielefeld both scientifically and socially rewarding.

Special thanks goes to my girlfriend, Maya, for her support and neverending ability to put me in a good mood despite computer-related frustrations.

My gratitude lastly goes to my parents whose love and support, and decision to move to Ireland for a few years, made this possible.

This work was funded by the Nordrhein-Westfalen Government (Germany).

Table of Contents

List of Figures

List of Abbreviations and Keywords

ARN	Artificial (gene) regulatory network
Bacteria	A less well defined term, in this work a synonym for eubacteria.
DE	Differential equation
EP	Evolutionary programming
ES	Evolutionary strategies
EST	Expressed sequence tag. A fragment of transcribed RNA.
Eubacteria	A domain of life (the others are archaea and eukaryotes).
FFL	Feed forward loop, a specific 3 node network structure.
GA	Genetic algorithms
GP	Genetic programming
GRN	Gene regulatory network
GUI	Graphical User Interface
HGT	Horizontal gene transfer
Negative feedback	When a gene's products directly or indirectly reduce its expression level.
ODE	Ordinary differential equation
ORF	Open reading frame. Refers to a DNA sequence with no stop codons.
Phenotype	A specific trait or biochemical function displayed by an organism
Prokaryotes	A term that includes the two domains of Archaea and Eubacteria.
ROC	Reciever opererator characterisitc. Used to evaluate prediction methods.

Chapter 1

Background

The structure of this chapter is as follows: the first section discusses the motivation for the work presented in this thesis. The second section introduces prokaryotes, which are placed in context with the two other domains of life: archaea and eukaryotes. This is followed by a discussion on the impact of genome sequencing technology in Section 1.4. Some important concepts regarding genome evolution are outlined in Section 1.3 which are a basis for the gene-function prediction methods in Section 1.6. Section 1.7 provides definitions of graphs and networks. The following Section (1.8) discusses different formalisms and models for representing and simulating gene regulatory networks. Section 1.9 discusses recent approaches in reconstructing gene regulatory networks from microarray data, while the last section (1.10) gives an outline of evolutionary algorithms.

1.1 Motivation

> "The greatest challenge today, not just in cell biology and ecology but in all of science, is the accurate and complete description of complex systems. Scientists have broken down many kinds of systems. They think they know most of the elements and forces. The next task is to reassemble them, at least in mathematical models that capture the key properties of the entire ensembles." E. O. Wilson (1998).

As we peer into the hidden world of the microscopic, like most things studied in science, we are met with unimagined complexity. One of the first steps in the face of such complexity is to catalogue; and for most of the previous century, biology was very much a reductionist science where the effort was spent trying to obtain a parts list for some biological system. This approach is still important today as there are many cellular processes that are poorly understood. However, for many systems there is a veritable flood of data, for example genomic sequences and microarray data. The availability of detailed information about certain biological systems means that a more systems-oriented approach may shed light on complex biological processes. Often, a systematic view results in some kind of network, as a principle shared by many natural systems is their organization into a network of many interacting units: atoms in molecules, molecules in cells, nerve cells, species in an ecosystem, computers in the Internet, and interacting people in a social environment are just a few examples.

In molecular and evolutionary biology, the principal units under study are the *genes*, which not only contain the information to grow an organism from a single cell, but also contain information about the evolutionary history of the species. Since the 1950's, after the discovery of the structure of DNA (Crick & Watson, 1953), molecular biology has come long way, with sophisticated sequencing technologies routinely providing the raw material for scientific analyses involving genes.

Genes can indirectly interact with one another, usually via the proteins they encode, for example, by producing transcription factors that influence the expression rate of another gene, or by producing a protein that then produces a metabolite that influences protein activity. Thus, genes are usually discussed as components in a larger network of interactions (Figure 1.12).

This thesis addresses two problems in elucidating the entire network of genetic interactions in a particular organism. The first problem is that most genes in most organisms have an unknown function. For example, in *E. coli*, the most intensively studied microorganism, around 30% of its genes have an unknown function. Reconstructing a network is of dubious value when so much is missing. Chapters 2 and 3 address this problem with a new method of gene-function prediction that combines homology and phenotype data in a novel way. The second problem is in piecing together the network of gene and protein interactions. Microarrays have provided the first opportunity to do this in a systematic way, however it is unclear if such a technology can furnish the required data for reconstruction significant parts of the cellular interaction network. Chapters 4 and 5 introduce a method of gene regulatory network reconstruction using microarray data. Figure 1.1 shows how these two problems are part of the overall problem of elucidating the function and structure of the genetic reaction network controlling cellular systems. Each section in the rest of this chapter discusses the necessary background for the two approaches discussed.

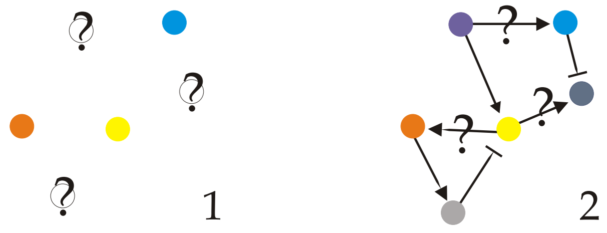


Figure 1.1: This thesis addresses two problems in piecing together a network of cellular interactions: 1) Many genes (coloured circles) have an unknown function; and 2) Microarray technology provides the first opportunity to reconstruct genetic interactions in a large-scale way. This can be be thought of as 1) reconstructing the nodes in a network, and 2) reconstructing the edges in a network.

1.2 A brief history of Prokaryotes

This thesis concerns the reconstruction of gene function and regulatory network topology in prokaryotes. The reason for this focus is that the lifestyle, selection pressure, gene content, evolution strategies, and core genetic replication machinery are substantially different to eukaryotes. Therefore, the methods for understanding core cellular processes in one domain often do not work for the other. Because of this restriction to prokaryotes, a brief history is given, followed by relevant sections concerning gene-function prediction.

Many of the core discoveries of the cellular machinery were acquired through studying microbes, mostly due to their ease of handling and manipulation. However, there are many other reasons to study microbes: they are crucial for all ecosystems from small to large scales, and are the causative agents for many diseases of humans and livestock, and play such a huge and important role in every ecosystem, acting mostly as ‘black boxes’, as so much is unknown.

The classification of bacteria has undergone much change over the last few decades, with the result of renaming and reclassifying many species. The cause of this have been the many recent advances in gene sequencing and related technologies, genomics, bioinformatics and computational biology, which have revealed important parts in the long story of life.

Originally, bacteria were thought to be a group of fungi, except the cyanobacteria which were considered to be blue-green algae. This perception changed with the discovery of the prokaryotic cell structure, which defined them until the 1970s as a distinct group, variously called Monera, Bacteria, and Prokaryota.

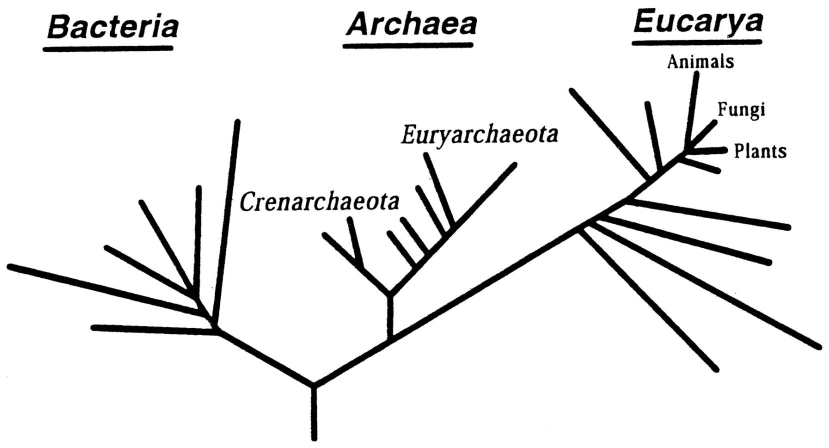
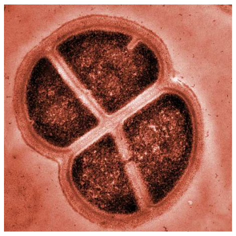


Figure 1.2: A universal phylogenetic tree inferred from comparative RNA sequences (from Woese (2000)).

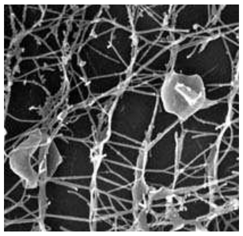
In 1977, Carl Woese shocked the biological community with the introduction of a new group of organisms, the Archaea (Woese & Fox, 1977). From studying RNA sequences, Woese concluded that prokaryotes comprise of two distinct groups, known today as Bacteria and Archaea. Biochemically, Archaea appeared as different to Bacteria as Bacteria to Eukaryotes, and were often isolated in extreme environments (see Box 1.3). These three groups, Bacteria, Archaea, and Eukaryotes, are called domains. At present, the dates and order of branching at the root of this tree is rather uncertain (Figure 1.2).

Box 1.3: Extremophiles

Many places, once assumed to be sterile, have turned out to contain thriving microorganism communities, and the resulting adaptions to extreme conditions are of great scientific and industrial interest. For example, the heat-resistant DNA copying enzyme *Taq Polymerase* which is the workhorse of molecular biology, was isolated from a bacterium living in hot springs (*Thermus aquaticus*).



Deinococcus radiodurans is the most radiation-resistant organism known, being able to survive exposure to more than 1,000 times the amount of radiation that would kill a human.
Image: dividing *D. radiodurans* cells, courtesy of Alice Dohnalkova, Uniformed Services University.



Pyrodictium abyssi, found on geothermally heated areas of the seabed. It has an optimum growth temperature of 105°C and growth maximum of 110°C.
Image: *Pyrodictium* cells, connected by a network of protein 'fibers' from Rieger *et al.* (1995), ©1997 R. Rachel.

At the other end of the spectrum are the Psychrophiles. These are organisms that prefer cold temperatures, such as *Polaromonas vacuolata* which lives in the sea ice of Antarctica and has a preferred temperature of 4C. The image to the right shows ice-worms that live in a combination of natural gas and ice, 500 metres down on the bottom of the Gulf of Mexico (Image publicly available from Fisher (1997)).

Halobacteria grow optimally in water with extremely high salt concentrations of 4 to 5 M NaCl. They produce a red pigment which colours the salt lakes shown left (Image used with permission (Armstrong, 1998), ©Wayne. P. Armstrong).

Microorganisms are even found as far as 3.5km under the earths crust (Krumholz *et al.*, 1997), in extremely acidic (pH < 2) and alkaline environments (pH > 10) (Finlay *et al.*, 1987). This huge diversity demonstrates the adaptability and hardiness of life, at the same time probably indicating limits of earth-based life.

1.3 Prokaryote genome evolution

The details of prokaryote genome evolution has important implications for gene-function prediction. The evolutionary forces acting on a bacterial genome are substantially different from those acting on a larger, multicellular eukaryotes, resulting in differences in genome size, gene regulation, and gene structure. The large differences in genome structure observed between prokaryotes and eukaryotes arise mainly because of the different evolutionary strategies taken.

1.3.1 r versus K selection

Two fundamental strategies for maximising fitness are long life or quick reproduction. These are somewhat mutually exclusive, in that an organism cannot devote the same resources for quick reproduction as developing a system that is long-lived, however they are also somewhat dependent: a minimum amount of reproduction must occur for the species to survive. This is a development-reproduction trade-off.

In mostly risky and/or hostile environments where energy sources and nutrients are not regularly available, most offspring will perish regardless of how much resources for development are invested. Thus, if many offspring are produced, at least one will hopefully survive to pass on the genetic information to the next generation. This strategy is called r selection.

In a predictable environment, those that invest the most resources in development are more likely to win the competition for survival. In other words they will be stronger, larger, taller etc. This strategy is called K selection (Roughgarden, 1977). Oak trees are an example of a K selected organism.

The terms r and K come from a mathematical model of population growth. For small populations, growth is exponential as represented by the r parameter. For large populations that are near the carrying capacity of the environment (represented by the K parameter), resources are limited, which allows only the strongest, largest, most developed, or most intelligent members of the species to survive and reproduce. r-selected populations are usually not close to the carrying capacity of their environment, and thus able to grow exponentially using an abundance of available resources. However, because of the dangers in the environment, the population is regularly decimated so that it never actually reaches the carrying capacity.

1.3.2 r versus K selection and implications for bacterial genome evolution

The selective pressure for fast generation times has several implications for bacterial genomes. Firstly the size: a larger genome requires more resources for division than a smaller genome. Thus individuals with smaller genomes will be able to divide more rapidly than individuals with larger genomes competing for the same resources. This means that genes that are not under positive selective pressure for long enough time periods will eventually be lost. The larger the selection pressure for a smaller genome, the quicker the rate of gene loss.

Contrast this with genomes from most multicellular eukaryotes. There is no obvious selection pressure against large genomes and, as a result, genomes can be very large, e.g. 670 billion base pairs for *Amoeba dubia*, compared with 3 billion for *Homo sapiens* (Gregory, 2001) and 4.5 million for *E. coli*.

Quick generation times not only affect genome size, they also affect the mutation rate (Martin & Palumbi, 1993). Errors introduced through copying DNA are a major source of mutations, thus a quicker generation time means a higher mutational load. This results in an additional source of gene loss for r-selected organisms. Although one may argue that this should favour better DNA repair mechanisms, in an r selected environment where few individuals of any population survive, investing resources in better DNA repair mechanisms does not necessarily pay off unless DNA damage is the major cause of death (see Box 1.3: *Deinococcus radiodurans*).

Kunin & Ouzounis (2003) analysed the relative contributions of gene loss, gene genesis, and horizontal gene transfer (HGT) in prokaryote evolution. They concluded that gene loss is the most important factor in shaping prokaryote gene content, being three times more frequent than HGT. This high rate of gene loss is compensated by a high rate of gene genesis via e.g. gene duplications.

1.3.3 Informational/operational genes

In Rivera *et al.* (1998), the genomes from members of the three domains[1] were compared, and it was attempted to determine to which other domain the genes in each genome were most related. The genomes included two prokaryotes (*E. coli* and *Synechocystis*), an archaea (*M. jannaschii*) and a eukaryote (*S. cerevisiae*). They found that prokaryotic genomes contain two classes of genes: *informational* genes which include GTPases, vacuolar ATPase homologues, and most tRNA synthetases; and *operational* genes coding for e.g. amino acid synthesis, the biosynthesis of cofactors, the cell envelope, energy metabolism, and intermediary metabolism. The informational genes appear to have much less functional divergence than the operational genes, implying that operational genes undergo functional divergence more often. In eukaryotes, the informational genes are most closely related to the archaea, *M. jannaschii*, whereas the majority of operational genes are most closely related to those of *E. coli*.

In further work, the authors formulated a hypothesis arguing that informational genes are much less frequently horizontally transferred than operation genes (Jain *et al.*, 1999). The reasoning is that most informational genes have interactions with many other genes, and these interactions are often critical to the functioning of the DNA, RNA, and translation apparatus. If an informational gene was transferred into another genome, the chances are very small that it would be able to integrate into the regulatory network, as many interactions must be similar to those in the original host organism. On the other hand, operational genes are usually under the control of one or a few regulatory elements. This means it is more likely for operational genes to be integrated into the cellular metabolism, compared to informational genes, as there are fewer regulatory regions that must be shuffled in front of the new genes(s) (see Figure 1.4).

[1]Archaea, Bacteria, and Eukaryotes.

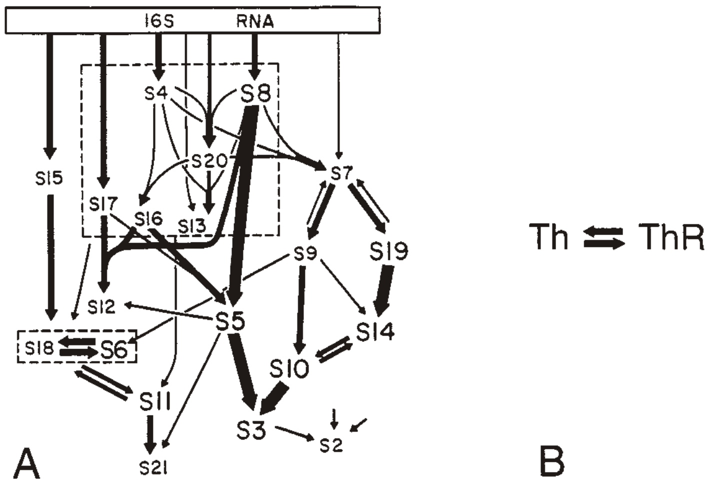


Figure 1.4: Examples of the complexity of gene product interactions in informational (A) and operational genes (B). The assembly map of the *Escherichia*, small ribosomal subunit is shown in A as an illustration of the high complexity that is frequently present in the translational apparatus. The thioredoxin (Th) and thioredoxin reductase (ThR) complex is shown in B as an example of the reduced complexity present in some operational genes (figure from Jain *et al.* (1999)).

1.4 The pace of genome acquisition

In 1995, the first complete genome of a free-living organism, *Haemophilus influenzae*, was sequenced (Fleischmann *et al.*, 1995). This was soon followed by the first multicellular organism (*Caenorhabditis elegans*) in 1998, and the rough draft of the human genome was announced in June 2000. The rate of acquisition of gene sequences has accelerated; currently there are over 200 complete genomes including several multicellular eukaryotes in the databases, and many more in the pipeline (Figure 1.5). This exponential increase in sequence databases fuels the exponential growth of other biological databases such as structure databases, presenting interesting challenges in database searching and integration (Davidson *et al.*, 1995).

Acquiring the sequence of an organism is just the beginning in terms of gaining new biological knowledge. The genome ultimately controls and directs the development and workings of an organism by regulating which genes are transcribed, at what time, and to which sub-cellular location. It is currently beyond our capabilities to build a relatively complete model of the organism starting with only the genome. Despite this, the genome is the first necessary building block in this endeavour.

Many problems remain in these first steps–in prokaryotes more than one third of the genes in most genomes have an unknown or hypothetical function (see Chapter 1). In eukaryotes the situation is still more complicated, as even identifying genes is problematic.

Even though many genes are either not properly identified or have an unknown function, the comparison of whole genomes opened the door for answering many questions, for example, the question of gene amelioration in isolated species (Tamas *et al.*, 2002). Cross comparisons of the metabolic pathways between species reveals striking similarities in the global structure (Jeong

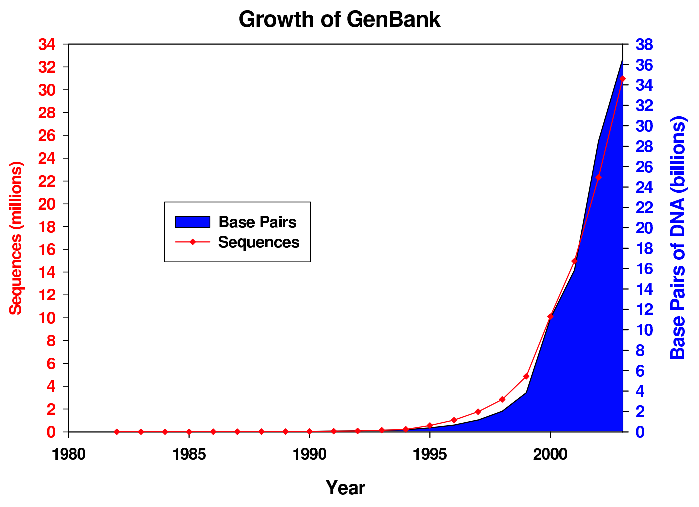


Figure 1.5: The number of sequences deposited in GenBank is growing at an exponential rate (Benson *et al.*, 2004).

et al., 2000), not only of metabolic networks compared with each other, but also metabolic networks compared with other networks such as power grids and the Internet (Albert *et al.*, 2000; Milo *et al.*, 2004).

1.5 Finding genes in a genome

Before discussing gene-function prediction methods, an overview of gene finding methods is given, as this is a crucial step that comes before gene-function prediction.

Given a raw genome sequence, the first step is to identify all the promoters, coding regions, and non-coding RNAs (for a review, see Mathe *et al.* (2002)). Most genes encode proteins, which are the biochemical machinery that actually do most of the biochemical work. Finding which stretch of DNA encodes for what protein is the next step after sequencing a genome. This is not a trivial task, especially in eukaryotes which can possess long regions of DNA that must be spliced out of the gene before it is translated (introns). Bacterial genomes do not usually contain introns, obeying the old dogma of molecular biology: one gene $\rightarrow$ one protein. To compare, about 85% of the *Haemophilus influenzae* genome is composed of genes (Fleischmann *et al.*, 1995), but only about about 2% of the human genome is composed of protein-coding sequences (exons) of genes (Pennisi, 2001).

1.5.1 Prokaryote gene identification

Compared to eukaryotes, gene finding in prokaryotes is less complex. Prokaryote genes tend to cluster in operons, do not usually have introns, and have relatively consistent gene begin and end signals. For bacterial genomes, a first step in finding all genes is to look for long regions uninterrupted by a stop codon, called ORFs (Open Reading Frames), as these regions have a higher likelihood of being genes. Although this can be done manually, an automated approach is normally applied. More sophisticated approaches are needed for automated *ab initio* gene prediction, such as hidden Markov models, which have been applied extensively to eukaryotic (Burge & Karlin, 1997), prokaryotic and archaeal genomes (Salzberg *et al.*, 1998; Lukashin & Borodovsky, 1998). As bacterial genomes usually lack introns, bacterial gene prediction is not as problematic as eukaryotic gene prediction.

1.5.2 Eukaryote gene identification

Although this thesis concerns gene regulatory networks in prokaryotes, often a clearer perspective is obtained by comparing the differences between prokaryotes and eukaryotes. Finding genes in eukaryotes is challenged by the large amounts of introns and spaces between genes, which do not exist so much in prokaryotes. There are four main methods for finding genes in eukaryotes:

cDNA library Complementary DNA (cDNA) is single-stranded DNA synthesized from a mature mRNA template. By indiscriminately amplifying all mRNAs in a cell, a cDNA library is constructed. Sequencing the entire library provides the mRNA of all the genes in the cell currently being expressed. By aligning the cDNA with the genomic sequence, a coding region can be identified. The advantages of using cDNA libraries to predict genes are that it is accurate, and can find exon boundaries and alternatively spliced genes. Disadvantages are that low-expressed genes can not be found (not in the library), and the quality of the library is not very high.

Genome comparison As regions coding for proteins or functionally required RNA like tRNA are under larger constraints than non-coding regions, it is hypothesized that highly similar (conserved) regions between the same chromosome fragment in different species, e.g. mouse and human, are more likely to be coding regions, and thus exons for genes (O'Brien *et al.*, 1999; Parra *et al.*, 2003). However, if genomes are too closely related, all regions are similar, not just genes, and if genomes are too far apart, analogous regions may be too dissimilar to be found. Thus a number of genomes with different evolutionary distances are needed to clarify the picture. This reasoning also applies to regulatory regions. Identifying genes by genome comparison is more useful for eukaryotes due to the difficulties in identifying the coding sequence boundaries (e.g. Taher *et al.* (2004)), and the fact that prokaryotes genes are shuffled at a higher frequency, making alignment of long DNA sequences less useful.

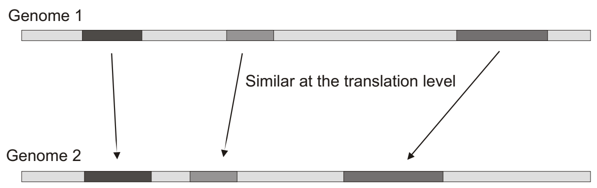


Figure 1.6: Comparing relatively long DNA sequences identifies conserved regions and, hence, putative coding regions.

Protein sequence comparison The most reliable method of gene finding is using homology searches (e.g. using BLAST and/or FASTA). By blasting an entire protein database against the genome sequence, a large number of genes can be identified. A problem with prokaryotic genomes is that they tend to be gene rich (80%-90% of the sequence is coding), which can make it difficult to determine which of two or more overlapping reading frames contains a gene. In addition, the quality of protein databases is not very high, which means finding genes without known homologues cannot be based on such searches.

Statistical gene-finding Many prokaryotic gene finding programs exist, often based on the identification of start and stop codons, and using statistical models (Delcher *et al.*, 1999; Salzberg *et al.*, 1998). Although many programs exist for eukaryotes too, the job is made much more difficult by the presence of splice sites, introns (which can be much larger than the coding sequence of the gene), and overlapping genes. In addition, the results must still be evaluated with other methods such as homology models or cDNA libraries (see Mathe *et al.* (2002) for a review).

1.5.3 Regulatory regions

Regulatory (promoter) regions in DNA sequences do not follow a strict pattern, which makes the identification of promoter regions more difficult. In prokaryotes and eukaryotes, promoter regions vary, although it is usually possible to find a DNA sequence (called the consensus sequence) to which all of them are very similar. For example, based on the study of 263 promoters, the consensus in the bacterium *E. coli* is TTGACA followed by 17 uncorrelated base pairs, followed by TATAAT, with the latter, called TATA box, located about 10 bases upstream of the transcription start site. None of the 263 promoter regions exactly match the above consensus sequence.

As genomes of prokaryotes are coding-sequence rich, the location of the promoter is almost always upstream of the gene, and methods of prokaryotic gene finding are relatively complete and accurate, automated prediction of promoters is of little use. The identification of general regulatory regions in prokaryotes usually follows from the identification of the gene or operon.

For eukaryotes it is more difficult. Not only can coding sequences be interspersed with introns, the regulatory regions may lie within the introns or even the coding region itself. It is not uncommon to find promoter regions hundreds of times bigger than the gene itself. For example, the gene involved in the disease cystic fibrosis, CFTR, produces a 6100-bp mRNA transcript. The transcript is produced from 27 exons scattered over 250,000(!) base pairs (250 kb) of genomic DNA. During transcription, the introns are spliced out and exons are pieced together (Zielenski *et al.*, 1991).

Despite advances in computational promoter-prediction in eukaryotes, multiple cross-species genome comparisons covering a range of distances, from closely to distantly related species, will likely be the most fruitful approach (Enard & Pbo, in press).

1.5.4 Gene function

Hidden among the avalanche of genome sequences is potential knowledge about gene function. Until recently, there have been two main ways of finding out more about a protein. The primary source of information comes from biological experiments, either genetic, structural, or biochemical. Once the function has been somewhat determined, one can search in the sequence databases for proteins with a high similarity. This is the 'homology' method, widely used to extend knowledge from one protein sequence to another under the assumption that genes with enough sequence similarity will likely be functionally similar as both sequences arose from the same ancestral sequence. BLAST (Altschul *et al.*, 1997) and similar programs are powerful enough so that this process can easily be applied to whole genomes, resulting in roughly 40%-70% of new prokaryote sequences receiving some functional assignment (Tatusov *et al.*, 2000).

A problem is of course when there is no characterized sequence similar to the unknown protein: sequence homology cannot be used in this case. But there are other methods, and this is the focus of Chapters 2 and 3.

The process of labeling genes with their functions, whether predicted or experimentally confirmed, is called *annotation*. Due to the human input required by current annotation methods, genome annotation has proceeded much slower than the acquisition of raw sequences.

An example of annotation nomenclature is as follows:

Known Gene Predicted gene matches the entire length of a known gene.

Putative Gene Predicted gene contains regions related to known genes. Also referred to as "like" or "similar to".

Unknown Gene Predicted gene matches a gene or EST of which the function is not known.

Hypothetical Gene Predicted gene that does not contain significant similarity to any known gene or EST.

General problems of annotation include:

a) The assignment of a gene function is highly probabilistic in nature. A major source of uncertainty arises from the fact that assignment of gene function is often based on the functional assignment of a gene with some sequence similarity.

b) The functional annotation can be correct, yet too broad. For example, the term "sigma-F transcribed gene"[2] is not very specific.

c) The annotation may contain no information about the context of the protein, for example, which broader biochemical reaction it participates in e.g. histidine biosynthesis. A regulatory protein may have a specific biochemical function that needs interacting proteins to make sense, and these interacting proteins may be not fully known. This requires an annotation that mentions the pathway the protein is involved in. Thus, the 'function' of the protein can have two correct, but very different meanings.

There are pitfalls with blindly annotating sequences based on some arbitrary similarity threshold. The difference of a single amino acid can change the function of an enzyme. For example, many single gene diseases are often the result of a single mutation, e.g. cystic fibrosis. Also, the sequential annotation of sequences can produce creeping errors. Suppose, for example, that we have gene A whose function has been determined experimentally. Gene B has some sequence similarity to A and is given a hypothetical function. Gene C is then found to be similar to B,

[2]bsu:BG10088 csfB, yaaM; sigma-F transcribed gene, from the *Bacillus subtilus* genome from the KEGG database (Kanehisa & Goto, 2000)

and given a putative function. This can continue until the query sequence shares little similarity to the original characterized sequence A (for a review of quality control in molecular biology databases see Aboa *et al.* (2000)). This means that annotation strategies must rely on a large amount of human input from knowledgeable annotators.

1.6 Gene function prediction methods

This section details the state-of-the-art in gene-function prediction using non-homology methods. To clarify and avoid confusion: most non-homology methods still use similarity searches, e.g. BLAST (Altschul *et al.*, 1997), to find homologous sequences. However, they do not base the prediction *only* on this information. Rather, this information is used as a basis for further exploration. What these methods have in common is that they use genetic variation between organisms to estimate protein function.

1.6.1 Domain fusion method

This method is based on the observation that sometimes functionally related proteins in one species are fused together in a multi-domain protein in another species (Marcotte *et al.*, 1999). The assumption is that if a composite protein is similar to two component proteins in at least one other species, then the two proteins are likely to interact or be involved in the same reaction or process. The composite protein is called a Rosetta Stone sequence in Marcotte *et al.* (1999) because it helps to decipher the interaction between the protein pair (see Figure 1.7).

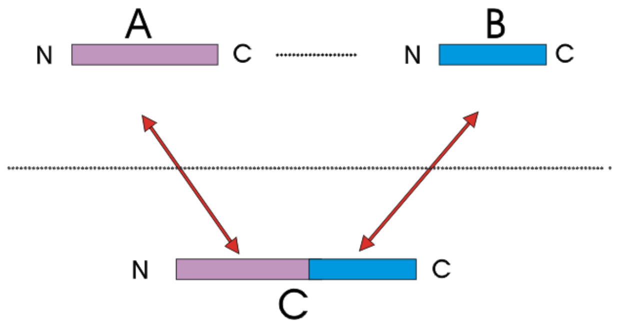


Figure 1.7: Domain fusion method to predict protein function: genes A and B are separate genes in one genome, gene C is a multi-domain protein in another genome. Genes A and B are likely to be involved in the same reaction or pathway because they have fused into C at some point. Figure from Enright *et al.* (1999).

In Enright *et al.* (1999), the authors compared the genes of two eubacteria (*E. coli* and *H. influenzae*), an archaea (*M. jannaschii*), and a eukaryote (*S. cerevisiae*) with each other. They showed that 215 genes or proteins from the three prokaryotes are involved in 64 gene fusion events. The minimum number of genes involved in fusion events is 2.8%, based on the three prokaryote genomes.

1.6.2 Conservation of relative gene position

Related to the analyses above, an inference of functional relatedness can be made between two proteins if they are located close to each other over multiple genomes (Figure 1.8). The conserved-position information can be the preservation of genes as neighbours (Dandekar *et al.*, 1998), or preservation of genes in runs (Overbeek *et al.*, 1999) where a set of genes share the same direction of transcription.

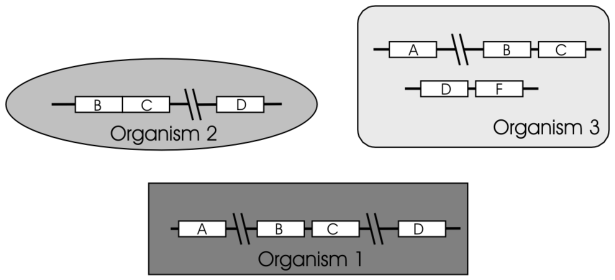


Figure 1.8: Three different organisms with different gene orders. It is inferred that genes B and C are functionally related because they are in proximity to each other in multiple genomes. (Figure adapted from Marcotte (2000))

Tamames *et al.* (1997) classed all the genes from *H. influenzae* and *E. coli* into 9 functional classes. They then performed statistics on neighbouring genes, and found that functionally related genes tend to be neighbors more often than unrelated genes.

The conservation of gene position has probably much to do with the organization of prokaryote genes into *operons*, a set of sequential genes that are regulated by the same promoter, for example the *lac* operon (Jacob & Monod, 1961). Horizontal gene transfer has been proposed as a driving force for this organization (Lawrence & Roth, 1996). True identification of an operon requires the identification of promoters and regulatory elements, but for gathering a subset of likely operons, the above methods are well suited.

1.6.3 Clusters of Orthologous Groups (COG)

COGs (Tatusov *et al.*, 1997, 2000, 2001, 2003) are sets of proteins from different (mostly prokaryote) organisms grouped so that their functional equivalence is maximised. A COG consists of putative orthologous proteins from at least three different organisms, where each protein is the symmetric top-scoring protein or bi-directional best hit in a sequence-homology search in the other genomes. In other words, a query sequence from genome A has an orthologue in genome B if the best match from genome B also returns the query sequence from genome A as the best match. This must be true for at least three genomes to define a COG.

An assumption is that genes clustered in the same COG often have the same function. Thus, functional information can be transferred to new sequences. The use of COGs can be a useful step in genome annotation projects, especially now that the COG database was recently updated to include more multi-cellular eukaryotes as their genomes have become available (Tatusov *et al.*, 2003).

Some interesting observations arose by computing COGs for many species: 55-83% of proteins encoded by bacterial and archaeal genomes can be placed into distinct and non-overlapping COGs, suggesting that many genes present in bacteria and archaea are highly conserved.

The COG database has provided a platform for subsequent analyses, such as:

Identification of species-specific genes

Using the COG database, Forterre (2002) searched for all the genes present in hyperthermophile genomes, but absent from all thermophile and mesophile genomes. He found only one, reverse gyrase. This gene is clearly a crucial adaption to life at high temperatures.

To infer Phylogeny

Snel *et al.* (1999) proposed using the presence or absence of a set of genes to infer phylogenies. Traditional phylogenies are constructed using sequence alignment and comparison, but are sensitive to inconsistencies due to horizontal gene transfer, unrecognized paralogues and highly variable rates of evolution. By using the COG database and gene content, a similarity score was defined between two species as the number of genes that they have in common divided by their total number of genes. The phylogenetic tree constructed correlated with the standard reference of prokaryotic phylogeny which is based on sequence similarity of 16s rRNA (Olsen *et al.*, 1994).

1.6.4 The use of the presence or absence of genes

Protein phylogenetic profiles

Pellegrini *et al.* provided a method for gene-function prediction based on the presence or absence of genes within a genome (Pellegrini *et al.*, 1999). The starting assumption is that the proteins that function together, e.g. by forming a functional complex, will likely evolve in a correlated fashion. Therefore during evolution, the functionally linked proteins will be preserved or eliminated as a unit. An additional argument supporting this assumption is the effect of horizontal gene transfer on genome structure (Lawrence & Roth, 1996).

The prediction of gene function is accomplished as follows: starting with a genome, each protein encoded in that genome is represented as a string of bits, where each bit represents the presence or absence of a corresponding homologue in other genomes. The string of bits is called a *profile* (Figure 1.9).

A zero means no homologue has been found. The profiles are clustered according to their similarity, and a group of similar profiles implies a common functional relationship. Note that two proteins in the same organism are thus considered to be functionally related if the collection of organisms in which homologues have been identified, more or less coincide (and not if their sequences exhibit some degree of similarity). By returning the proteins with one or more bits different, the scope is widened to include proteins that have had a counterpart replaced in one or more species.

It was shown that proteins with a similar profile also have similar keywords in the annotation, and vice versa, proteins with similar keywords tended to have similar phylogenetic profiles.

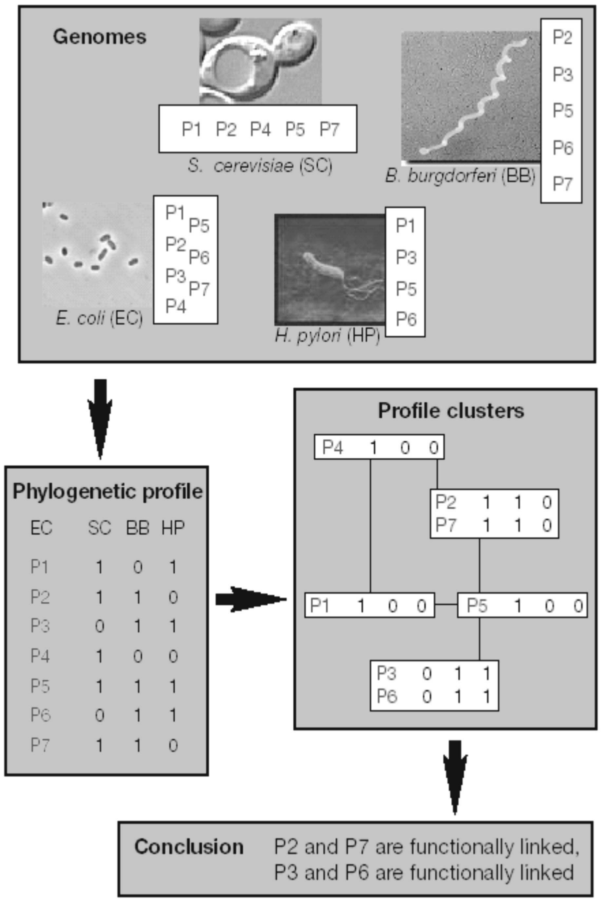


Figure 1.9: The method of phylogenetic profiles is illustrated with four hypothetical genomes (top), each containing a subset of several proteins labelled P1, ..., P7. The presence or absence of each protein is indicated by 1 or 0, respectively, in the phylogenetic profiles given on the lower left. Identical profiles are clustered in boxes on the right, with profiles differing by one bit connected by lines. The conclusion at the bottom is that proteins P2 and P7 are functionally linked because they have the same phylogenetic profile and, similarly, that proteins P3 and P6 are functionally linked. Note that two proteins classified as functionally linked from this classification procedure neither requires nor implies sequence similarity. Original figure from Pellegrini *et al.* (1999).

Three proteins[3] were examined in more detail to see if the proteins with similar phylogenetic profiles were functionally linked. For all three, almost all the proteins with identical profiles were involved in the same function.

Extended phylogenetic patterns

Reichard & Kaufmann (2003) released an application that computed phylogenetic patterns as in Forterre (2002) and Pellegrini *et al.* (1999) using the COG database ((Tatusov *et al.*, 1997)

[3]The ribosomal protein RL7, the flagellar protein FlgL and the histidine biosynthesis protein His5.

Perfect Match	One-needed
$\{\mathrm{G_c}\} = \{\mathrm{G_t}\}$	$\{\mathrm{G_{c'}}\} \cap \{\mathrm{G_{c''}}\} \cap \{\mathrm{G_{c'''}}\} ... = \{\mathrm{G_t}\}$
Similarity Measure	**All-needed**
$\frac{\parallel \{\mathrm{G_c}\} \cap \{\mathrm{G_t}\} \parallel}{\parallel \{\mathrm{G_c}\} \cup \{\mathrm{G_t}\} \parallel} \geq \mathrm{X}$	$\{\mathrm{G_{c'}}\} \cup \{\mathrm{G_{c''}}\} \cup \{\mathrm{G_{c'''}}\} ... = \{\mathrm{G_t}\}$

Table 1.1: Algorithms used in Levesque *et al.* (2003). $\mathrm{G_c}$ represents the set of genomes with $\mathrm{COG_c}$. $\mathrm{G_t}$ represents the set of genomes with trait t. X is a threshold value that is adjusted between 0 and 1 to set the stringency of the algorithm. A value of 1 for X generates the same results as the Perfect Match algorithm (and the Include/Exclude method described in Section 2.1.3).

and Section 1.6.3), but allows one to relax the restrictions. Unfortunately, the program runs only under Windows.

Differential genome analysis

Huynen *et al.* (1998) first introduced the idea of classifying genomes according to the phenotype and then finding genes specific to that phenotype. From the genome of *H. pylori*, they subtracted the orthologues shared with *E. coli* and *H. influenza*. This resulted in a subset of *H. pylori's* genes which would more likely contain genes responsible for features only seen in *H. pylori*. The subset of genes was then analysed by the authors. This is the earliest work that is similar to the research presented in the next chapter. The problem encountered in this work was the small dataset (only three genomes), and the lack of a phenotype definition, as species specific genes could cover a potentially large number of phenotypes.

Trait to Gene

This method by Levesque *et al.* (2003) follows from the analysis in the previous paragraph and is closest in ideas and implementation to the research presented in the following chapter. The authors wrote some simple set-theoretic algorithms designed to extract genes from a genome that has orthologues in some genomes but not in others. The genomes are classified into those possessing a phenotype and those that do not. Then, one could gather the genes present in all or most of one set of genomes (exhibiting the phenotype), and not present in another subset (not exhibiting the phenotype).

The authors tested combinations of four algorithms (Table 1.1) using *B. subtilus* as the query genome, and the COG (Tatusov *et al.*, 1997) database for the orthologues. With genomes classified according to the presence of flagella or not, a set of genes was predicted to play a functional role in flagella. Under stringent settings, most of these predicted genes did indeed play a role in flagella function, synthesis, or regulation (according to some pre-established sequence annotation). A subset of genes without flagella annotation was selected for further exploration. Three of the genes from this subset were knocked out, and a reduction in motility was observed for two of them (see Figure 1.10), confirming their role in flagella functioning. Potential limitations of this method is the reliance on the COG database for the source of orthologues, as the COG database only contains a representative set of genomes (see next chapter).

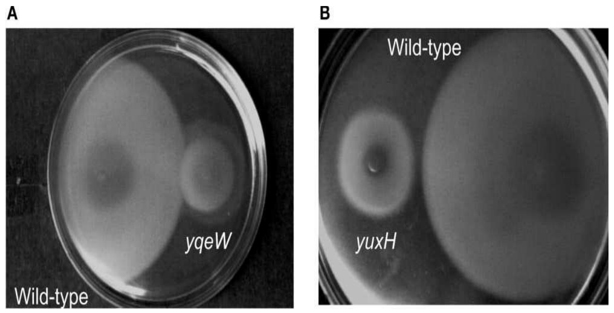


Figure 1.10: The motility was assayed for two *B. subtilis* gene knockouts, yqeW (B) and yuxH (B). Compared to the wild type, a reduction in motility was observed (the mutant covers less of the agar plate), confirming a role in flagella function for the two genes. (Photo from Levesque *et al.* (2003)).

1.6.5 Correlated mRNA measurements

The methods described above require the use of homologous genes, even though sequence homology was not the basis for function assignment. Other methods do not require direct sequence homology. They still rely on the assumption that genes rarely work in isolation and that functionally linked genes are often expressed at the same time or place. By measuring the cellular mRNA levels using DNA microarrays (Lashkari *et al.*, 1997), serial analysis of gene expression (SAGE) libraries (Velculescu *et al.*, 1995), or expressed sequence tag (EST) libraries (Adams *et al.*, 1991), and varying the conditions that cells are grown in or choosing cells from different tissues/species, it is hoped that enough variation in gene expression is observed to delineate the functionally-linked genes. Clustering seems to perform well for strongly expresses genes, but less so for other genes, and also requires large data sets.

1.6.6 Protein co-expression analysis

Similar to mRNA co-expression analysis, proteins can also be measured under different conditions and co-expression patterns established. Protein levels can be measured directly via mass spectrometry of protein mixtures and by various two-dimensional electrophoresis gel techniques. This approach is likely to yield better results as the mRNA and protein levels in a cell are basically uncorrelated (Gygi *et al.*, 1999). There is a growing awareness that post-transcriptional gene regulation is more complex than previously anticipated (especially in eukaryotes), and could explain the lack of correlation between gene numbers and complexity (Mattick, 2001). In addition, the fate and function of a protein within a cell is often dependent on the relative spatial localisation within a cell, meaning that multiple combined technologies (e.g. Schubert (2003)) are probably needed to fully understand protein function.

1.6.7 Gene function prediction using protein 3D structures

Gene function prediction is also possible if there is a similar structure in a database, even if there is very little sequence similarity. The active site of an enzyme is highly constrained during evolution, and thus is the most conserved, while the rest of the protein may be mostly ‘scaffolding’. In this case, alignment of two structures may show which amino acids are conserved. Functional similarity is implied between two proteins when the active site of both is conserved.

The increase in genomic sequences also fuels the growth of related databases, such as protein sequence, 3D structure, and other more specialized databases. The exponential increase in biological data presents several challenges, from searching and storage to integration (Davidson *et al.*, 1995).

1.6.8 Network reconstruction from protein function predictions

Most methods described above are large scale in nature or amenable to genomic-scale approaches. This means that one can computationally begin building a genome-wide network of interactions. This was done experimentally for yeast proteins by Osman (2004). In addition, much work is being done on reconstructing networks computationally from e.g. microarray data. This is the focus of the second part of this thesis, beginning with Chapter 1.

1.7 Graphs and networks: definitions and examples

The study of networks pervades all areas of science, from neurobiology to statistical physics (Strogatz, 2001), including the analysis of gene and protein interactions. The concept of networks is not just a philosophical concept. Research into real world examples of networks reveal properties such as the dynamics and stability of systems (Albert *et al.*, 2000), and motivate hypothesis-driven research into new directions (Strogatz, 2001). For example, many networks, both biological and man-made, exhibit a property called ‘scale-free’ (see Figure 1.11). Put simply, this means that the distribution of edges per node does not change with the size of the network. Several properties distinguish scale-free networks from other types of networks such as ‘exponential networks’ (exponential refers to the node degree distribution). Scale-free networks have a smaller proportion of nodes with many links; scale-free networks retain overall connectivity much longer with the random removal of nodes. However, the overall connectivity can fall faster after the targeted removal of nodes.

In many biological fields, a network is a principal means to describe what is known about an entire collection of entities. In simple terms, they detail the relationships of entities with each other. As biology is very much about its objects and how they interact, it is a very natural formalism.

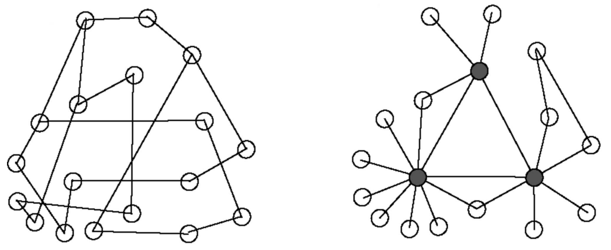


Figure 1.11: The analysis of global properties of different networks have identified two types that differ according to the probability distribution of edges. The exponential network (left) is homogeneous: most nodes have approximately the same number of edges. The scale-free network (right) is inhomogeneous: the majority of the nodes have one or two links, but a few nodes have a large number of links. This property has several interesting consequences, for example a higher resistance to random attacks (Albert *et al.*, 2000).

1.7.1 Definitions

A graph is a symbolic representation of a network. In the work presented in this thesis, it implies an abstraction of the reality it represents as a set of linked nodes. Intuitively speaking, a graph is a set of objects called vertices (or nodes) connected by links called edges (or arcs). Typically, a graph is depicted as a set of dots (i.e., the vertices) connected by lines (i.e., the edges).

Formally, a graph $G = (v, e)$ is a set v of vertices (nodes) connected by edges (links) represented by a set e of pairs x,y of elements of v representing the nodes x and y linked by the edge represented by the pair x,y. Depending on the network studied, edges may or may not have a direction; edges joining a vertex to itself may or may not be allowed, and vertices and/or edges may be assigned weights, that is, numbers. If the edges have a direction associated with them (indicated by an arrow in the graphical representation), then it is a directed graph. A graph with only one vertex and no edges is the trivial graph or “the dot”. A graph with no edges is known as the empty graph, but is sometimes also known as the Null graph.

Graphs can be represented conveniently by an *adjacency matrix*. This is an $N \times N$ matrix, where N is the total number of vertices in the graph. If there is an edge from some vertex x to some vertex y, then the element $M_{x,y}$ would be 1, otherwise it would be 0. This makes it easier to find subgraphs, and to reverse graphs if needed. In addition, for weighted graphs, the number may be any allowed number representing the weight of the edge from vertex x to vertex y.

Commonly used properties of graphs: the *distance*, which is the sum of the edge weights between two nodes; the *diameter*, which is the number of nodes which must be traversed in order to travel from one node to another when paths which backtrack, detour, or loop are excluded from consideration; and *connectivity*.

1.7.2 Examples of network studies in different fields

Metabolic networks: The availability of whole genomes makes it possible to catalogue all possible reactions within a cell. Metabolic networks describe the metabolism of an organism, the chemical system that generates nucleic and amino acids, sugars: in fact all the chemical constituents that the cell needs in order to survive (Figure 1.12).

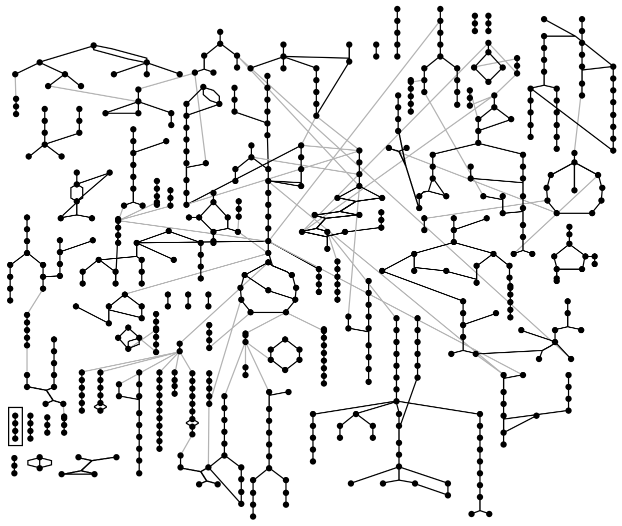


Figure 1.12: The biochemistry of a cell is commonly regarded a large network. The above biochemical reaction network shows all the known mass transformation reactions of *E. coli* intermediate metabolism from EcoCyc (Karp *et al.*, 1996). Each node represents a particular biochemical species, and an edge indicates an enzymatic reaction into another species. No details of the enzymes, regulatory reactions, or reaction directions are shown.

Ecological networks: In ecology, food webs (Figure 1.13) describe the food sources of species. Food webs are then digraphs where the vertices are species, and the directed edges link to those other species they consume, resulting in sometimes rather complex webs. Food webs are crucial in aiding understanding of ecosystem dynamics. Food webs enable the identification of 'keystone' species (Kotliar *et al.*, 1999). A classic example of a keystone species is the sea otter (Enhydra lutris). The sea otter preys on sea urchins in large numbers. When sea otter populations were removed by trappers and fishermen, sea urchin populations increased dramatically which led, in turn, to overgrazing of algae and kelp. Entire kelp beds were consumed, which caused declines in important commercial fish species that were dependent on the kelp beds. When sea otters were reintroduced, the kelp beds recovered (Estes & Palmisano, 1974).

Social networks: Social network analysis uses graph-theoretic concepts to describe, understand, and sometimes even predict social structures indicating, e.g., emergent patterns of relationships (e.g. Figure 1.14).

1.8 Modeling and simulating GRNs

The integration of recent advances in high-throughput biological techniques and the growing biological databases have enabled the construction of large-scale models of gene transcription and metabolic pathways (Covert *et al.*, 2004). These models form the basis in answering questions

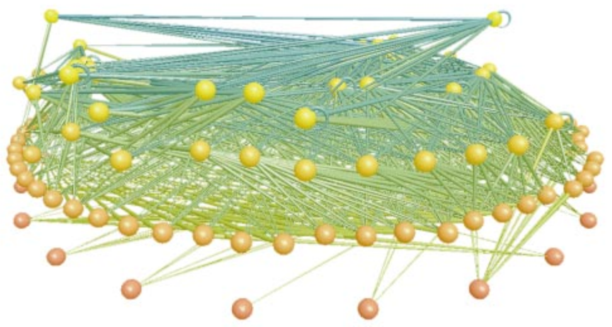


Figure 1.13: Food web of Little Rock Lake, Wisconsin, currently the largest food web in the primary literature (Williams & Martinez, 2000). Nodes are functionally distinct 'trophic species' containing all taxa that share the same set of predators and prey. Height indicates trophic level, with mostly phytoplankton at the bottom and fishes at the top. Cannibalism is shown with self loops, and omnivory (feeding on more than one trophic level) is shown by different coloured links to consumers. (Figure from Strogatz (2001))

.

concerning the evolution of entire genomes and regulatory networks (Papp *et al.*, 2004), as well as the fine grained structure of the latter (Shen-Orr *et al.*, 2002).

Typically, metabolic and regulatory (or signaling) networks are viewed as different entities. Modeling metabolic networks is an older research field due to the more developed experimental techniques to quantify the network components. In metabolic networks, the flow of mass and energy is the essential purpose of the machinery. In regulatory networks, the purpose is the regulation of other processes, i.e., the flow of information. The use of energy and mass flow is a requirement, but not the point. However, there is an essential component of regulation also in metabolic networks–the enzymes are regulated through interactions with substrates and products so that certain conditions in the cell are upheld.

Genomic databases such as KEGG (Kanehisa & Goto, 2000) contain not only genes and proteins but also metabolic pathways: a user can graphically navigate through common metabolic pathways. As no single organism possesses all pathways, organism specific pathways can be established by highlighting the genes present only in the query organism.

Several high-throughput methods have been developed recently that provide a broad glimpse at various cellular networks. The large-scale yeast two-hybrid screens (Uetz *et al.*, 2000) measure the binding of one set of proteins with a library of 6000 others, providing a large graph and a context for protein interactions. Lee *et al.* (2002) performed a large-scale system-wide assay of transcription binding factors, producing a network of transcription regulation. The raw data from these experiments has been used by others as a basis for studies into network topology and evolution (Milo *et al.*, 2004; Conant & Wagner, 2003).

Literature mining and sequence comparisons together (e.g. Salgado *et al.* (2004)) are also used for reconstructing global cellular regulation networks, and provide the raw data for several interesting studies into network structure (Shen-Orr *et al.*, 2002; Conant & Wagner, 2003; Milo

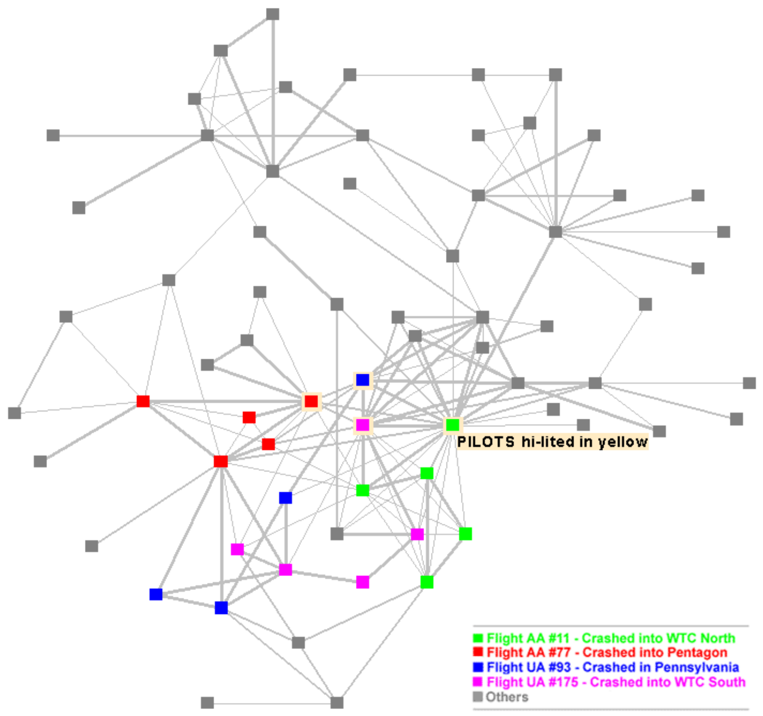

PILOTS hi-lited in yellow
Flight AA #11 - Crashed into WTC North
Flight AA #77 - Crashed into Pentagon
Flight UA #93 - Crashed in Pennsylvania
Flight UA #175 - Crashed into WTC South
Others

Figure 1.14: The September 11th terrorist network. In the network map above, the hijackers are colour coded by the flight they were on. The dark grey nodes are others who were reported to have had direct, or indirect, interactions with the hijackers. The gray lines indicate the reported interactions (from major newspapers on the Internet), a thicker line indicates a stronger tie between two nodes. From http://www.orgnet.com/hijackers.html

.

et al., 2004).

Recently, there has been much interest in the possibility of reconstructing[4] regulatory interactions from DNA microarray experiments. Several methods for modeling and reconstructing GRNs have been proposed in the literature and will be reviewed in the next two sections.

The different computational methods used to reconstruct gene regulatory networks depend very much on the mathematical formalism used to model the networks. What follows is a discussion of different mathematical formalisms and computational methods to model and simulate gene regulatory networks. What is not discussed here is metabolic modeling: modeling gene regulatory networks is different in that small metabolites are usually ignored, the focus is on the information processing of biomolecules rather than understanding the flux of metabolites.

[4]In this text, the word ‘reconstruction’ means the process of elucidating the structure and dynamics of gene regulatory networks from experimental observations.

Mathematical models of gene networks range in their detail, from simply stating if there is a connection or not between two nodes, to modeling the strength and type of the interaction, to full differential equations describing the dynamics of the system. Nodes usually represent genes or proteins, but can also represent groups of genes, depending on the granularity and detail of the model. Edges represent the interactions between the nodes.

There are many ways to represent gene networks (de Jong, 2002), and the following discusses the most common and relevant representations for regulatory network reconstruction. Although some simulation methods are not currently utilised by reconstruction methods, such as stochastic simulations, they are included because they are important for analysing dynamic genetic regulatory networks and provide a context for the choice of formalism and simulation used in this study. Some formalisms are not discussed here because they are not suited for network reconstruction methods, such as rule based formalisms (Meyers & Friedland, 1984; Brutlag *et al.*, 1991). Others are omitted because their impact regarding the dynamic analysis of genetic regulatory networks has been limited, including Petri nets (Goss & Peccoud, 1998; Hofestaedt & Thelen, 1998; Matsuno *et al.*, 2000), transformational grammars (Collado-Vides *et al.*, 1998), and process algebra (Regev *et al.*, 2001).

1.8.1 Directed graphs

All formalisms discussed here refer to directed graphs. This is perhaps the simplest description of networks, where edges are represented simply as links from one node to another, with possibly a direction (Figure 1.15). Extra information is sometimes included, such as the edge weight. This description is most often shown in published studies of gene regulation. In fact, most databases of gene and protein interactions can be interpreted as storing directed networks, for example KEGG (Kanehisa & Goto, 2000), the database of genes and metabolic networks. A directed graph description says nothing about the dynamics of the network, it is simply a description of the topology.

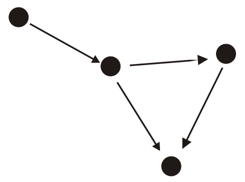


Figure 1.15: A simple directed graph. The circles are called nodes (or vertices) and could represent genes for example. The arrows connecting the nodes are called edges, and represent regulatory interactions. The network is directed because gene interaction is not necessarily symmetric. This is a simple way to represent gene regulatory networks.

1.8.2 Boolean networks

The study of Boolean networks as abstractions of genetic regulatory networks was pioneered by Stuart Kaufmann in 1969 and summarised in his book “The Origins of Order : Self-organization and Selection in Evolution" (Kauffman, 1993) where, for example, Boolean network models were used to study the global properties of large-scale regulatory systems.

In Boolean networks, genes are either in one of two states: active or "on", represented by 1, and inactive or "off", represented by 0. The interactions between genes are expressed as boolean functions computing the state of any gene as a function of the states of other genes. As each gene can have only two states, the state space of the network consists of 2^n states. The transition from one state to the next can be computed in one step (synchronous) or with a delay (asynchronous). Boolean networks are deterministic which means the state space is finite and that the network will eventually reach a steady state or a state cycle, also called a point or dynamic attractor, respectively.

Boolean models make simplifications about the dynamics of gene networks, enabling the efficient analysis of large networks. However, there are several drawbacks. By ignoring intermediate states of gene expression and updating genes synchronously, realistic dynamic behaviour is lost. A simple example illustrates this: imagine two genes that negatively interact with each other, i.e., reduce the other gene's state to 0 when this state is 1. In addition, both genes have positive interactions from other genes. In real cellular systems, this is called a bi-stable switch, where one gene will suppress the activity of the other resulting in a stable steady state. However, in a Boolean model, this setup will cause an oscillation, as each gene successively reduces the state of the other to 0. This is a completely different behaviour from that observed in real genetic systems, which will consequently affect the dynamics of the entire network.

1.8.3 Generalised logical networks

These can be viewed as an extension of Boolean networks, where instead of only two possible states for genes, multiple states exist. These models have been described most recently by Thomas (1991). In these models, genes have an abstract level of concentration. That is, its states are not given by numbers expressing its concentration; rather, they are characterised by the various sets of genes it can influence at its various states, where each state is associated with an influence on a subset of it targets. Specifically, if a gene influences the concentration of p other genes, then it may have up to $p + 1$ different states. The state of a gene is determined by a function of the states of the influencing genes.

As the model consists of a finite number of states, the number of steady states can be exhaustively determined. The effectiveness of this model has been demonstrated in a number of small-scale regulatory systems such as λ phage infection in *E. coli* (Thieffry & Thomas, 1995), pattern formation in *Drosophila* (Snchez & Thieffry, 2003), and flower morphogenesis in *Arabidopsis* (Mendoza *et al.*, 1999).

Despite this formalism showing promise in modeling small systems, it does not scale well for large networks and lacks the predictive power of physically based models (Gilman & Arkin, 2002).

1.8.4 Differential equations

Differential equations are probably the most frequently used formalism for modeling dynamical systems in science and engineering and have also been used to study gene regulatory networks. If a gene network consists of n entities (proteins, mRNA, and/or small molecules), there exist n rate equations, one for each entity, describing its rate of synthesis and decay. The general form of the equation is:

$$\frac{dx_i}{dt} = f_i(x_1, \ \dots, x_n) \tag{1.1}$$

where x_i is the concentration of an entity , and its rate of synthesis and degradation is dependent of the concentrations of $x_1, ..., x_n$ (the other entities in the network). f_i is called the *regulation function*, as it is a function of other metabolites and gene products. If f_i is positive, x_i will increase in concentration. If f_i is negative, the concentration of x_i will decrease, however x_i cannot go below zero, as this makes no biological sense. Time delays, for example waiting for the transcription to finish, can also be represented. Mathematical methods for modeling metabolic biochemical reactions by rate equations are quite highly developed (for introductions see Cornish-Bowden (1979); Heinrich & Schuster (1996); Voit (2000)).

Linear models

Despite the fact that gene regulatory systems are inherently non-linear, assuming linear interactions can make the system much more tractable, in that solutions to the system can be found. Linearity can also be assumed when analysing a system in a small neighbourhood of a given state. Most linear equations of gene regulation have at their core a linear regulation function, where total regulatory input from other genes is determined by summing the regulatory influences:

$$h_i = \sum_j w_{ji} x_j \tag{1.2}$$

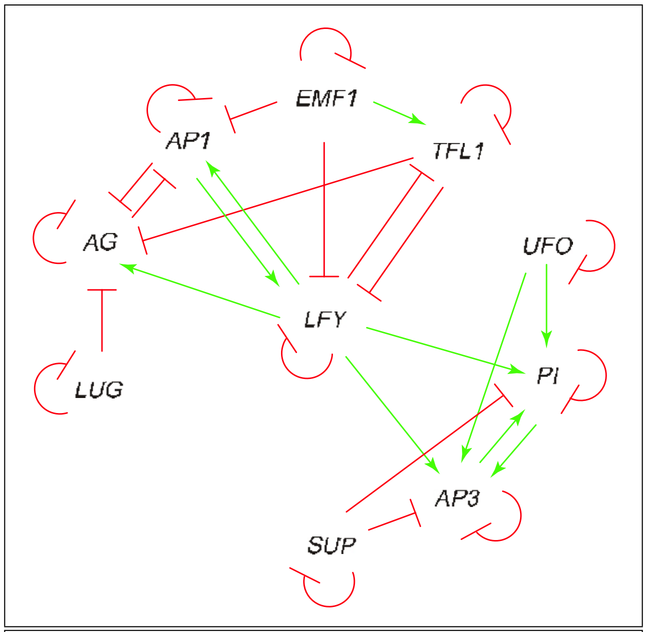
where x_j is the expression level of gene j and w_{ji} is the weight of gene j to gene i. A positive value models activation while a negative value models repression. This results in a linear equation of the form

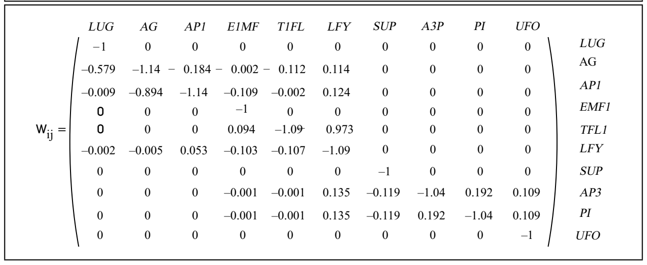
$$\frac{dx_i}{dt} = h_i + b_i \tag{1.3}$$

where b_i is a basal rate of expression. Assuming the basal rate of transcription, b_i, is the same for all genes, the regulatory network can be represented by the matrix w_{ij} of regulatory coefficients (See Figure 1.16).

An example of a linear system of two genes is shown below. The slope of $\frac{dx_1}{dt}$ is given in Figure 1.17.

$$\frac{dx_1}{dt} = w_{11}x_1 + w_{12}x_2 + b_1$$
$$\frac{dx_2}{dt} = w_{21}x_1 + w_{22}x_2 + b_2$$





$$W_{ij} = \begin{array}{c|cccccccccc|l}
 & LUG & AG & AP1 & E1MF & T1FL & LFY & SUP & A3P & PI & UFO & \\
\hline
 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & LUG \\
 & -0.579 & -1.14 \; - & 0.184 \; - & 0.002 \; - & 0.112 & 0.114 & 0 & 0 & 0 & 0 & AG \\
 & -0.009 & -0.894 & -1.14 & -0.109 & -0.002 & 0.124 & 0 & 0 & 0 & 0 & AP1 \\
 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & EMF1 \\
 & 0 & 0 & 0 & 0.094 & -1.09 & 0.973 & 0 & 0 & 0 & 0 & TFL1 \\
 & -0.002 & -0.005 & 0.053 & -0.103 & -0.107 & -1.09 & 0 & 0 & 0 & 0 & LFY \\
 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & SUP \\
 & 0 & 0 & 0 & -0.001 & -0.001 & 0.135 & -0.119 & -1.04 & 0.192 & 0.109 & AP3 \\
 & 0 & 0 & 0 & -0.001 & -0.001 & 0.135 & -0.119 & 0.192 & -1.04 & 0.109 & PI \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & UFO
\end{array}$$

Figure 1.16: Top: directed graph showing the gene network controlling flower morphogenesis in *Arabidopsis thaliana*, proposed by Mendoza *et al.* (1999). Green arrows represent activation, red lines ending with a perpendicular line indicate inhibition. The network can also be represented by the matrix below the network. Adapted from de la Fuente *et al.* (2002).

The reaction rates of x_1 and x_2 are linear functions of the gene product concentrations plus a constant b_1 or b_2, respectively, representing a basal level of transcription. There are variants of this model (Mjolsness *et al.*, 1991; Chen *et al.*, 1999; D'Haeseleer *et al.*, 1999; Weaver *et al.*, 1999) which all at the core have linear additive weights for the regulation function, in the simplest case being simply the addition of weights and gene levels of Equation 1.4:

The assumption of linearity may be reasonable if the system is near some given steady state, however the predictive power of the model is limited outside that particular state.

Non-linear models

In reality, regulation functions must be non-linear because they saturate at some point. The most common approach is to use some kind of saturating function. Two such functions often used to model gene regulatory interactions are the sigmoidal transfer function, which is commonly used to model neural networks, and the Hill function. Both models assume that regulatory effects are

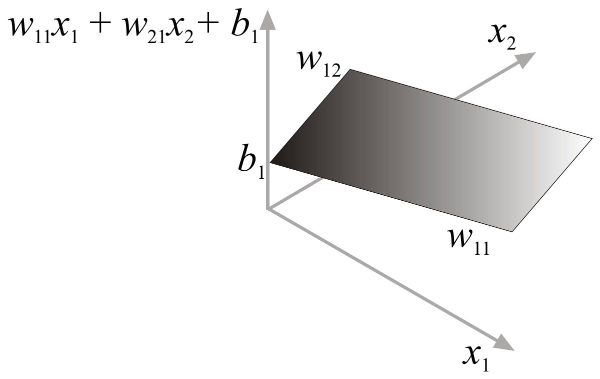


Figure 1.17: The quantity $w_{11}x_1 + w_{12}x_2 + b_1$ is the rate of change of the first gene x_1 as a function of the first gene (x_1) and the second gene (x_2). Geometrically, it is given by a plane with slopes w_{11} and w_{12} in the first and second co-ordinate directions, respectively. Biologically, these two numbers represent the strength of the effect of the two genes on the first gene (so that w_{11} in particular quantifies the feedback of the first gene back on itself). The parameter b_1 is the height of the plane above the origin $(x_1, x_2) = (0, 0)$. Biologically it represents the background level of transcription of the first gene. Figure adapted from Stark *et al.* (2003).

approximately additive, with the non-linearity confined to a saturating function. Both functions sum the regulatory inputs to a gene i as in the linear model

$$h_i = \sum_i w_{ji} x_j \tag{1.4}$$

where w_{ij} is the matrix of regulatory coefficients (Figure 1.16), and then convert the regulatory sum h_i to a saturating function. The *sigmoidal transfer function* has the form:

$$\frac{dx_i}{dt} = \frac{1}{1 + e^{-(\alpha_i h_i + \beta_i)}} - \mu_i x_i \tag{1.5}$$

where α_i and β_i are two gene-specific constants that define the shape of the dose-response curve (Figure 1.18) for gene i (the term *dose* means the total input weight e.g. from Equation 1.4, and *response* is the rate of transcription), and μ_i is the degradation rate for gene product i. The constant α_i can be any positive real-number value and defines the slope of the curve at its inflection point (50% maximal expression). The constant β_i can be any real number and moves the curve to the right as it becomes negative (Figure 1.18).

The other non-linear regulation function often used is the *Hill equation* (Hill, 1985). For modeling gene regulatory interactions it often has the form:

$$\frac{dx_i}{dt} = \frac{\Phi(h_i)^m}{\Phi(h_i)^m + \theta_i^m} - \mu_i x_i \tag{1.6}$$

where $\Phi(h_i)$ is a Heaviside function, which ensures that negative weights do not lead to negative concentration values, defined by $\Phi(x) := x$ for all positive x and $\Phi(x) := 0$ otherwise, θ_i is the threshold for the regulatory influence , and $m > 0$ determines the steepness of the slope

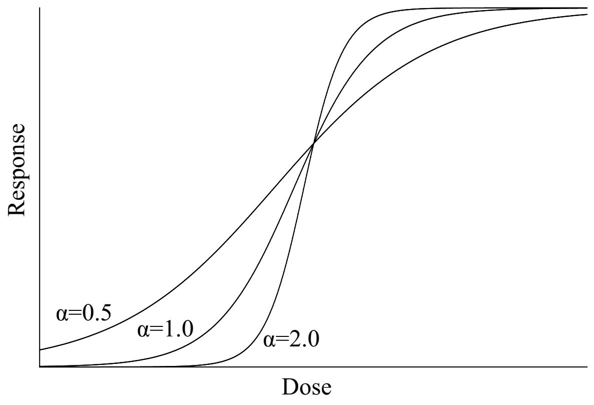


Figure 1.18: Dose-response curve for the sigmoidal transfer function (Equation 1.5). The dose is the sum of regulatory inputs given in Equation 1.4, and the response is the change in expression, $\frac{dx_i}{dt}$. Genes with a large corresponding α_i will shift rapidly from near zero expression to near maximal expression when the activating inputs surpass some gene specific threshold, i.e. they will display switch-like behaviour. Those with a small α_i will have a nearly linear response over the range of regulatory inputs. As β_i increases the curve shifts to the left.

(see below), and μ_i is the degradation rate for gene product i. Ignoring the degradation term $\mu_i x_i$, the regulatory function ranges from 0 to 1, approaching 1 as $h_i \rightarrow \infty$. When $m > 1$, Hill curves have a sigmoid shape, in agreement with experimental evidence (Yagil & Yagil, 1971; Yagil, 1975), see Figure 1.19.

Inhibition is handled slightly differently by the two functions. In the sigmoidal transfer function, a negative value of h_i means a lower rate of gene expression, but never reaching zero. For the Hill function, lowering inhibition can result in an expression rate of zero.

Despite being more realistic than linear models, there are a number of limitations to the above equations. Both regulation functions (Equation 1.5 and 1.6) assume that each gene has a static dose-dependent response to activating and repressing regulatory influences. It is assumed that each gene will respond with the same response curve to both positive and negative regulatory influences. Although the expression functions are non-linear, it assumes that the overall effect of the involved "partners" $x_1, ..., x_n$ on x_i can be modelled by composing a linear real-valued

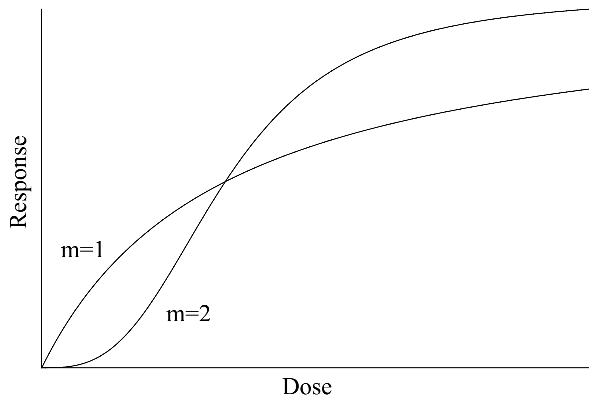


Figure 1.19: Hill plot: dose-response curve for the Equation 1.6. The dose is the sum of regulatory inputs given in Equation 1.4, and the response is the change in expression, $\frac{dx_i}{dt}$. For $m = 1$, the curve is of the Michaelis-Menten form (Voit, 2000). For $m = 2$, the curve is similar to the sigmoidal transfer function (Figure 1.18).

function of $x_i, ..., x_n$ with a non-linear function in just one variable, coming about by a weighted sum of the concentrations of the individual gene products. However, it is known that some transcriptional regulators have different activities depending on their binding partners (Garrell & Campuzano, 1991)

Because the Hill and sigmoidal transfer functions are non-linear, analytical solutions to the rate equations (1.1) are difficult to obtain. The equations can either be simplified into e.g. linear forms (see above), or numerical techniques can be applied. By numerically simulating the equations, the solution to the equations is approximated by computing values $x_i(t_0), ..., x_i(t_m)$ for x_i at consecutive time steps $t_0, ..., t_m$, $i = 1, ..., n$ (Lambert, 1991). There have been a number of well-studied regulatory systems that have been simulated numerically (see de Jong (2002) for a review).

A major problem with numerical simulations of regulatory systems is the lack of *in vivo* or *in vitro* measurements of the kinetic parameters of the rate equations. This problem may be alleviated by the fact that (a) regulatory systems are by necessity *robust* and tolerant to changes (i.e. mutations or measurement errors) in the kinetic parameters (von Dassow *et al.*, 2000), and (b) large-scale gene-expression measurement techniques may produce bounds for certain values.

According to the literature reviewed here, the sigmoidal transfer function is used more commonly for modeling gene networks than the Hill function. The Hill function is often used to model co-operative binding systems, where it has some validity (Hill, 1985). However in all the papers reviewed here, it is assumed that the binding sites are independent or, at least, the regulatory inputs are linearly additive (Equation 1.4). In most studies, it appears that only a general saturating function is needed and that either functions could be used without altering results or conclusions.

Other rate equations

Piecewise-linear differential equations
These are a special case of non-linear differential equations. The sigmoid shape of the curve in Equation 1.6, which gives rise to switch-like behaviour, is approximated by discontinuous step functions (Thomas & D'Ari, 1990). They are conceptually similar to Boolean networks and the simplicity enables easier mathematical analysis.
Qualitative linear differential equations
These are another generalisation of linear equations where the variables take on discrete *qualitative values* which abstract the real value. An example of an application has been given by Heidtke & Schulze-Kremer (1998), where a model was constructed that consisted of seven QDEs representing different stages of λ phage growth. Each QDE describes the infected bacterium including λ phages, viral DNA, ribosomes, mRNA and proteins, and how they interact to control cellular events. A major problem with qualitative differential equations is that they do not scale well unless the behaviour in phase space is tightly constrained, which means that the system must already be well studied.

1.8.5 Stochastic approaches

In real cellular systems, genes and regulatory processes are stochastic processes subject to biochemical noise (Rao *et al.*, 2002; Blake *et al.*, 2003). Although differential equations allow highly detailed models, down to the level of individual molecules binding to each other or to DNA, they assume that molecular concentrations vary continuously and deterministically. When the number of molecules in a particular reaction is small, stochastic effects can play a large role, and models should then include a stochastic component (Thattai & van Oudenaarden, 2001). There are three main stochastic approaches used to model gene regulatory networks: stochastic differential equations, stochastic particle simulations, and Bayesian networks.

Stochastic differential equations and particle simulations

Modeling of gene regulatory networks using stochastic differential equations is at a developmental stage. Tian & Burrage (2004) introduce a model based on the sigmoidal transfer function

(Equation 1.5) where Poisson random variables are introduced into the gene product synthesis and degradation process.

Rather than using differential equations, some authors have proposed a discrete and stochastic model of gene regulation (Gillespie, 1977; McAdams & Arkin, 1997; Arkin *et al.*, 1998). The numbers $\boldsymbol{X} = X_1, ..., X_n$ of distinct molecules are the state variables, and a joint probability distribution $p(\boldsymbol{X}, t)$ is introduced to express the probability that at time t the cell contains X_1 molecules of the first species, X_2 molecules of the second species, etc.

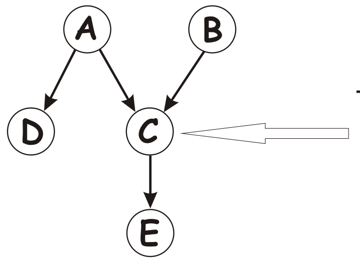
Another technique for simulating genetic systems is to directly simulate the time evolution of a discrete number of particles. In such a stochastic simulation by Gillespie (1977), stochastic variables indicate when the next reaction will take place and what kind of reaction it will be.

Despite many improvements to stochastic algorithms (for a review see Meng *et al.* (2004)), the complexity of estimating, solving, and analysing stochastic equations means that they are rarely used for larger networks with more than a few genes (Kepler & Elston, 2001).

Although stochastic simulations are closer than differential equations in approximating the underlying reality, this is not necessarily an advantage. The stochastic approach is computationally very expensive as individual molecules and reactions are simulated, limiting the size of the simulations. At large scales, stochastic effects may level out, meaning the continuous and deterministic models may be accurate enough an approximation (Gillespie, 2000).

Bayesian networks

Bayesian networks are probabilistic graphical models that represent the state of each gene as a multivariate joint probability distributions via a product of terms (Friedman, 2004). The joint distributions over a set $X = \{X_1, ..., X_n\}$ of random variables is represented as a product of conditional probabilities, where each variable X_i is associated with a conditional probability $P(X_i \mid \boldsymbol{U}_i)$ and $\boldsymbol{U}_i \subseteq X$ is a set of variables that are called the parents of X_i. Thus, the parents of X_i directly influence the choice of values for X_i (Figure 1.20).



P(C| A,B)

A	B	0	1
0	0	0.9	0.1
0	1	0.2	0.8
1	0	0.9	0.1
1	1	0.01	0.99

Figure 1.20: Left: a Bayesian network with five random variables, where vertices (*A* to *E*) are labeled with variable names. Edges represent direct dependencies. Right: the table is the conditional probability distribution for the vertex *C*, where the expression level of its parents is discretised to a Boolean value. The product form specified by this Bayesian network is $P(A, B, C, D, E) = P(A)P(B)P(C \mid A, B)P(D \mid A)P(E \mid C)$.

An advantage of using Bayesian networks to model gene networks is that it deals with the stochastic aspects of gene expression and noisy data in a natural way, and can be used when knowledge of the system is incomplete. However, the dynamical aspects of gene expression are not directly modeled, as the networks are acyclic, preventing network structures such as feedback loops to be modeled. This can be overcome with the use of *Dynamic Bayesian Networks* (Murphy & Mian, 1999).

1.9 GRN reconstruction methods

What follows is an overview of recently published methods of reconstructing gene regulatory networks from microarray data. The reconstruction methods are categorised according to the underlying model of gene regulation.

1.9.1 Reconstruction methods using directed graphs

As all gene-network models are in some sense directed graphs, the term here means *only* directed graphs, in that there is no modeling of the regulatory dynamics.

Many experimental techniques produce data in the form of directed graphs (Lashkari *et al.*, 1997; Osman, 2004) and many databases store information in the same form (e.g. Kanehisa & Goto (2000)), making comparison and validation of results possible. However, directed graphs are often produced from experimental data because such a representation contains less ambiguities, which means however that subtle, complex, or dynamic interactions usually cannot be represented. Most methods that use a dynamic model of gene regulation have difficulty testing predictions, as experimentally determined parameters for dynamic models are very difficult to acquire.

Reconstruction methods that ignore complex or dynamic interactions are often based on a form of *clustering*, where genes with a similar change of expression following a perturbation are clustered together (Arkin *et al.*, 1997; Tavazoie *et al.*, 1999; Jorgensen *et al.*, 2002). This can identify a set of genes for closer investigation, or establish a functional cluster to which a particular gene belongs. Clustering genes is not sensitive enough to distinguish intermediate reactions between two genes, to determine the nature of reactions within a cluster, or to understand the regulatory dynamics. It is useful as a first step for understanding complex processes.

Qian *et al.* (2003) used support vector machines (SVMs) to predict regulatory relationships of yeast expression data. SVMs require a training set for which Qian *et al.* used two publicly available transcription-factor-binding databases. A problem encountered was the lack of negative examples required to train the SVM. This was partially solved by using well studied transcription factors where the DNA binding sites are known. Any genes without a matching upstream DNA binding site constituted a negative example. They found that simple correlations of gene expression are not enough to correctly predict a regulatory relationship; gene expression time-series (profiles) were needed. The overlap between the predicted and experimentally derived regulatory relationships was small. However, the overlap between different experimental data sets from different groups was also small, indicating that the time and the state of the cell can drastically alter the network of regulatory interactions, as suggested by Schubert (2003); Luscombe *et al.* (2004).

1.9.2 Reconstruction methods using Boolean models

Liang *et al.* (1998) have developed an algorithm called REVEAL to infer Boolean networks from state transition tables, which correspond to time-series data. Akutsu *et al.* (1999) proved that $O(\log n)$ expression patterns are necessary and sufficient to identify the underlying Boolean network of n genes correctly with a high probability if the maximum number of incoming edges is bounded.

Akutsu *et al.* (2000) propose two models: a qualitative Boolean network model with noise, and a power-law model called an S-system, i.e., systems based on a particular type of non-linear differential equations. An inference algorithm is presented and tested on both network models. They found that for both models, correct prediction requires many different time-series with different initial values, representing different environmental inputs or gene mutations. The method using S-systems is not robust when noise is included in the system. They concluded that using linear equations in network reconstruction methods will make the methods difficult to apply to real data, as many real regulatory relationships are non-linear. The predictions were not compared with experimental data, as none were available.

1.9.3 Reconstruction methods using differential equations

Linear models, while being less realistic, can employ a large amount of tools available from linear algebra. Given an equidistant time series of expression data, one can use linear algebra to find a least-squares fit to the model. D'Haeseleer *et al.* (1999, 2000) showed that a simple linear model could infer biologically relevant regulatory relationships from real data.

Chen *et al.* (1999) presented a linear model including both mRNA and protein concentrations and two methods (one of which uses Fourier transforms) to reconstruct the model from gene expression data. Both reconstruction methods require extra assumptions: for the linear algebra approach, called Minimum Weight Solutions to Linear Equations, the assumption is that the number of regulators for each gene is a small constant, otherwise the method is intractable. For the method that uses Fourier transforms, gene expression has to be periodic in the cell cycle. Interestingly, the models did not lead to acceptable results with only mRNA levels measured – the protein concentrations were also required.

Methods of reconstruction using non-linear models require more general techniques due to complexity problems. Mjolsness *et al.* (1991) presented a model of development, called the connectionist model (D'Haeseleer *et al.*, 2000), that incorporated a phenomenological model including gene regulation based on neural net dynamics coupled with cell events such as cell division. Simulated annealing was used to fit the model to spatial data. Simulated annealing is a global optimisation method that is designed to avoid getting stuck in (non-global) local optima, however, unlike the evolutionary algorithms outlined in Section 1.10, there is no recombination between solutions.

Tominaga *et al.* (1999) represent gene regulatory networks with S-systems, which they attempt to fit to data using Genetic Algorithms (GAs) (section 1.10.2). Although the algorithm could accurately reproduce the parameters of a given S-system, the size of the system had to be kept very small ($>= 5$), otherwise the numbers of parameters to estimate became too large. GAs are also used in Wahde & Hertz (2000) where genes are modeled as continuous-time recurrent neural networks, similar to Equation 1.5, however the reconstruction algorithm deals with *clusters* of genes rather than individual genes. Similarly to Tominaga *et al.* (1999), the approach was only evaluated on very small networks, in this case with 4 genes.

It is demonstrated in Weaver *et al.* (1999) that a 'weight matrix model' (D'Haeseleer *et al.*, 2000) (which also uses Equation 1.5) could accurately reconstruct some randomly generated networks. Gene networks were modeled to include external inputs, representing environmental signals. The time series from a simulated network is 'de-squashed', giving a linear equation to be solved, which then results in a predicted weight matrix. The reconstruction method has difficulty with genes with very high or very low expression rates. In addition, the method relies on the assumption that it is possible to detect the maximal expression rate for each gene.

A 'coarse-grained' approach is described in Wahde & Hertz (2000) where regulatory networks are modeled by continuous-time recurrent neural networks, and reconstructed with a genetic algorithm. The term 'coarse grained' is used because individual genes with similar expression patterns are clustered into 'waves', reducing the complexity of the problem but also reducing the resolution. This was done because the reconstruction algorithm could not cope with a gene size greater than 4. A limitation to the method was that edges are not created or destroyed during the reconstruction procedure.

1.9.4 Reconstruction methods using Bayesian networks

The natural way Bayesian networks can account for stochastic aspects and errors in the raw data have made them a popular tool for network reconstruction (Friedman, 2004). Pe'er *et al.* (2001) used the framework of Friedman *et al.* (2000) to go beyond clustering genes based on expression data to infer a finer structure, such as mediation, activation, and inhibition. Instead of assuming a Boolean expression level for each gene as in Figure 1.20, Pe'er *et al.* introduce a procedure that calculates how many discrete levels of expression are needed for each gene. The methods provide an automated approach to finding significant subnetworks of interacting genes, using microarray experiments with perturbed levels of gene expression. Unlike most other studies, the authors applied their techniques to real data, in this case $\sim$280 *S. cerevisiae* deletion mutants. For three closely inspected genes, Pe'er *et al.* claim that the regulatory role was correctly reconstructed.

1.10 Evolutionary algorithms

Evolutionary algorithms represent a form of machine learning that is inspired by biological evolutionary processes. It has been observed since the 1950s that evolutionary processes can lead to highly optimised solutions to complex problems, where the solution space is too big for an exhaustive search for the global optimal solution. There are a variety of evolutionary computation models that have been proposed and studied. They share a common conceptual base, where a population of randomly generated solutions are evolved. The individual solutions in one generation are given a fitness value that reflects how good the solution is, and the best individuals are recombined and mutated to make the next generation. Although simplistic from a biological viewpoint, these algorithms are sufficiently complex to provide robust and powerful adaptive search mechanisms. What follows is an overview of the most common form of evolutionary algorithms.

1.10.1 Evolutionary strategies

Evolutionary strategies (ES) were developed by Rechenberg (1973) and extended by Schwefel (1981). In standard recombinative ES, children are produced via recombination then mutated. Then either the best N children survive, or the best N children and parents survive. Each individual can have an adaptive mutation rate, like Evolutionary Programming (EP, see below), enabling adaptive mutations. However, unlike EP, crossover (Figure1.21) plays an important role.

1.10.2 Genetic algorithms

Genetic algorithms (GAs) were invented by Holland (1975) and further developed by Goldberg (1989). In genetic algorithms, each individual in a constant-sized population is represented by a finite string of symbols such as bits, known as the *genome*. The string is sometimes called a chromosome or genome, inspired from biology. The genome can represent the weights, biases, or structure of a neural network, for example. Each genome represents a possible solution in the search space, which consists of all possible solutions. An example genome of bits may look like this:

010101110101001010011101101110111111110

At the beginning, a large number of genomes are randomly generated. Each one, when decoded will represent a different solution to the problem at hand. Biologically inspired operations, namely ‘crossover’ and ‘mutation’, on a population of genomes produce the change required for evolution. Crossover involves two ‘parents’ exchanging parts of their genomes. Mutations can be implemented as bit flipping on bit strings. With an initial population of N genomes, the procedure in a genetic algorithm for generating the next generation of solutions usually looks something like this:

- Assign each genome a fitness value which measures how well that genome solves the given problem
- Select two members from the current population. The chance of being selected is proportional to the genome’s fitness.
- Crossover the bits from each chosen genome at a randomly chosen point.
- Step through the chosen genome bits and flip dependent on the mutation rate.
- Repeat the above steps until a new population of N members has been created.

The algorithm clearly depends on the mutation and crossover rates. Crossover between two genomes is as follows: given two genomes to crossover, randomly choose a bit in one genome (a zero or a one), and swap all bits between genomes after that chosen bit. For example, given two genomes:

> 10001001110010010
> *01010001001000011*

choose a random bit e.g. at position 9, and swap all bits after that position, so the above two genomes become:

> 100010011*01000011*
> *010100010*10010010

After many rounds of selection, a hopefully optimal solution to the problem is encoded by the resultant genomes.

1.10.3 Genetic programming

Genetic programming (GP) (Koza, 1992; Banzhaf *et al.*, 1998) aspires to evolve actual computer programs, by inducing programs to change and automatically improve processing training data. As a computer program can be represented as a tree structure (Figure 1.21), genetic programming involves graph operations on a population of programs. Structures other than programs can be evolved, for example neural networks. Crossover and mutation are applied to the population to generate change. The GP representation is regarded as a general representation of many classes of algorithms including GAs. However, the programs evolved by GP can vary in length, whereas the individuals in GAs are fixed in length.

1.10.4 Evolutionary programming

Evolutionary programming (EP) was developed by Fogel *et al.* (1966)[5]. This approach has also been called evolutionary computation, however that term is used in this thesis as a general description rather than a specific methodology. In evolutionary programming, the representations of individuals are usually tailored specifically to the problem, for example using graphs to represent finite state machines. In this approach, the N parents are mutated and produce N children, and the next generation is chosen from the 2N individuals. Recombinations are not generally performed since the forms of the mutations are quite flexible, for example adapting the mutations rates to the current evolutionary improvement.

[5]Lawrence Fogel has set up a company, Natural Selection, Inc., which claims to apply evolutionary computation to a wide variety of real-world problems.

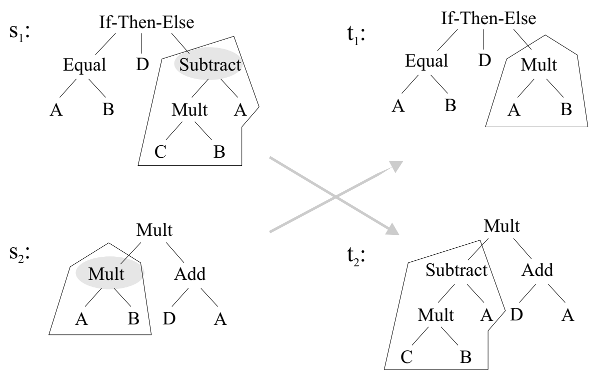


Figure 1.21: Crossover between two programs. In Genetic Programming, computer programs are represented as trees, and populations of programs are evolved to compute a particular function. To create the next generation, two parents of the current generation, s_1 and s_2, exchange subtrees (at the crossover point marked in grey) to form the next generation, t_1 and t_2. This process is called *crossover*, and is inspired by the biological process carrying the same name. Other mutations can be applied to individual programs, for example by randomly changing operators, or adding new variables.

Chapter 2

GeneTrawler: Motivation and Methodology

This chapter introduces Genetrawler, a tool for predicting the function of prokaryote genes. Developing gene-function prediction tools is highly relevant even with the current deluge of genomic information. The most studied microbe, *E. coli*, still has around 30% of its genes uncharacterised. An often used approach to assigning a function to an uncharacterised gene sequence is to transfer the function of the gene with the most sequence similarity. However, if the similarity is too low, or the most similar gene is also poorly characterised, then this approach fails. As outlined in the previous chapter, computational methods show promise in identifying the function of a gene, especially via comparative genomics. Several methods, especially those outlined in Section 1.6.4, indicate that it is possible to predict a specific function for certain genes, using other information in addition to sequence homology. However, each has at least one of the following drawbacks:

- The criteria for grouping genomes or genes are too vague, or poorly defined (Huynen *et al.*, 1998).

- Homologues are computed using COGs which currently provide poor coverage of the available sequenced genomes (Levesque *et al.*, 2003; Reichard & Kaufmann, 2003).

- The software provided by the developers is difficult to use for others, either being a set of scripts, requiring a commercial program, or limited to a single operating system (Levesque *et al.*, 2003; Reichard & Kaufmann, 2003).

The methods described in this chapter, and implemented in the tool GeneTrawler, builds on previous research (Forterre, 2002; Huynen *et al.*, 1998; Levesque *et al.*, 2003; Pellegrini *et al.*, 1999; Tamames *et al.*, 1997), and corrects or improves on the above shortcomings. GeneTrawler is a tool for predicting the function of bacterial genes, using only homology and phenotype information–it does not rely on previous annotation or knowledge of gene sequences or any other database derived information. It is available either as a stand-alone program or via the Internet[1].

The structure of this chapter is as follows: the method for gene-function prediction is described, starting with a discussion on the reasoning and assumptions behind the method, followed by a description of the input data required. The implementation in a web tool called GeneTrawler is then described.

2.1 Prokaryote gene-function prediction

In the following discussion, the term ‘phenotype’ means a specific biochemical function or measurable trait that is observed in a prokaryote species. For example, the ability to metabolize sulfur, a physical organelle such as flagella, or antibiotic resistance are all phenotypes.

To predict gene function using the information from some phenomenon, that phenomenon must in some way influence or be influenced by the gene(s). To put it another way, to use one signal to infer another, there must be some interaction between the signals. One signal is the gene sequence. The other signal here is phenotypic information, in prokaryotes mostly a direct result of a gene sequence.

In the work discussed here, one of the most important differences between eukaryotes and prokaryotes is the different selection pressure on genome size (see Section 1.3.2). Because bacteria lose genes relatively quickly, only those genes under sufficient positive selection pressure are retained. This difference in selection pressure is exploited to predict gene function. Assuming that if in some genome[2] all the genes have an unknown function, simply the presence of a gene obviously doesn’t tell us anything useful. However, if we compare one genome against another, quite a bit of information becomes available. For instance, if there are no homologous genes between two genomes, then we can say with confidence that not only are the two genomes not closely related, but they also probably do not share the same biotope, or habitat (as no horizontal gene transfer is observed).

If we are armed with extra information about the species, we can make some more inferences. If, for example, we know that some species share some trait or function, which we will denote by Θ, and others do not have this trait, we can become more specific in our predictions. To see what sort of predictions might be possible, a brief discussion about the two different categories of bacterial genes is necessary.

The genes in a bacterial genome can be loosely divided into two sets: operational genes and informational genes (see Section 1.3.3). Informational genes are involved with DNA replication, repair, RNA synthesis, translation etc. These genes are more or less present in all members of a particular domain, as they are absolutely necessary and involved in many interrelated functions which makes the replacement or loss of one of these genes usually lethal. Operational genes, on the other hand, are involved in the more specific functions that may be present only within closely related members.

[1]http://bibiserv.techfak.uni-bielefeld.de/genetrawler/

[2]Unless otherwise stated, ’genome’ refers to ’bacterial genome.

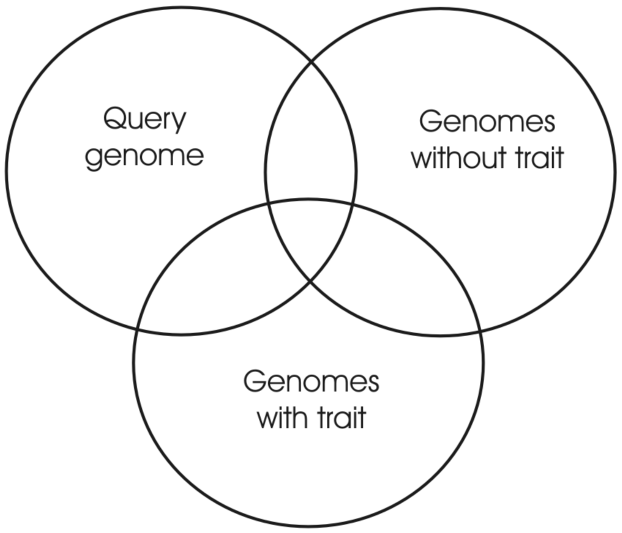


Figure 2.1: In the diagram above, the intersection of two circles represents common orthologues between genome(s). The goal is to find genes that are functionally involved in a specific phenotype. To find these genes, we find all the orthologues between the genomes that possess the phenotype, and exclude all the between the genomes **without** the phenotype. Because the orthologues are based on BLAST searches, one genome is the query and all others are database genomes. The area we are interested in is represented by a star, which contains orthologues present in only those species with the phenotype.

Now if we compare any two genomes, they will obviously share many informational genes, but whether or not they share any operational genes will depend on many factors including common selection pressures, time of divergence, ecological and genetic histories, and chance. However, assuming two species inhabit very different habitats after they have diverged, and do not exchange genes with each other via horizontal gene transfer, it will only be a matter of time before the number of operational genes shared is very small. If, however, they overlap somewhat in their ecology, lifestyle, or physiology, for example both requiring the function Θ (which could be for example the ability to fix atmospheric nitrogen), then the genes involved in Θ are going to remain in both species far longer.

If we assume that an arbitrary long period of time has passed, and that no genes are shared except the informational genes and the genes involved in Θ, then if we only compare those two genomes, knowing that both species possess the function Θ gives us no further information. However, if we include in the comparison a species that does *not* have Θ, the situation changes. We can now classify the genes into three groups: those shared only by those genomes with Θ, those shared by the genomes without Θ, and those genes shared by all. The genes shared by all are likely to be informational genes, and are of no interest here except for the fact we have identified them as such. The genes shared only by those species absent in Θ are of no interest, as there is no further information about these genes–they are likely operational genes, but no further information can be gained. Lastly, and most importantly, are those genes shared *only* by those species possessing Θ. Due to the evolutionary consequences of time previously discussed, these genes will very likely be involved with Θ. Thus, knowing only the presence or absence of genes and some phenotypic information, it is possible to acquire more information about the genes themselves (Figure 2.1).

Thus, we need two things for predicting gene function in this manner: the genes in common between genomes (orthologues) and a binary classification for each genome for a given phenotype. Before describing the algorithm for identifying gene function in further detail, these input requirements will be further discussed.

2.1.1 Phenotype data

Phenotypes are traits or characteristics that differentiate distinct species, or specify a biochemical function or ability. For example, some sample phenotypes are:

- Formation of flagella
- Aerobic metabolism
- Production of antibiotics
- Photosynthetic metabolism
- Methanogenesis

The phenotype information used to predict gene function is necessarily a binary classification. This means phenotypes have to be chosen or defined in such a way that biologically make sense and that a precise definition can be given. For example, the metabolic phenotype ‘Sulfur Metabolism’ is too broad, and can be more precisely defined by being split into three phenotypes; ‘Dissimilatory Reduction’, ‘Dissimilatory Oxidation’, and ‘Assimilatory Reduction’ (Rueckert, 2004).

For each phenotype, the data relevant to each species was gathered from Levesque *et al.* (2003), Bergey *et al.* (1984), expert users, and many Internet sources. For each datum, the source of the data is included and visible to the user in the tool (see example in Table 2.1). This is crucial because the binary phenotype classification is the most important parameter for function prediction here. Examining the source of data enables a user to eliminate genomes from the queries where the classification could be unreliable, or an error is suspected.

In other words, the phenotypes that could be used were yes/no type traits.

2.1.2 Orthologous genes

To describe how orthologous genes are computed, some definitions are in order. Often in this text it is written ‘genes in common’ for explanatory purposes. What is meant precisely are genes that are defined to be orthologous according to some algorithm. The definitions below are paraphrased from Eisen (1998). An example showing the difference between orthologues and paralogues is given in Figure 2.2.

Species	Phenotype	Value	Source
Agrobacterium tumefaciens (C58 Cereon)	Flagella	1	Genetic and environmental factors affecting T-pilin export and T-pilus biogenesis in relation to flagellation of Agrobacterium tumefaciens. Lai EM, Chesnokova O, Banta LM, Kado CI (2000).
Caulobacter crescentus	Flagella	1	Synthesis and Structure of Caulobacter crescentus Flagella. Lucille Shapiro and J. V. Maizel, Jr. (1973).
Streptococcus pneumoniae TIGR4	Gram Stain	1	http://www.ncl.ac.uk/dental/oralbiol/ oralenv/tutorials/gramstain.htm

Table 2.1: Sample phenotype data.

Homologue A gene related to a second gene by descent from a common ancestral DNA sequence. The term, homologue, may apply to the relationship between genes separated by the event of speciation (see orthologue) or to the relationship between genes separated by the event of genetic duplication (see paralogue).

Orthologues are genes in different species that evolved from a common ancestral gene in the course of speciation. Normally they retain the same function in the course of evolution. Identification of orthologues is critical for reliable prediction of gene function in newly sequenced genomes. (See also Paralogues.).

Paralogue Paralogues are genes related by duplication within a genome. Orthologues normally retain the same function in the course of evolution; paralogues evolve new functions (even though these may be related to the original one).

Xenologue A homologue that has diverged from its ancestor after a *horizontal gene transfer* event (e.g. antibiotic resistance genes in bacteria).

Positional homology Common ancestry of specific amino acids or nucleotide positions in different genes.

Two available methods for comprehensively determining orthologous genes are looking up the COG database (Tatusov *et al.*, 1997) and using bi-directional best hits from an alignment program such as BLAST (Altschul *et al.*, 1997).

The COG database groups clusters of genes deemed to exhibit sufficient sequence similarity to be orthologous. The COG database contains only a representative set of available genomes and is updated approximately annually (Tatusov *et al.*, 1997, 2000, 2001, 2003) as the process requires manual validation. At the time of writing (March 2004), there were 41 different bacterial species (42 including multiple strains) present in the COG database, compared to 124 (155

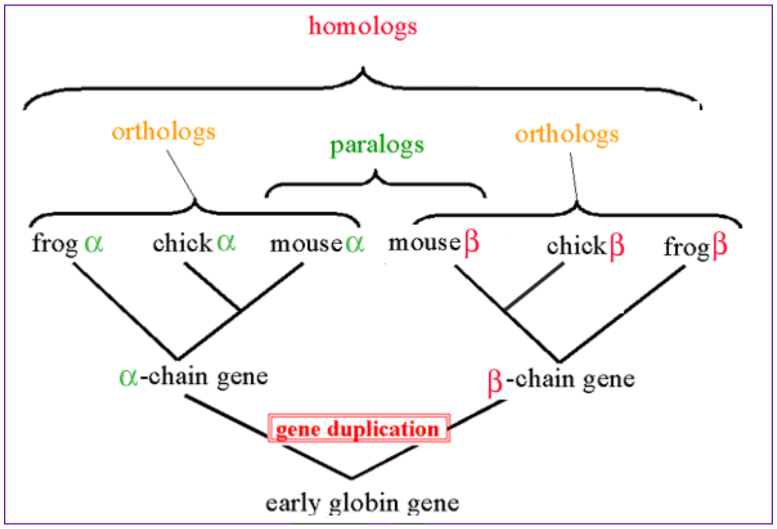


Figure 2.2: **Homologous sequences**. Orthologues and paralogues are two types of homologous sequences. Homologues are genes in different species that derive from a common ancestor. Homologues may or may not have the same function. Paralogues are homologous genes that diverged after gene duplication (Figure from http://www.ncbi.nlm.nih.gov/Education/BLASTinfo/Orthology.html).

including multiple strains) publicly available complete bacterial genomes. Even when the COG database is updated, the manual intervention required means that the number of genomes present in the COG database will always be considerably less than the total number of available genomes. Thus, newly available genomes are not available for incorporation into gene-function prediction methods that rely on the COG database.

The technique of bi-directional BLAST hits for orthologue determination is a simple and fully automated process. For each gene in one genome, a BLAST search is performed against all the genes of one other genome. If two genes from different genomes are each other's best BLAST hit, then they are classified as orthologues. This process suffers from the problem of false positives, however, as gene duplication followed by gene loss can result in the false classification of homologues as orthologues, when really the genes are (maybe distant) paralogues.

The need to include as many genomes as possible because of the scarcity of good phenotype data meant that the method of bi-directional BLAST hits was chosen to supply putative orthologues.

2.1.3 Gene prediction algorithm description

So far, orthologue and phenotypic classifications have been determined. Next, the algorithm that predicts gene function is described. The predictions made are "high level", in that the exact function of a gene cannot be predicted, only that a gene is functionally involved in creating a given phenotype.

Almost all gene-function prediction algorithms use some form of gene orthologue information, and the algorithm presented here is no exception. If the orthologues are determined by bi-directional best BLAST hits, then this leads to a problem of higher false positives than manually determining , e.g. via the COG database (Tatusov *et al.*, 1997). The algorithms in Levesque *et al.* (2003) gave poorer results when bi-directional best BLAST hits were used instead of COGs, except for the 'Similarity Measure' defined in Levesque *et al.* (2003) (see Chapter 3). A more flexible algorithm, called the Include/Exclude method, was developed to deal with the higher number of false positives.

In the Similarity Measure from Levesque *et al.* (2003), each gene is given a score based on its presence in other genomes. If this score is above some given threshold, the gene is included in the list of genes predicted to be functionally involved in the phenotype of interest. The score is given by a function of two variables, a gene h and a phenotype Θ, where $G_H = G_H(h)$ is a subset of G, the set of all genomes, and depends on h, comprising of all genomes with a homologue to h, and $G_T = G_T(\Theta) \leq G$ is the set of all genomes with the phenotype Θ. The score is given by the function:

$$f(h, \Theta) = \frac{\|G_H \cap G_T\|}{\|G_H \cup G_T\|}$$

If $f(h, \Theta) \geq X$, where X is some given threshold, then the gene h is regarded as being functionally involved in phenotype Θ.

The Include/Exclude method (used in GeneTrawler) works by checking two thresholds instead of one: given a phenotype Θ, for each gene h in the query genome one computes

$$\|G_H(h) \cap G_T(\Theta) \geq G_T(\Theta) - Inc\|$$

which indicates if enough genomes with the phenotype Θ also have an orthologue to h given a threshold Inc, and

$$\|G_H(h) \cap (G - G_T(\Theta)) \leq Exc\|$$

which indicates if enough genomes *without* the phenotype Θ *do not* have an orthologue to h, given a threshold Exc

By using two thresholds one could, for example, gather all those genes present in 6 out of 10 genomes with the phenotype, and absent in at least 10 genomes without the phenotype. The justification for modifying two thresholds separately is that bacterial genomic evolution is a story of exchanges, swaps, replacements, and general genetic tinkering. Relaxing, i.e. raising, the include threshold corresponds to allowing gene replacements for a phenotype. That is, some species may have replaced some genes required for a function with other, unrelated genes. In addition, some species may have lost a phenotype but adapted the genes previously used in that function into a different function. For example, flagella biosynthesis genes have in some species evolved into type III secretory systems which are used to inject toxins into the eukaryotic host (Blocker *et al.*, 2003). Relaxing, i.e. increasing, the exclude threshold allows the possibility that genes from the phenotype of interest have been adopted for different functions in some genomes. The practical difference between the two thresholds is discussed in Chapter 3.

The complexity of the algorithm is linear with respect to the number of genomes, however it depends on the preprocessing of the orthologues beforehand, which is also linear if only one species is being queried. If all genomes will be compared with each other then the complexity becomes $O(n^2)$, where n is the number of genomes. Despite being linear, if the orthologue data is not all loaded into RAM then queries can take approximately 10 seconds on a 500MHz CPU.

2.2 Implementation

This section details the implementation of GeneTrawler, the software tool written to perform the analyses described in the previous section. The goal was to build a tool that implemented the algorithm described, also integrating the phenotype and orthologue databases within a tool that could be used by the geneticist or biologist. In addition, the tool may need to be integrated into a larger genome analysis tool such as `GenDB` if it turned out to be useful, which was kept in mind when designing the tool.

The tools presented here are meant to be used in the wider community. Therefore, the following principles were established:

Maintain generality. The program was designed to be extendable to include more complicated analyses and/or other data sources, or to be integrated into a larger tool set in the future. For this reason the core system was designed to be simple and modular.

Open source. Users can download the source code and modify and extend the basic system. In addition, by releasing the source code previously unnoticed bugs may be found.

Portability. To ensure that users can download and install the system themselves with a minimum of problems, the code for GeneTrawler was written in Java, which can run on most operating systems without compilation. In addition, database servers and external libraries used by GeneTrawler are all open source and platform independent.

Web availability. As the installation is not trivial, and the databases are quite large, Genetrawler can also be accessed via the Internet at

http://bibiserv.techfak.uni-bielefeld.de/genetrawler/

2.2.1 Databases

The database of orthologues, genomes, and associated phenotype data is the backbone of GeneTrawler. As one of the previously stated goals was to make the system easily distributable, a publicly available open-source database was required that was simple to install and at the same time powerful enough for our purposes. MySql[3] was used to test the programs. Due to administrative issues, the database was migrated to PostgreSQL, another open source database.

2.2.1.1 Phenotype database

The phenotype or trait data is crucial for GeneTrawler to return accurate results, as a few wrongly categorised genomes can introduce substantial errors into the results when the sample of genomes

[3]http://www.mysql.com/

is small. For this reason, every datum describing an organism's phenotypic trait has additionally the source of that datum and the entry date recorded. This is for a variety of reasons: 1) In GeneTrawler, the quality of the phenotypic data is of more importance than the accuracy of the orthologue classification in predicting gene function (see Chapter 3). 2) The orthologues are acquired via automated BLAST searches, but the phenotype data has to be manually recovered from textbooks, Internet sources etc. This is prone to errors, and also is usually not complete. 3) The actual data may be based on flawed or incorrect experiments or observations, for example erroneous lab tests, or incorrect identification of the species, or other mundane occurrences.

Initially, the phenotype data from Levesque *et al.* (2003) was parsed and downloaded into the database. According to the authors, this was manually validated data, however a substantial number of errors were still found. This meant that much of the data used in the analyses presented here had to be checked again.

Additional data was gathered from a variety of sources, including the Internet, microbiology reference manuals such as Bergey *et al.* (1984), and scientific experts. When the source of the information was not somehow official, e.g. refereed papers or textbooks, attempts were made to find multiple sources.

It is envisioned that the phenotype data will be added to and edited by expert users. For this reason, the facility to add and edit was made available in the online version of GeneTrawler.

2.2.1.2 Orthologue database

As of March 2004, all prokaryote (archaea & eubacteria) genomes in the Kegg database (Kanehisa & Goto, 2000) were indexed in the GeneTrawler database. Protein sequences were used rather than DNA sequences as sequence conservation is higher at the amino acid level than the nucleotide level.

For each gene in a given genome, the orthologue (if any) in all other genomes was required. Bi-directional best hits in BLAST are determined for each gene; e.g. if BSU012 has a best hit (is most similar to) KAN325, and KAN325 has a best hit BSU012, the pair is considered a bi-directional best hit (putative orthologue). All other homologues are excluded. This prevents one gene from being related to multiple, less similar, genes in the other genome.

BLAST version 2.2.6 was used with default settings. The raw BLAST results were parsed with the BioJava (Mangalam, 2002) library and added to the `Genetrawler` database. From there, the orthologues were computed and indexed to each genome.

2.2.2 Program structure

For the standalone program, the user interacts with the program which connects directly to the phenotype and orthologue database. This results in essentially instantaneous query results. The web version has a script to interpret the HTML form commands into database queries and then sends the results back as an HTML page that is viewed in a new window in the web browser (see Figure 2.3).

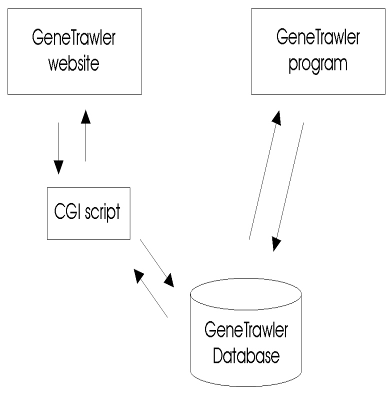


Figure 2.3: This diagram shows the essential difference between the stand-alone version of GeneTrawler and the web based version, where the web based version has an extra interface layer, which results in slower performance.

2.2.2.1 Stand-alone application

A stand-alone GUI [4] to the databases was written in Java. When a genome and phenotype is selected, the application loads all relevant data in the computer RAM which results in essentially instant computations (compared with about 10 secs per query from the web interface). The speed of computations is an important difference, as a user often needs to compare the results from incrementally changing a parameter: having instant results enables one to quickly gain an overview of a parameter effect.

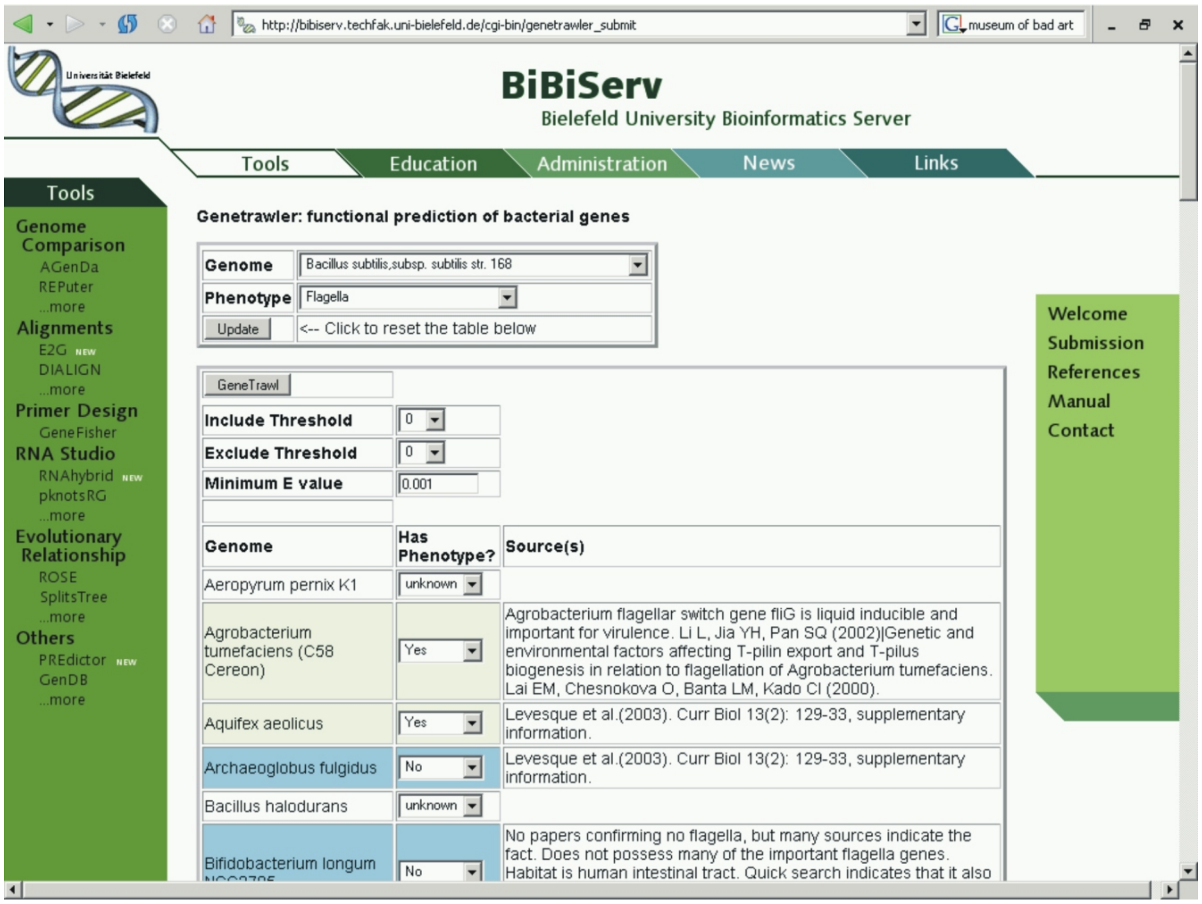
The application used only open-source libraries: BioJava (Mangalam, 2002) and third party open source database connection libraries.

2.2.2.2 Web interface

The installation of the stand-alone version of GeneTrawler includes not only the application, but also the associated non-trivial database installation. Because of potential technical difficulties for end users, a web interface was also built, enabling the same queries to be made via any Internet connection.

In the stand-alone application (SAA), all phenotype data was loaded in memory at once, enabling instant queries. This was not possible over the Internet without using Java Applets. However, Java applets proved too slow. A regular HTML form-based interface was then constructed, using a Perl script as the bridge between the HTML form requests and the GeneTrawler database. The database queries were modified so that all the processing occurred in the database and Perl script, as the client machine could perform no processing. However, the results could not be stored in between requests like the SAA, resulting in repeated subqueries and reducing the speed of the query. However, the query processing time by the server is under 10 seconds.

[4] Graphical User Interface.



Genetrawler: functional prediction of bacterial genes

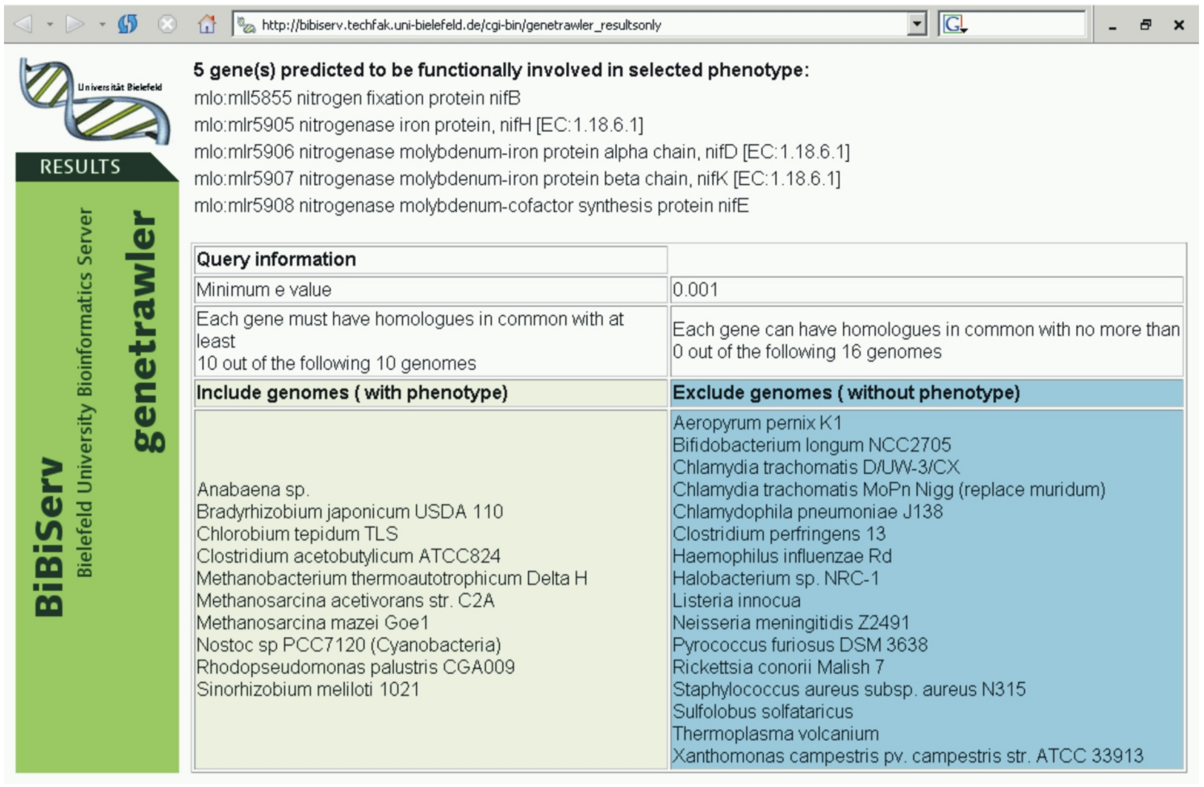
Genome	Bacillus subtilis,subsp. subtilis str. 168
Phenotype	Flagella
Update	<-- Click to reset the table below

GeneTrawl

Include Threshold	0
Exclude Threshold	0
Minimum E value	0.001

Genome	**Has Phenotype?**	**Source(s)**
Aeropyrum pernix K1	unknown	
Agrobacterium tumefaciens (C58 Cereon)	Yes	Agrobacterium flagellar switch gene fliG is liquid inducible and important for virulence. Li L, Jia YH, Pan SQ (2002)|Genetic and environmental factors affecting T-pilin export and T-pilus biogenesis in relation to flagellation of Agrobacterium tumefaciens. Lai EM, Chesnokova O, Banta LM, Kado CI (2000).
Aquifex aeolicus	Yes	Levesque et al.(2003). Curr Biol 13(2): 129-33, supplementary information.
Archaeoglobus fulgidus	No	Levesque et al.(2003). Curr Biol 13(2): 129-33, supplementary information.
Bacillus halodurans	unknown	
Bifidobacterium longum	No	No papers confirming no flagella, but many sources indicate the fact. Does not possess many of the important flagella genes. Habitat is human intestinal tract. Quick search indicates that it also

Figure 2.4: A screenshot of the GeneTrawler query form, showing the query genome (top box, *Bacillus subtilus*), the chosen phenotype (Flagella), the thresholds, and the genomes to compare with associated phenotype classification (bottom box).


http://bibiserv.techfak.uni-bielefeld.de/cgi-bin/genetrawler_resultsonly
Universität Bielefeld
RESULTS
BiBiServ
Bielefeld University Bioinformatics Server
genetrawler

5 gene(s) predicted to be functionally involved in selected phenotype:
mlo:mll5855 nitrogen fixation protein nifB
mlo:mlr5905 nitrogenase iron protein, nifH [EC:1.18.6.1]
mlo:mlr5906 nitrogenase molybdenum-iron protein alpha chain, nifD [EC:1.18.6.1]
mlo:mlr5907 nitrogenase molybdenum-iron protein beta chain, nifK [EC:1.18.6.1]
mlo:mlr5908 nitrogenase molybdenum-cofactor synthesis protein nifE

Query information	
Minimum e value	0.001
Each gene must have homologues in common with at least 10 out of the following 10 genomes	Each gene can have homologues in common with no more than 0 out of the following 16 genomes
Include genomes (with phenotype)	**Exclude genomes (without phenotype)**
Anabaena sp. Bradyrhizobium japonicum USDA 110 Chlorobium tepidum TLS Clostridium acetobutylicum ATCC824 Methanobacterium thermoautotrophicum Delta H Methanosarcina acetivorans str. C2A Methanosarcina mazei Goe1 Nostoc sp PCC7120 (Cyanobacteria) Rhodopseudomonas palustris CGA009 Sinorhizobium meliloti 1021	Aeropyrum pernix K1 Bifidobacterium longum NCC2705 Chlamydia trachomatis D/UW-3/CX Chlamydia trachomatis MoPn Nigg (replace muridum) Chlamydophila pneumoniae J138 Clostridium perfringens 13 Haemophilus influenzae Rd Halobacterium sp. NRC-1 Listeria innocua Neisseria meningitidis Z2491 Pyrococcus furiosus DSM 3638 Rickettsia conorii Malish 7 Staphylococcus aureus subsp. aureus N315 Sulfolobus solfataricus Thermoplasma volcanium Xanthomonas campestris pv. campestris str. ATCC 33913

Figure 2.5: A screenshot of a the results from a query in GeneTrawler. At the top are 5 genes predicted to be associated with the phenotype, while below is the query information (genome classifications, thresholds, etc).

Chapter 3

GeneTrawler: Results

This chapter presents the results of the GeneTrawler system (discussed in the previous chapter) when tested against known phenotypes and genes. Firstly, in Section 3.1 the effects of adjusting and optimising the the parameters is discussed. Section 3.2 presents the comparison of the algorithm developed in the course of this thesis (Include/Exclude method) with a similar algorithm (Similarity method) in Levesque *et al.* (2003). Section 3.3 details the analysis of the Include/Exclude method applied to four different phenotypes. From this detailed investigation, the results are summarized, and future work is outlined.

3.1 Parameters

The three main parameters in GeneTrawler are the Include and Exclude thresholds that determine the stringency of genome-set membership, and the minimum e-value for determining orthologues.

Thresholds

The Include/Exclude method has two discrimination thresholds that can be independently adjusted. It was found that to return accurate predictions, only the Include threshold (Inc) should vary, while the Exclude threshold (Exc) needed to remain relatively stringent (i.e., low). This means that, rather than keeping the orthologues from species with the phenotype, it seems more useful to exclude the genes shared with the species without the phenotype.

Practically speaking, it was observed that the Exclude threshold should be either 0 or 1 for the phenotypes and genome sets examined in this work (see the Appendix for the genome listings). The Include threshold is then gradually relaxed. This procedure produces the lowest fraction of false positives in almost all result sets (data not shown).

One reason for this is that in using bi-directional best BLAST hits to assign putative orthologues, some false assignments are produced. This results in some genomes by chance being falsely classified as possessing an orthologue. However, if more species are included in the analysis, especially species classified as **not** having the phenotype, this kind of error is reduced, as the likelihood of the same gene falsely classified as a homologue in multiple genomes decreases as the number of genomes added to the analysis increases. Another factor could be the nature of genome evolution. Some genes involved in specific functions can be replaced over time. However, if a genome does not have a phenotype it is less likely to have co-opted genes from that phenotype, via horizontal gene transfer for example. This is because genes involved in a function usually need to be transferred together, as individual genes are less likely to perform some useful function. This shortfall seems to be alleviated by the addition of more phenotype data. It must be noted that the `Include/Exclude` method still requires a minimum of about 8 genomes with the phenotype in question (i.e. include genomes).

e value

The precision of GeneTrawler was highest when only bi-directional best BLAST hits were used with a minimum e value of between $0.01 - 0.0001$, as this limited erroneous matches. There are other BLAST attributes that also could have been used to limit the number of false orthologue assignments, such as percentage of sequence identity or sequence lengths. These were tested, but gave the same or worse results than the e-value threshold, even in combination.

3.2 Comparison of algorithms

The Similarity Measure from Levesque *et al.* (2003) (see Section 2.1.3 in the previous chapter) and the Include/Exclude method used in Genetrawler were compared.

Comparison measures

The prediction of gene function is a binary classification, classifying the members of a given set of objects into two groups on the basis of whether they have some property or not. In this case, the property is *'functionally involved in a given phenotype'*.

To measure the performance of a classification system, the concepts of sensitivity and selectivity are useful:.

$$\texttt{sensitivity} = \frac{\texttt{true positives}}{\texttt{true positives + false negatives}}$$

$$\texttt{selectivity} = \frac{\texttt{true negatives}}{\texttt{true negatives + false positives}}$$

ROC curves are used to evaluate the results of a prediction by plotting the sensitivity and selectivity as some discrimination threshold is varied. Developed in the area of signal processing, the technique has proved to be very useful in the biological and biomedical sciences (Zweig & Campbell, 1993).

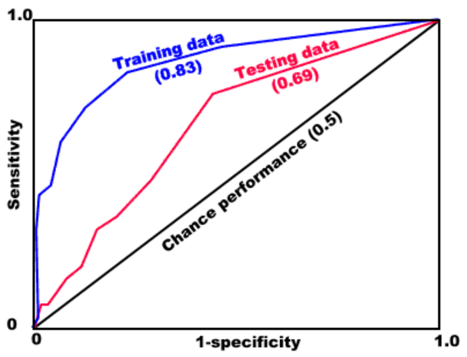


Figure 3.1: Example ROC plot. As a parameter is changed, the sensitivity and selectivity values are plotted. The area between the ROC curve and the no-discrimination line gives a single number that estimates the discrimination. The red line shows the method being evaluated applied to test data, the blue line shows the method applied to real data. A steeper slope implies a clearer discrimination.

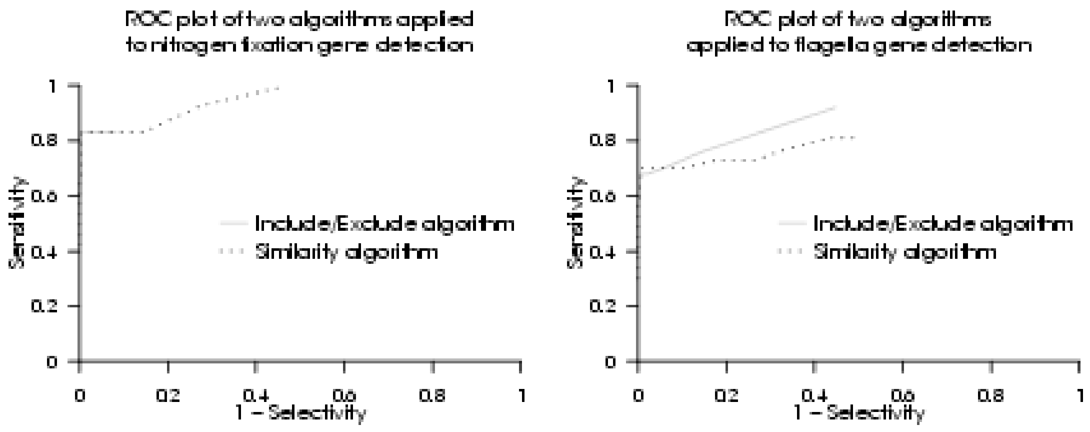


Figure 3.2: ROC plots comparing the Similarity Measure and Include/Exclude methods. The comparison is inconclusive because of lack of coverage and different parameters altered (Similarity Measure has one parameter, Include/Exclude has 2).

ROC analysis

However, plotting ROC curves from results from Genetrawler proved to be not very informative. For the same phenotype, the plots did not have enough resolution to show a difference between algorithms, even though the difference for some phenotypes was substantial (see results below). In addition, the algorithms did not use the same parameters (Similarity Measure has one parameter, Include/Exclude has 2). Interpreting differences from comparisons between phenotypes was difficult as there were multiple variables that also changed between phenotypes, such as the species included in the analysis, the numbers of genes, and how well the genes have been studied and annotated.

	Include/Exclude	Similarity Measure
Sensitivity	Fraction false	Fraction false
0.54	0.13	0.05
0.57	0.13	0.16
0.7	0.9	0.54

Table 3.1: Comparison of the Similarity Method and the Include/Exclude method for the Flagella phenotype. For equal sensitivity scores, the fraction of genes falsely predicted as positives is compared. At more relaxed thresholds, the Include/Exclude method shows a higher fraction of falsely predicted hits.

	Include/Exclude	Similarity Measure
Sensitivity	Fraction false	Fraction false
0.42	0	0
0.67	0.47	0.53
0.83	0.55	0.8

Table 3.2: Comparison of the Similarity Method and the Include/Exclude method for the Nitrogen Fixation phenotype. For equal sensitivity scores, the fraction of genes falsely predicted as positives is compared. At more relaxed thresholds, the Similarity Measure shows a higher fraction of falsely predicted hits.

Include/Exclude and Similarity Measure comparison

From a detailed examination of the results over multiple phenotypes, two measures proved useful in comparisons: sensitivity and the fraction of false positives

$$\texttt{fraction of false positives} = \frac{\texttt{false positives}}{\texttt{false positives + true positives}}$$

To compare two different algorithms, the fraction of false positives were compared for those values where the sensitivity was equal, e.g. Table 3.1. For the phenotype Photosynthesis, not enough genes were detected due to problems with the phenotype classification (see Section 3.3.3). For the phenotype Sulfur Reduction, the number of known genes was small and both algorithms performed similarly. For the phenotype Flagella, the Similarity Measure performed better, and for the Nitrogen Fixation phenotype the Include/Exclude algorithm performed better. At high sensitivity, the Include/Exclude method fares worse than the Similarity Measure for the Flagella phenotype, but better with the Nitrogen Fixation phenotype (see Tables 3.1 and 3.2).

For the Nitrogen Fixation phenotype, the `Include/Exclude` method performed substantially better than the Similarity Measure: for the detection of 10 out of 12 true positives, only around half the hits were false positives for the `Include/Exclude` method, compared to 80% false positives for the `Similarity Measure`. For the Flagella phenotype, the Similarity Measure retained a lower fraction of false positives as sensitivity was lowered.

3.3 Results: Phenotypes

In this section, four phenotypes are examined with respect to the ability of Genetrawler to detect genes specific for the given phenotype. Although annotation information was not used to predict the function of genes, it was used to validate predictions. For example, the set of genes known to play a role in flagella was defined as the set of genes with the word 'flagella' (or a derived word such as 'flagellin') in the annotation. This set of genes corresponded to the set of true positives. In the examples below, a parameter is incremented and the numbers of genes returned by the algorithm is noted, including the true positives. A comment for each phenotype then follows.

3.3.1 Flagella

The bacterial flagellum is a readily identifiable and definable trait, and identifying previously unknown flagella genes has already been used as a proof of principle (Levesque *et al.*, 2003).

Genes and genomes

For the set of true positives, a `grep`[1] search was performed on all protein sequences from the *Bacillus subtilis* genome downloaded from Kanehisa & Goto (2000), using the regular expression (`.*flag.*`). This returned all annotations with the words flagella, flagellar and flagellin, which was 37 gene annotations in total. The annotations were examined to confirm involvement in flagella function and regarded as the set of true positives.

The species included in analysis are listed in Table A1 in the Appendix.

Performance

GeneTrawler identified the majority of flagella genes, and indicated some genes worth further investigating. The table below shows the number of Flagella and other genes given the specified Include value. See Section 3.1 for a discussion of suitable thresholds.

Exclude value = 1

Include value	Flagella genes (of 37)	Total genes
0	15	15
1	19	20
2	20	22
3	20	30
4	24	61
5	24	145

Most ($>$50%) of the known flagella genes are returned when there are less than half false positives. When the Include value is three, there are 10 genes returned without a flagella annotation: these genes would be good candidates for further investigation. As not all the genes classed

[1]Grep is a pattern matching program that searches the lines of an input file for character patterns. http://www.gnu.org/software/grep/grep.html

as false positives neccesarily are false positives, it is possible that some genes have some as yet unknown role in flagella function (see Discussion). Such an investigation was performed for the phenotype Sulfur Reduction. All the genes whose direct role in sulfur reduction appeared to be involved in reactions in the periphery of the network.

Comments

Genetrawler could detect more that 50% of the known flagella genes with few false positives. This represents a useful level of accuracy, for example, if no genes had been previously identified, this is a useful number of genes for beginning to characterise the phenotype.

It must be noted that the false positives are putative, in that the genes may have some as yet unknown role in flagella. In fact, the genes without a flagella annotation that are returned by the algorithm when the stringency is high are excellent candidates for further investigation. However, further investigation requires an expert, and such analysis was only performed for one phenotype (Section 3.3.4), where such an expert was available.

A potential problem for these methods are genes that were previously involved in a function that have subsequently been co-opted for a completely different function, possibly followed by a loss of the original function. For flagella, this is especially relevant. During the construction of the flagella, protein subunits are secreted into the periplasm. The secretory apparatus, involving a type III secretory system, has been co-opted by bacterial pathogens to inject toxins into their eukaryotic hosts (Blocker *et al.*, 2003). Thus, there are many species who do not have flagella, but possess a subset of the genes associated with flagella. This is one reason why approximately a third of the flagella genes could not be detected by GeneTrawler.

3.3.2 Nitrogen fixation

Genes and Genomes

The query organism chosen was *3.2Mesorhizobium loti*. The core nitrogen fixation pathway in *M. loti* consists of 12 genes, which were taken from the Kegg pathway database (Kanehisa & Goto, 2000). These were used as true positives:

mll5837 nif-specific regulatory protein [KO:K02584]
msl5852 hypothetical protein(fixU protein) [KO:K02593]
mll5854 hypothetical protein(nifZ protein) [KO:K02597]
mll5855 nitrogen fixation protein nifB [KO:K02585]
mll5857 nif-specific regulatory protein,nifA [KO:K02584]
mll5864 nitrogenase stabilizer NifW [KO:K02595]
mlr5905 nitrogenase iron protein, nifH [EC:1.18.6.1] [KO:K02588]
mlr5906 nitrogenase molybdenum-iron protein alpha chain, nifD [EC:1.18.6.1] [KO:K02586]
mlr5907 nitrogenase molybdenum-iron protein beta chain, nifK [EC:1.18.6.1] [KO:K02591]
mlr5908 nitrogenase molybdenum-cofactor synthesis protein nifE [KO:K02587]
mlr5909 nitrogenase molybdenum-iron protein beta chain, nifN [KO:K02592]
mlr5911 nitrogenase molybdenum-iron protein nifX [KO:K02596]

Exclude=0, max e=0.01

Include value	Nitrogen genes (of 12)	Total genes
0	5	5
1	5	5
2	5	5
3	5	5
4	5	6
5	8	15
6	10	22*
7	10	78

Table 3.3: GeneTrawler results for the phenotype *nitrogen fixation*. The species in the analysis are from Table A2 in the Appendix. The Exclude value is kept at 0 while the Include value is incremented. As the Include value is relaxed, more nitrogen-fixation genes are detected, but also more probable non-nitrogen fixation genes. The star (*) on row seven indicates the best results, with 10/12 nitrogen fixation genes detected, with only 12 other (putatively false positive) genes.

At the time of writing (November 2004), there were 8 other bacterial species with the ability to fixate nitrogen. The ability to fix nitrogen is usually easy to determine from the habitat and lifestyle of the organism, as it is a highly specialized trait. This specialization also means that the genes involved in this phenotype have not been adapted for some other non-nitrogen related function, as far as this author is aware. For the list of species included in the analysis see Table A2 in the Appendix.

Performance

GeneTrawler could detect 5 nitrogen fixation genes (out of 12) with no false positives. Detecting 10 true positives resulted in about 50% false positives (see Table 3.3). There are many other genes involved in the periphery of nitrogen fixation, however none of those genes were detected. See Section 3.1 for a discussion of suitable thresholds.

Comments

The detection rate for this phenotype was relatively high. A factor in this is likely the biology and evolutionary history of nitrogen fixation: as the genes for nitrogen fixation are so specific, it is less likely that have been co-opted for other biochemical functions, thus less likely that species exist with orthologues to these genes yet without the phenotype.

The two genes not detected were nitrogen fixation specific regulatory proteins (*mll5837* nif-specific regulatory protein and *mll5857* nif-specific regulatory protein, nifA). Regulatory sequences are less likely to be detected with these methods because they are not under the same constraints as genes involved in a more specific biochemical reaction, such as nitrogen fixation. The proteins for such a primary function as nitrogen fixation are highly optimized and therefore constrained, whereas the regulation of those genes cannot easily be ’optimized’, as regulation is often complex with subtle interactions by many other genes, such as transcription factors.

3.3.3 Photosynthesis

Genes and Genomes

Synechocystis sp., a member of the Cyanobacteria group, was chosen as the query genome. From the Kegg pathway database (Kanehisa & Goto, 2000), 65 genes were identified as parts of the photosynthetic machinery, and used as known true positives.

Seven genomes were known to be photosynthetic, including five in the taxonomic group Cyanobacteria, one Alphaproteobacteria, and one Green Sulfur bacteria. As with nitrogen fixation, photosynthesis is an easily detected trait, and accurately classifying species as non-photosynthetic was not difficult, compared with other phenotypes, such as flagella or sulfur reduction. The list of species in the analysis is in the Appendix, Table A3.

Performance

Initially, almost no photosynthesis genes were detected, despite very relaxed thresholds, until the two non-Cyanobacteria were removed from the list of genomes possessing photosynthesis (hence no table is shown as in the earlier phenotypes). After their removal, 25 out of 65 true positives were detected, but with 85 additional false negatives.

Comments

Photosynthesis evolved billions of years ago and, since then, horizontal gene transfer has played an important role in the evolution of this phenotype in different species (Raymond *et al.*, 2002). From phylogenetic studies, it is postulated that oxygenic photosynthesis (which produces oxygen) evolved later that anoxygenic photosynthesis (Xiong *et al.*, 2000). Cyanobacteria use oxygenic photosynthesis while the two bacteria that were removed from the list (*Chlorobium tepidum* and *Rhodopseudomonas palustris*) use non-oxygenic photosynthesis mechanisms (Eisen *et al.*, 2002; Larimer *et al.*, 2004) and are phylogenetically very distant. This is reflected in the results, in that photosynthesis genes could only be detected when species in the same phylogenetic group only were used. The poor ability to predict photosynthesis genes might be improved when the genomes from a more representative sample of photosynthetic bacteria can be included. However, while the addition may improve the precision, it would probably not predict more photosynthetic genes, as many may have been substituted in different species over the course of evolution (Raymond *et al.*, 2002).

3.3.4 Assimilatory Sulfur Reduction

For this phenotype, an expert was available (Rueckert, 2004) which enabled a more detailed and comprehensive study.

From Christian Rueckert (Rueckert, 2004)	From Kegg database (Kanehisa & Goto, 2000)
Known	
NCgl2713	
NCgl2714	
NCgl2715	NCgl2715
NCgl2716	NCgl2716
NCgl2717	NCgl2717
NCgl2718	NCgl2718
NCgl1923	
Suspected	
NCgl0328	
NCgl2350-2353	
NCgl2462/2463	
NCgl2658	
NCgl2719	
NCgl2940-2943	

Table 3.4: Assimilatory Sulfur Reduction genes from *C. glutamicum*

Genes and Genomes

Rueckert (2004) studied the pathway in a specific species, *C. glutamicum*. Therefore, this species was chosen as the query genome. The genes from *C. glutamicum* that were involved in assimilatory sulfur reduction were compiled by Rueckert (2004), coming from personal knowledge and numerous literature sources. In addition, the genes from the Kegg pathway database (Kanehisa & Goto, 2000) were also gathered for comparison purposes (see Table 3.4).

From Rueckert (2004), a list of species was compiled exhibiting assimilatory sulfur reduction (see Table A4 in the Appendix).

Performance

GeneTrawler retrieved the genes involved in the core pathway specified by Kegg under stringent conditions[2]: NCgl2715 - NCgl2718. The other genes were not detected even when the thresholds were very relaxed (hence no table is shown as in the earlier phenotypes). According to Rueckert (2004), the sulfur metabolism gene cluster in *C. glutamicum* is not conserved well, i.e. it is quite unique.

Comments

When the e-value=0.0001 threshold was raised and lowered, the core pathway from KEGG remained in the set of genes returned, however extra genes were also part of the result set, depending on the e-value threshold. As these genes were returned under the strictest Include and

[2]Include=0, Exclude=0, e-value=0.0001

Name	Annotation	Extra data from investigation
NCgl0290	NTP pyrophosphohydrolase	Performs oxidative repair mechanisms, involved in thioreduction, thus electron transfer or assembly.
NCgl2115	cytochrome C oxidase subunit II	Uses electron transer or assembly.
NCgl2471	hypothetical protein	Cofactor repair, cobalimine.
NCgl2283	pirin-related protein	Containes iron, therefore probably sulfur cluster.
NCgl0824	metalloendopeptidase-like membrane protein	No other info
NCgl0210	ABC-type transporter, permease component	Involved in molybdate metabolism.
NCgl1511	polyprenyltransferase	
NCgl1183	electron transfer flavoprotein alpha-subunit	Uses electron transer or assembly.
NCgl1903	FAD synthase	Uses electron transer or assembly.

Table 3.5: By varying only the e-value threshold, several interesting genes were returned with the core sulfur pathway. All of these genes shared one characteristic: they use metal-sulfur clusters to transfer electrons.

Exclude thresholds, and in preference to other genes known to be involved in the pathway, they were of interest as the sulfur reduction pathway is in no way fully characterised.

A list of these genes was compiled and investigated further, Table3.5. All of these genes shared a characteristic: they were either involved in the metabolism of metal-sulfur groups, or they required metal-sulfur groups for enzymatic activity. Metal-sulfur groups are used in reactions that involve electron transport, as sulfur is relatively easily oxidised and reduced. Iron is often the metal associated with the sulfur group, but genes involved in molybdate metabolism were also observed.

For example:

- NCgl0210 is involved in molybdate metabolism. Molybdenum is an essential trace metal required for the activity of some enzymes that are involved in catabolism, including the catabolism of purines and the sulfur amino acids.
- NCgl2283 is a pirin-related protein. Pirin is an iron-binding nuclear protein and transcription cofactor.
- NCgl2115 is a cytochrome C oxidase subunit. These use electron transer which requires iron-sulfur clusters.

This demonstrates that GeneTrawler can gather genes not only directly involved a a pathway, but either dependent on it or somehow involved in periphery reactions. This will be of considerable use to those who study pathways. Often gene knockouts are used to establish whether a gene is involved in a particular pathway, but sometimes the knockout is lethal or displays no effect. This method shows that a set of genes can be found that are involved indirectly in the pathway or are dependent on the pathway.

3.4 Discussion

The above results demonstrate that, given accurate phenotype data, it is possible to predict the function of bacterial genes without the use of any previous knowledge of those genes. This will be of particular use as the pace of genome sequencing accelerates. For three of the examined phenotypes, a large enough proportion of true positives was returned to demonstrate that the assumptions about the evolution of bacterial genomes hold. As the sulfur reduction phenotype demonstrated (Section 3.3.4), genes dependent on (or in the periphery of) the phenotype can also be detected (Figure 3.3). Dependent genes have previously been observed only after the reconstruction of metabolic pathways from (direct or indirect) experimental data.

Comparison of algorithms

The Include/Exclude method and the Similarity Measure gave almost identical results for two of the four phenotypes. The main difference was for the nitrogen fixation phenotype, where the amount of phenotype data was substantially larger than the other phenotypes and thus many more genomes could be included in the analysis. In fact, most genomes could easily be classified as not having the phenotype, leading to proportionally more exclude genomes (see Table A2 in the Appendix). The Include/Exclude method handles this kind of biased data set better, as two thresholds can be adjusted separately. As the Similarity Measure only has the one threshold, the two categories (include and exclude) are given equal weight when really one category is weighted more due to the extra data.

For this reason the Include/Exclude method is more useful for larger datasets, as in this analysis where orthologues are computed via bi-directional best BLAST hits rather than the COG (Tatusov *et al.*, 1997) database.

Comparison of phenotypes

The analysis of four different phenotypes demonstrated the capabilities and limitations of Gene-Trawler. In summary:

The definition of the phenotype is of primary importance. As the photosynthesis phenotype analysis demonstrated, two phenotypes may have the same general name and function, but are actually two different phenotypes, due to the geological time passed in which they have diverged, and the different physiochemical basis for the function. The genomes possessing the similar but different phenotype should be excluded from the analysis. Horizontal gene transfer will also be an issue in this case, as gene substitution will make the genes difficult to detect with these methods.

Knowledge about the evolution of the phenotype can help when a subset of the phenotype has been co-opted into another function, as in flagella genes being adapted into a secretory apparatus. By excluding the genomes with the co-opted trait, a bigger subset of genes can be found, although in this case determining which genomes have flagella but not type III secretory systems and *vice versa* is not trivial.

Certain genes involved in a function are more likely to be found than others. Regulatory genes are less likely, as regulatory genes are under less constraint than enzymatic genes. Indeed, shuffling of regulatory genes is required to alter the regulation of a function, or to incorporate a new function via HGT, which results in far less conservation for regulatory genes and sequences than for non-regulatory coding DNA. The result is that these genes are more likely to be missed by GeneTrawler.

The genes collected by GeneTrawler include not only genes directly involved in a function, but also genes in the periphery of the network or pathway. These are genes that require the function, or act in some auxiliary role. E.g. many genes were predicted for sulfur reduction that required the sulfur produced by the pathway in their own reactions, usually as electron transporters. This ability of GeneTrawler can be used to discover unknown biochemical reactions within a network that may have very subtle phenotypes from gene knockout experiments.

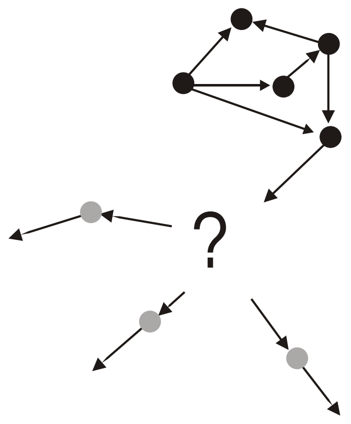


Figure 3.3: For the assimilatory sulfur reduction phenotype, genes were detected that are dependent on the assimilatory sulfur pathway, as all use metal-sulfur complexes as co-factors. Normally, pathways need to be reconstructed before this kind of information becomes available.

Future work

GeneTrawler is a useful tool, however it is only one tool that should be used in conjunction with many others. Because of this, the methods should be incorporated into a larger scale genome analysis tool, such as WIT (Overbeek *et al.*, 2000). This would simplify the process of discovery and hypothesis testing in a genomics scale framework.

The phenotype database shows the most promise. If this database grows from input from the biological community, then its value to these methods will correspondingly increase. Discussions are underway into the feasibility of text mining a resource such as Bergeys Manual of Systematic Bacteriology (Bergey *et al.*, 1984). This series of books already has much phenotypic data, however it is not easily accessible by computers and would require a large investment in resources to mine useful information.

Extending Genetrawler to eukaryotes

The next step in this work will be adding eukaryote genomes into Genetrawler. There are several theoretical reasons why including higher eukaryotes, such as multicellular eukaryotes, will not produce useful results. The main reason is that the algorithms presented here rely on a strong selection pressure against large genomes, and this does not appear to be an important factor in higher eukaryote genomes. However, yeast genomes are under this selection pressure. When investigating phenotypes, one will have to be careful when defining phenotypes, as the correspondence between prokaryotic phenotypes may not be clear, which may have negative consequences for gene-function prediction. Although this author doubts that including eukaryote genomes will produce useful results, it has not yet been tested.

Chapter 4

Gene network reconstruction: Motivation and Methodology

This chapter introduces the second approach to understanding gene regulatory networks. The previous two chapters tackled the problem of predicting gene function, and in our conceptual model (Figure 1.1) this represents elucidating the *nodes* of the network. In the next two chapters the focus is on the *edges*.

The advances of high-throughput molecular technologies have presented interesting possibilities for large-scale construction of biological models. DNA microarrays, or DNA chips, are one such recent technology that can measure the mRNA levels of thousands of genes simultaneously, providing a ‘global’ view of gene expression (Schena *et al.*, 1995). By measuring simultaneously the relative mRNA levels of all genes in a cell, a snapshot of the state of a cell is taken. By repeating this a number of times, a time series is produced which reflects the dynamics of gene-expression networks

There has been substantial interest in the possibility of reconstructing the network of gene regulatory interactions using data from such microarray experiments. This is called a *reverse problem*. The reverse problem has been approached in many different ways (Section 1.9), depending on the formalism used to model the gene regulatory network (GRN).

So far, previous methods have been tested on random networks. The main motivation for this work is to find out the limitation for this kind of approach, in terms of what kinds of networks can (and cannot) be reconstructed, using the time series of gene transcription levels (from simulated microarray experiments) to reconstruct the network of gene interactions. To this end, we take what we consider the best aspects of previous reconstruction attempts to develop a reconstruction pipeline, and apply this pipeline to different types of networks. We then compare how well the networks topology is reconstructed for different types of networks.

Due to the complexity of the model networks, an evolutionary approach is used to recover the topology of artificial gene regulatory networks (by topology we mean the set of edges for a given set of nodes). An overall strategy is used similar to approaches discussed in Section 1.9 (e.g. (Tominaga *et al.*, 1999; Akutsu *et al.*, 2000; Wahde & Hertz, 2000)) where an artificial network is constructed to simulate a microarray experiment, and then the network is reconstructed based on the data. The effectiveness of the algorithm can be determined by comparing the topology of the solution networks with the original target network.

The description is presented as a pipeline, with the details of each component of the pipeline discussed sequentially. Before the reconstruction pipeline is discussed, the mathematical model of gene regulation used here will first be described.

4.1 The model of gene regulation

A dynamic model of gene regulation is needed so that the expression levels of genes at different time points can be measured, as these measurements are then used by the evolutionary algorithm to evolve networks similar in topology to the original target network. Although reconstruction methods using directed/undirected graphs and Bayesian networks are popular, these models do not capture the network dynamics and are, then, unsuitable for our purpose of reconstruction using time-series data. On the other hand, stochastic models are too complicated, requiring too many parameters that need to be estimated (i.e. guessed) that cannot be measured experimentally or validated. Boolean models are dynamic as well as mathematically very tractable, however they possess several problems when applied to gene regulatory networks. Gene expression can spend a lot of time at intermediate values, rather than having binary values (Arkin, 2000). Concepts from control theory that seem important and applicable to gene networks, such as amplification of a signal, subtractions and addition of signals, and smoothly varying periodic behaviour, either cannot be implemented in Boolean networks or lead to very different behaviour (Arkin, 2000; Vilar *et al.*, 2002). Negative feedback is a common phenomenon in real gene circuits which has a stabilising effect (Becskei & Serrano, 2000). However, in Boolean networks the outcome of negative feedback is always oscillation. This problem is somewhat alleviated by the use of piecewise linear equations or logical equations, but these formalisms are not quite appropriate for quantitative modeling of large-scale expression data.

Ordinary differential equations were chosen as these have the advantage that the dynamics of the model can be simulated, and that the system can be encoded in a very simple and intuitive manner (e.g. Figure 1.16). Specifically, a non-linear regulation function was used, as all regulation functions must saturate. In addition, several rules for genetic circuitry require continuous methods (Savageau, 1998).

In our formalism, we simplify the biological process of gene regulation for tractability and simplicity. The nodes in the network represent genes coding for transcription factors. We assume that there is a direct relationship between transcription factors binding to the regulatory site of a gene and the rate of production of the corresponding gene product: in other words we ignore regulation at the mRNA transcript level. Transcription factors bind to specific promoter regions on the genome, causing up-regulation or inactivation/inhibition. In addition, we initially assume that all nodes in the network are measured which means that other non-transcribed effectors of regulation, such as metabolites, are currently ignored. Thus, the words ‘gene’, ‘gene product’, and ‘transcription factor’ are used interchangeably in regards to the model specified here, the nodes in the gene regulatory network representing all three.

This simplification is justified by two reasons: firstly, there is no nuclear membrane in prokaryotes, so transcription and translation can occur simultaneously. In eukaryotes, the picture is complicated by extensive mRNA processing and regulation. However, this study initially focuses on prokaryotes. Secondly, there is little justification for including many extra parameters that cannot be measured experimentally, as this would not only increase computation time but also complicate analysis. In further work, it is planned to model the unmeasured elements specifically (see Section 5.5).

The model of gene regulation presented here is based on Salazar-Ciudad *et al.* (2000). It is conceptually similar to most models discussed in Section 1.8.4 where genes and gene products were represented conceptually as a single node in the network.

Non-linear ordinary differential equations are used to model the dynamics of the regulatory interactions. A non-linear function was used because linear equations do not capture long-term dynamics as the validity of linear equations applies only in small neighbourhoods of given states (Stark *et al.*, 2003). Linear equations also do not display aspects important for regulation (Arkin, 2000). Consequently, the reconstruction method presented here is not limited to linear approximations like other methods (Chen *et al.*, 1999; D'Haeseleer *et al.*, 1999; de Hoon *et al.*, 2003).

The Hill function (Equation 1.6, previous chapter) was chosen over the sigmoidal transfer function as it has some biomolecular significance (Yagil & Yagil, 1971; Yagil, 1975; Hill, 1985) and is widely used to model molecular binding processes (Cornish-Bowden, 1979). A problem with the sigmoidal transfer function is that a gene with an incoming weight of zero will still show a positive expression (see Salazar-Ciudad *et al.* (2000) for a discussion).

The dynamics of the networks are specified by the equation

$$\frac{dx_i}{dt} = \frac{\Phi(h_i)}{\Phi(h_i) + \theta_i} - \mu x_i \tag{4.1}$$

where x_i is the concentration of gene product i, and $\Phi(h_i) = max(0, h_i)$ is the Heaviside function which ensures that negative weights do not lead to negative concentration values, h_i is the incoming regulatory influence (see below), and μ is a term representing gradual decay. For simplicity and tractability, we used the same degradation rate for all genes. The regulatory function ranges from 0 to 1, approaching 1 as h_i tends to infinity. The regulatory influence is calculated as described in Chapter 1, Equation 1.4:

$$h_i = \sum_j w_{ji} x_j \tag{4.2}$$

where w_{ji} measures the regulatory influence of gene j relative to gene i, and x_j is the concentration of gene product j. Thus, the gene regulatory network is described by a matrix such as in Figure 1.16. The random creation and mutation of gene regulatory networks is performed by creating and changing matrices.

4.2 Methodology Pipeline

The reconstruction pipeline consists of four steps:

1. Create a random gene regulatory network.
2. Simulate the gene expression data using this network and record the resulting artificial microarray data.
3. Evolve a population of gene regulatory networks to produce the same microarray results.
4. Compare network topologies.

Each of these steps is detailed below, see Figure 4.1 for an overview. A possible point of confusion is the fact that random networks are discussed in two separate places in this pipeline. It must be kept in mind that the first random network, created in Step one, is the *target* network and we wish to reconstruct the topology of this network. We then create a population of random networks (Step 3) and, using an evolution algorithm, we evolve them to have a similar microarray time series to the target network (and hopefully similar topology). The target network does not have to be random, but it is because we apply the pipeline to many networks to get some measure of confidence in the results, and generating many networks by hand is time consuming and unnecessary. The population of random networks of course must initially be random.

4.2.1 Step 1: Create a random target gene regulatory network

The 'target network' is the network that we wish to reconstruct or recover. First the actual model of gene regulation will be described, followed by the process of constructing a random gene regulatory network.

Generation of random gene regulatory networks (GRNs)

When the pipeline to recover network topology (described in Figure 4.1) is applied to real data (future work, not attempted in this work), we will not know the true structure of the network. But for development and testing purposes, and for comparing different reconstruction approaches, the structure of the target network must be known beforehand so it can be compared with the predicted networks produced by our method. The target network is randomly chosen because this whole pipeline is applied many times on many different target networks, so that differences can be detected in the predictive ability of the methods due to different network structures.

One possibility of creating artificial regulatory networks (ARNs) at random is to generate a bit string and mimic the process of gene regulation occurring from this artificial genome (Banzhaf, 2003). The network then emerges from the interaction of the components. In contrast to this, we generate the network topology directly.

Although initially it appears to be a trivial task to create a random target network, there is the complication that the randomly created networks should "make sense", i.e. they should exhibit dynamic behavior that is arbitrary, but reasonable in the domain they are created for. In this case, reasonable ARNs are those which do not stop expression at an early time or where activity is confined to only a small subset of the whole network. Therefore, we applied a method to prune and modify random networks to produce interesting dynamics:

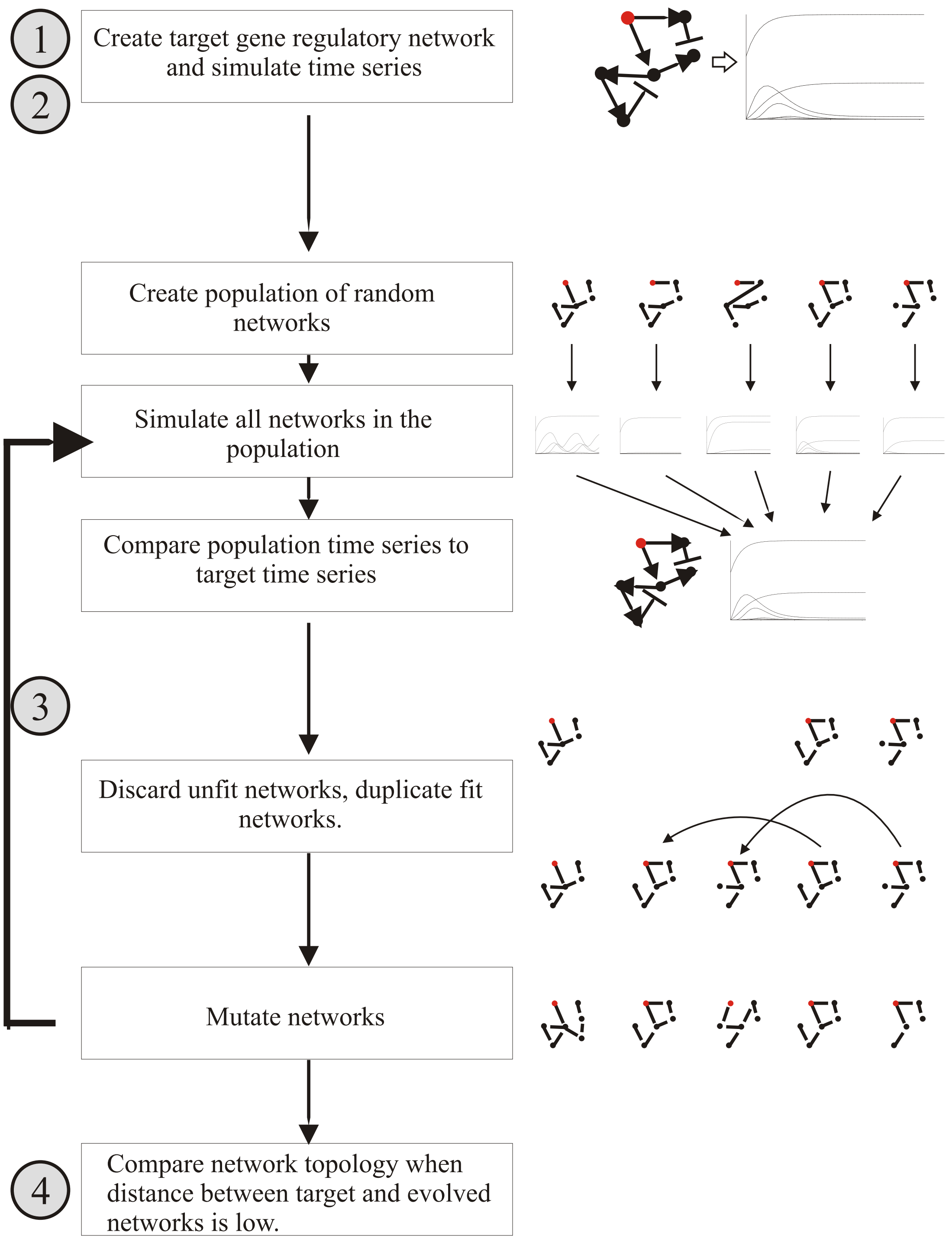

1
2
Create target gene regulatory network and simulate time series
Create population of random networks
Simulate all networks in the population
Compare population time series to target time series
3
Discard unfit networks, duplicate fit networks.
Mutate networks
4
Compare network topology when distance between target and evolved networks is low.

Figure 4.1: Pipeline for the reconstruction of gene regulatory networks using an evolutionary algorithm. The steps are as follows: 1) Create a random gene regulatory network. 2) Simulate the network and record microarray data. 3) Evolve a population of gene regulatory networks to produce the same microarray results. 4) Compare network topologies.

First, a directed random network is created by applying the Erdös-Renyi approach (Erdös & Renyi, 1960) to directed graphs: for a number of nodes each possible edge is tested and drawn if a random number is lower than a given probability p. Each edge is assigned a uniform random weight in the range of $[-1, 1]$. The node with the highest "out degree" is chosen as the *starting node* for the simulation (for the concept of a starting node see also Section 4.2.2). All nodes unreachable by the first node are connected at random to the reachable nodes (also chosen at random).

All nodes in the network are now reachable from the first node with respect to edge direction, but not necessarily with respect to reaction rates because negative reaction rates might inhibit the expression of complete subnetworks "behind" a node. Thus, a second pruning is performed considering the assigned edge weights (which serve later as reaction rates) and all nodes that are not reachable due to negative weights are simply deleted from the network (Figure 4.2).

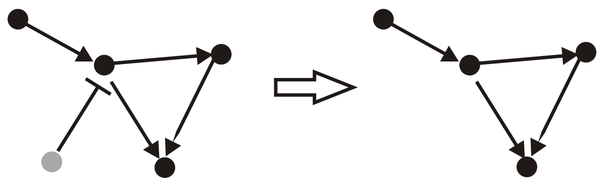


Figure 4.2: When generating the random target networks, nodes such as the grey node of the network on the left can never be activated as there is no positive connection from any other node, and therefore is deleted.

Removing unconnected nodes improves the efficiency of the simulations, as otherwise one would be required to take account of many genes that will never be activated.

The final network is represented as a matrix, W, such as Figure 1.16.

4.2.2 Step 2: Simulate network dynamics

The target network is simulated and the results of the network dynamics are recorded. This becomes the time series that represents an idealised microarray experiment, in that we capture dynamics at quite regular and short time intervals, and we do not introduce any error that is always present in real experimental work. The reason for this is simplicity: to evaluate the limitations of this technique (and, in the future, other techniques), the reconstruction process firstly assumes that there is no error in the results of the microarray experiments that produce the time series of gene expression. Once the capabilities and limitations of the method are determined, the contribution of error to reconstruction methods can be evaluated by introducing error into the simulated microarray experiments.

Simulation of microarray data has been performed by other researchers in a variety of ways and with different motivations. Repsilber & Kim (2003) start with an ARN and simulate the network dynamics for a number of time steps to build a time series. They then change the initial conditions of the ARN and simulate another time series. A simple model combines the two sets of expression values deriving a time series of log ratios. The danger with this approach is that any kind of averaging or combining of microarray results can obscure true differences in network dynamics that result from different starting conditions (Stark *et al.*, 2003). Cui *et al.* (2003) take a different approach. They use a more complicated model of the error implicit in microarray experiments, but generate the raw expression values randomly from statistically plausible probability distributions.

In this study, microarray experiments are done similarly to Repsilber & Kim (2003), however without the averaging over multiple experiments. We have assigned the first gene as the ‘starting gene’: this gene has a strong self loop and starts the simulation with a positive expression level (2 in our simulations) while all other genes begin at zero. As the starting node has a self loop, it is constantly activated and pushes the network out of the null steady state.

Using Euler’s method and a fixed step size, the ARN is simulated using numerical integration of the differential equations (ie. Equation 4.1) for a small number of time steps. The network is simulated for s time steps with node values recorded for each time step to form a time series of network activity (s is chosen such that the steady state begins immediately after). In an approach inspired by oligonucleotide microarray experiments (eg. Affymetrix experiments), a single gene expression value is generated for each measured node. At this stage, sampling error and other forms of noise are not added to the simulated data. The result of this step is a matrix of values with a row for each measured node and a column for each time step (Figure 4.3). The time series for the population of evolved networks are generated in the same manner.

$$\begin{array}{c|ccc}
 & 1 & \dots & t \\
\hline
1 & 2.0 & \dots & \\
2 & 0 & & \\
\vdots & \vdots & & \\
n & 0 & &
\end{array}$$

Figure 4.3: The time series of a simulated network is a matrix, with each row corresponding to a gene, $1, ...n$, and each column corresponding to a time step, $1...t$. The first gene starts with an expression level of 2, all others begin with zero.

4.2.3 Step 3: Evolve population of networks

Next, ARNs are evolved using an evolutionary computation algorithm to discover networks that produce output similar to the target ARN.

A challenge when inferring ARNs is the difficulty in assessing the effectiveness of algorithms because the real biological regulatory network structure is mostly unknown. Some researchers approach this challenge by comparing the induced ARNs with target ARNs, for example Wahde & Hertz (2000). This difficulty has led some researchers, e.g. Repsilber & Kim (2003) and Mendes *et al.* (2003), to test proposed methods with artificially generated, but plausible GRNs. Microarray data is simulated for the ARN and the proposed solution to the reverse problem is compared with the known ARN. A similar approach is used in this work.

To use an evolutionary approach to discover networks, it must be possible to encode each candidate ARN as an artificial genome. In Section 4.1 above, we describe how ARNs are represented as a square matrix W of interaction weights (Figure 1.16). This matrix completely defines the behaviour of a particular ARN from given initial conditions. For this reason, we simply encode each ARN by its matrix W.

For the evolved networks, each node i is mapped to node i in the target network. However, the evolved networks can undergo gene duplications and deletions. For this reason, the top leftmost $n \times n$ sub–matrix (where n is the size of the target matrix) in each evolved network is constrained such that these nodes may not be lost by evolution, as the evolution protocol includes node duplication and deletion operations. Wiring between these nodes, however, is adaptable by evolution. Also, the first node (w_{11} has a hard–wired feedback to itself to simplify the evolution task. All other weights, however, can be changed during evolution.

The artificial evolution begins with a population of randomly created networks 10% larger than the target matrix with weights drawn uniformly and randomly from the range [-1, 1]. However, evolution may increase or decrease the number of genes (rows/columns) in the matrix, but not less than the number of nodes in the target network. That is, the evolution protocol consists of a set of network mutations with an associated probability of occurrence at each time step, for example edges may switch nodes or be deleted, weights may be changed by a small amount, and nodes can be duplicated (see Table 4.1 for a list of mutation operators). At present we have not implemented cross-over between child networks (see Section 1.21).

Mutation	Matrix Operation
Node duplication/deletion	A duplication involves copying a node plus all its incoming and outgoing connections. Deletion is the opposite.
Edge duplication/deletion	When an edge is duplicated, the source remains the same but the target is chosen randomly.
Edge switch	Two edges have their weights exchanged.
Edge move	The target of an edge is changed to a random node.
Edge creation	An edge is created with a random source and target.
Weight change	The weight for an edge is altered by a small random amount.
Network duplication	The entire network is duplicated. This operation was experimental and not used in all simulations.

Table 4.1: During the evolution process networks are evolved by randomly applying mutations. Each mutation represents a particular change in the matrix encoding the network.

Algorithm parameters

For the experiments in this study, the population size was typically 1000 individuals and run lengths were up to 6000 generations. All networks were mutated, and the fittest 50% were taken for the next generation.

Fitness function

The fitness of each individual network in the population is computed by a fitness function. The fitness function compares the time series of the individual with the time series of the target network and derives a distance, where a low distance corresponds to a high fitness So, for each genome, the GRN is simulated using equations described in the above section (Equation 4.1) with the same initial conditions and constant parameters as for the target GRN. Only expression values for the first n genes (ie. the upper leftmost $n \times n$ submatrix matching the target) are recorded. The distance of the resulting time series of expression values for each of the n genes to the target values is computed using a simple squared error approach:

$$d = \sum_{i=1}^{n} \sum_{t=1}^{s} (x_{it} - y_{it})^2 \tag{4.3}$$

where d is the distance of the evolved genome time series X to the target genome time series Y. Evolution seeks to minimise the distance d. d sums over each of the n genes x_{it} indexed by gene i and time t, for each of the s time steps. The value y_{it} is the target gene expression value of gene i at time t.

An elitist strategy is used to manage the population. After calculating the fitness of the population, the weakest 50% of the population is deleted and replaced with the fittest 50%. Genetic operators (Table 4.1) are then applied to the population with (small) fixed probability. Genetic operators do not modify the feedback edge to the ‘initial’ node, ie. w_{11}, nor do they delete any of the nodes in the top leftmost $n \times n$ submatrix, as these correspond to the target network nodes. At this stage, no recombination was implemented.

Fitness is computed for members of the new population and evolution continues until networks with suitably small fitness values are evolved or the maximum number of generations is reached.

4.2.4 Step 4: Compare networks

At the time of this study, no sophisticated comparison methods were applied to compare the topology of the target network and the topology predicted from the evolutionary algorithm, in other words the networks were compared ‘manually’. This was not as laborious as it sounds. For networks producing interesting dynamics (which is defined as any activity that *isn’t* monotonic increase to a steady state or no activity) there was usually a small core of genes that defined the overall network behaviour.

Each target network was first classed according to the network dynamics. After a number of different simulation runs, the evolved networks were compared with the target to determine how much of the network topology was recovered. This is explained in more detail in the next chapter.

Chapter 5

Results and Discussion

This chapter discusses the results of the gene regulatory network reconstruction algorithm described in the previous chapter. The target networks that will have their topology reconstructed are classified according to their dynamical behaviour. This classification aids the interpretation of the reconstruction results. The evolution algorithm is then discussed, followed by a description of the effectiveness of the reconstruction algorithm. A discussion of future work concludes the chapter.

5.1 Classes of randomly generated networks

As discussed in Section 4.2.1, to evaluate the evolutionary algorithm presented in the previous chapter (or other algorithms), many randomly generated networks are needed. For the model of gene regulation used in this work, the first gene has a self loop and begins the simulation with a non-zero expression level, 2.0 in this case, which pushes the network out of the null steady state. This can be thought of as a single constant input. Other authors have experimented with multiple inputs, e.g. (Weaver *et al.*, 1999), but we began here with one of the simplest models.

Hundreds of networks were generated, and a time series of the network dynamics was generated for all networks. After examining a few hundred networks, it became clear that the dynamics of each network could be generally classified into 3 categories (Figure 5.1):

Monotonic increase to steady state: this is by far the most common class of dynamical behaviour seen. All genes that are activated increase monotonically, that is without decreasing, until they reach a steady state. For almost all networks in this class, the steady state for all genes was reached before 40 time steps.

Negative feedback: this class involved a negative feedback loop, where one gene is initially activated, which causes the activation of second gene which represses the first gene. This is seen as a steady state that is lower than the maximum expression level, i.e. there is a ‘spike’ of activation before a reduction.

Oscillations: this class of behaviour was the rarest. It involved a negative feedback loop of three genes, causing the expression level of all three genes to oscillate.

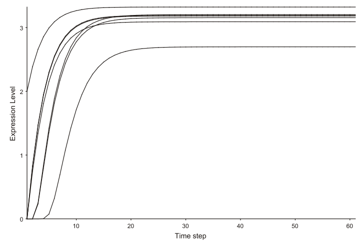
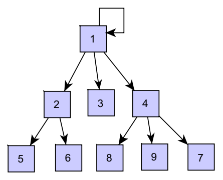
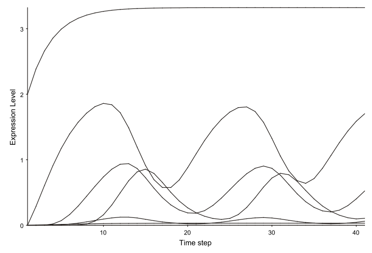
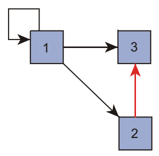
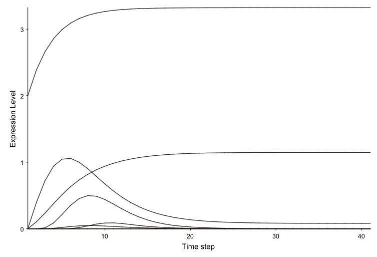
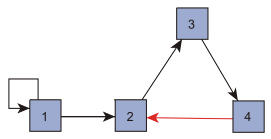
Class description	Example dynamics	Example network topology
Monotonic increase to steady state: this is by far the most common class of dynamical behaviour seen. All genes that are activated increase monotonically, that is without decreasing, until they reach a steady state. For almost all networks in this class the steady state for all genes was reached before 40 time steps.		
Negative feedback: this class involved a negative feedback loop, where one gene is initially activated, which causes the activation of second gene which represses the first gene. This is seen as a a steady state that is lower than the maximum expression level, i.e. there is a ‘spike’ of activation before a reduction.		
Oscillations: this class of behaviour was the least common. It involved a negative feedback loop of three genes, causing the expression level of all three genes to oscillate.		

Figure 5.1: Classes of GRN dynamics

The differences in network dynamics represent differences in network topology. As will be discussed later in Section 5.4, the information contained in a time series for reconstruction purposes depends very much on the general category of dynamical behaviour. Classifying the randomly generated networks into these three classes aids in interpreting the results.

5.2 The evolutionary algorithm

To optimise the evolutionary algorithm (described in Section 4.2.3), its structure parameters were systematically altered, such as population size, mutation rates, and number of generations.

In all the observed runs of the evolutionary algorithm, the average distance decreased monotonically (after smoothing for random fluctuations). After 500 generations the largest average increase in fitness usually occurred, followed by incremental improvements. This occurs because the solution space is defined by the first generation of randomly generated networks. An estimation of the stability of the final solution is acquired by repeating the whole procedure many times, and comparing the final networks (see next section). Repeating the procedure multiple times corresponds to starting at different places in solution space.

Within the limits of the computing resources available, population sizes of 1000 were found to be of sufficient size. Larger population sizes did not significantly improve the final fitness or the stability of the solutions.

5.3 Network Topology Reconstruction

Three networks of each category in Figure 5.1 were chosen for closer analysis and comparison. The objective was to run the reconstruction procedure many times for each target network to determine if a) the network topology could be reconstructed, and b) if the solutions were consistent between attempts.

5.3.1 Monotonic increase to steady state

Three networks that fit this category were chosen semi-randomly, so that the dynamics of each network were not too similar, yet all in the above mentioned category. The reconstruction procedure was applied a number of times to each network, and the resulting predicted networks were compared to the target network.

The evolutionary algorithm furnished networks that reproduced almost exactly the time series of the target network, with the time series distances (Section 4.2.3, Equation 4.3) very low, (< 0.5). For a target network with seven nodes and a time series of 30 steps, this corresponds to an average difference of less than 0.001 per node per time step, meaning the time series of the target and predicted networks are almost identical. This ability was consistent, in that almost all repetitions of the reconstruction method produced similarly low distances, regardless of the target network (for networks within this category).

However, when the topology of the target network and the reconstructed networks were compared, there was little in common. Apparently, this is due to the type of topology of the target network. Almost all the target networks in this category consist of cascades of sequentially activated nodes, with little interaction between the cascades. From the starting node, which has a self loop, most nodes are activated by a single incoming edge. Feedback loops, both positive or negative, are uncommon.

This topology results in the sequential activation of most genes, with a steady state reached after the starting node has reached its steady state. The lack of interaction between cascades means subtrees in the cascade could be switched, and the resulting topologically rearranged network produces an almost identical time series (see Figure 5.2). In other words, there are too many network topologies that can produce the same time series in this category, meaning the evolutionary algorithm is not sufficiently constrained. There are several possibilities to further constrain the time series of such networks, that are discussed below in Section 5.4.

The reconstruction procedure did not work well for networks in this category. The time series did not contain enough information to constrain the number of possible solutions, in that many network topologies can reproduce the time series of the target network, due to the cascade structure of the target networks. However, our analysis suggests that this cascade structure is typical for networks exhibiting this kind of output.

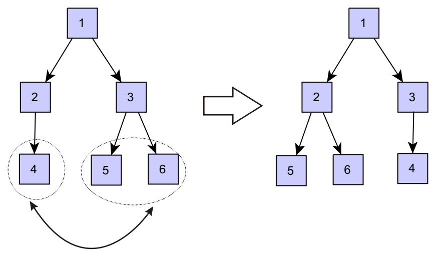


Figure 5.2: The topology of networks in the category ‘Monotonic increase’ is usually a cascade of nodes from the starting node with little interaction between cascades. This means that ‘sub-cascades’ can be switched, such as between nodes 2 and 3, without any large difference in the time series of the network. The result is that many networks can reproduce the time series of a target network of this category and there is little chance of the exact topology of the target network being correctly reconstructed using the methods discussed here (although the type of topology can be).

5.3.2 Negative feedback

The reconstruction procedure was applied to 3 networks in this category. For each target network, the reconstruction procedure was repeated at least 10 times. The distance measure between the time series of the target and predicted networks varied between reconstruction attempts. For approximately half the attempts, the distances were relatively low, ~ 0.5, corresponding to very similar time series. For the other half, the distance was between 0.5 and 2.0.

A manual comparison of network topologies was done for all networks in this category. For clarity, only one is discussed here, however the results apply to all three. The network (Number p03_n10_4967) consisted of ten nodes (see Figure 5.3).

The main part of the network that determines the dynamics of this network is the negative feedback loop, where the starting node (node 1) activates both node 7 and node 10. Node 10 then strongly activates the rest of the network. When node 7 begins to be expressed, it then represses node 10 and the subsequent connected nodes.

A small problem in terminology arises when describing the structure of this network, which must be elaborated upon to avoid confusion. Specifically, we use the term ‘feed-forward motif’ and ‘feedback loop’ when describing the same parts of a network but we mean different things. A feed-forward motif is a network structure consisting of three nodes such as in Figure 5.4. Note that the edges are not differentiated between positive and negative regulatory interactions, but the direction of the edges is what defines the feed-forward motif. The feed-forward motif can have different functions depending on the sign of the interactions Mangan & Alon (2003). Now when we use the term ‘feedback’, we are describing a more general feature, for example, a node that

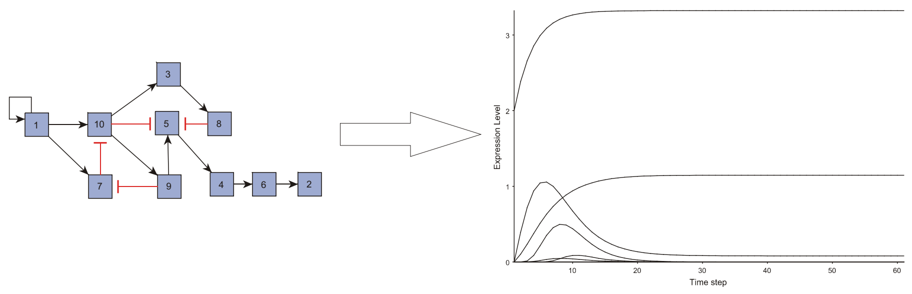


Figure 5.3: Network number p03_n10_4967 and the resulting time series. Activating interactions have the arrow ↑ and repressing interactions have the arrow ⊤. The starting node (node 1) has a self loop and begins with a positive expression which drives the network out of the null steady state. After node 1 initially activates node 10 and subsequently activating the rest of the network, node 7 eventually represses node 10 and the subsequent nodes.

activates its own expression exhibits 'positive feedback'. The network structure in Figure 5.4 could display positive or negative feedback depending on the sign (positive or negative) of the interactions. For example, if the edge from node 2 to 3 is negative, then the structure exhibits negative feedback. Thus we differentiate between a specific topological structure, the feed-forward motif, and properties of the network structure, such as negative and positive feedback.

The subnetwork structure (or *motif*) consisting of nodes 1, 7, and 10 is termed a feed-forward motif (Figure 5.4 and previous paragraph) (Mangan & Alon, 2003). The feed-forward motif is one of three motifs that occurs more often in the *E. coli* transcriptional regulation network compared to randomised networks with the same edge distribution, when ignoring the signs of the edges (Shen-Orr *et al.*, 2002). The motif can perform an information-processing function depending on the signs of the interactions (activation or inhibition), such as a sign-sensitive transcription accelerator (speeding up the transcription rate in one direction e.g. off to on), or a sign sensitive delay, or can generate pulses of transcription activity (Mangan *et al.*, 2003; Mangan & Alon, 2003; Kashtan *et al.*, 2004). Pulse generation is also the function observed in the network examined here, which depends on a specific edge being negative.

All networks that display a pulse of transcription activity have the feed-forward motif or a very similar motif directly connected to the starting node. Approximately half the reconstruction attempts on the target network p03_n10_4967 showed this motif, often only with one or two edges different. In many cases, the exact motif was present but involving different nodes, with the equivalent nodes in the target network immediately adjacent to the feed-forward motif.

For the target networks examined in this work, the rest of the network connected to the feed-forward motif was usually a cascade. This meant that, as above, the time series for these nodes could be effectively reproduced by the reconstructed networks, but the topology could not. This limitation is probably a result of the simple model used in this work where an initial node starts with a self loop and quickly reaches a steady state. This is equivalent to a single constant input, which contains little information. Improved designs are outlined in the discussion, Section 5.4.

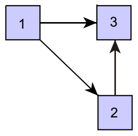


Figure 5.4: The feed-forward motif: this network structure, or *motif*, occurs many more times in the transcription factor network of *E. coli* than could be attributed to chance. In this figure, arrows mean a positive or negative interaction. Depending on if the interactions are positive or negative, the feed-forward motif can perform several different functions, such as a sign-sensitive transcription acceleration (speeding up the transcription rate), or a sign-sensitive delay, or it can generate pulses of transcription activity (Mangan *et al.*, 2003; Mangan & Alon, 2003; Kashtan *et al.*, 2004).

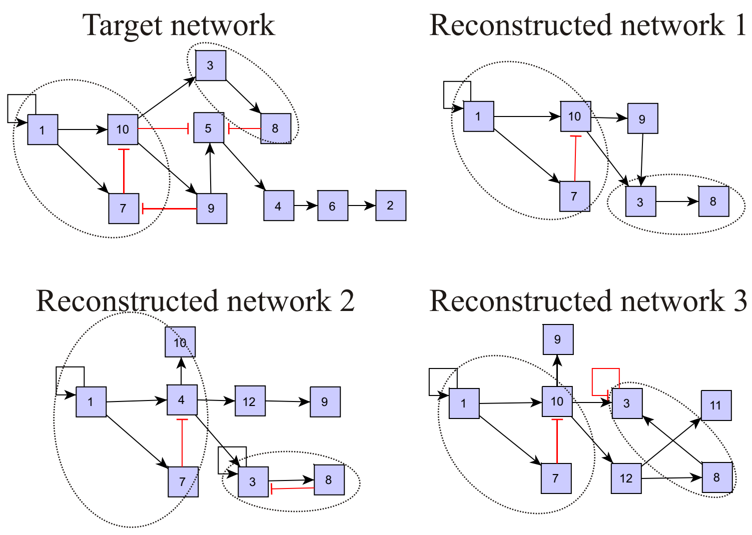


Figure 5.5: Target network p03_n10_4967 and the best scoring networks from three reconstruction attempts. Activating interactions have the arrow $\uparrow$ and repressing interactions have the arrow $\top$. The target network is circled where the reconstruction algorithm has correctly predicted the topology, the feed-forward motif of nodes 1,7 , and 10, and the cascade of 3 and 8 from node 10. In two of the three reconstructed networks, node 9 is activated directly by node 10, as in the target network. The distances of the time series of the three reconstructed networks to the target time series are 0.22, 0.73, and 0.36 for the reconstructed network 1, 2, and 3 respectively.

5.3.3 Oscillations

In contrast to the previous two categories, the reconstruction procedure could not produce networks that closely replicated the time series of the target network, i.e., the distance measure between the target and predicted time series was always relatively high, the lowest being 1.4, but most were ~ 3.0. As the time series could not be replicated, the topology of the target network could not be reconstructed.

The time series could not be replicated because the topology that produces oscillations is relatively rare when randomly generating networks are considered, therefore, relatively few randomly generated networks display this category of dynamics. For a network to display oscillations using the model of gene regulation used in this work, a very specific subnetwork structure must be present. This structure was present in all the randomly generated networks that displayed oscillation. The subnetwork consists of a unidirectional loop, with one edge a repressing interaction and the rest activating interactions (see Figure 5.6).

This subnetwork structure is highly specific for producing oscillations. This meant that during the reconstruction procedure, if no network in the first generation of evolved networks displayed oscillations or cycles, then the evolution procedure would try to fit the time series of a network with the category 'monotonic increase', however it is unlikely that the evolution algorithm can get out of this local 'optima' in sufficient time. In addition, there is no gradual path in the solution space from oscillating to non-oscillating networks. When this subnetwork is present, then the time series may show oscillations, however if it is absent then the network cannot produce oscillations. Thus the problem in reconstructing this category of networks is the rarity of the network structures that produce oscillations in the time series.

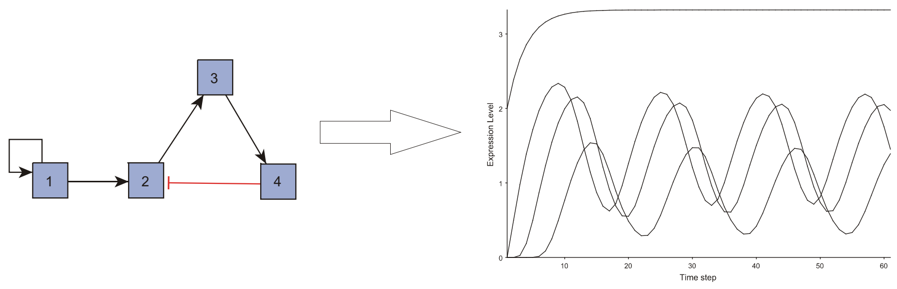


Figure 5.6: An oscillation producing network. The network on the left produces the oscillating time series on the right. All randomly generated networks producing oscillations had a similar subnetwork, consisting of a unidirectional loop with one edge a repressing interaction (⊤) and the rest activating interactions (↑).

5.4 Discussion

Figure 5.7 summarises the results of the evolution procedure applied to the three categories of networks. For the first network category, monotonic increase, the target time series could be replicated by the predicted networks produced by the evolution algorithm, however the network topologies predicted were only in type similar to the target network topology. This is because there are many networks that can produce the same time series. For example, in the cascade shown in the left column in Figure 5.7, the nodes activated by Node 2 could instead be activated by Node 4, and vice versa. In other words the subnetworks could be switched with little noticeable difference in the time series. Because of this, there is little chance that the evolutionary algorithm will find a network that happens to closely match the target network.

For networks showing negative feedback in the dynamics, the time series could be reproduced by the reconstructed networks, and the important network motif that determined the network behaviour, the feed-forward motif, could be reconstructed in approximately half the reconstruction attempts. Other nodes that are directly connected to the feed-forward motif were also correctly reconstructed in many cases.

Target networks that display oscillations in the time series could not be reconstructed because network motifs that produce oscillations are very rare in randomly generated networks. The result is that the time series of networks of this category could not be reproduced by reconstructed networks, making it impossible for the topology to be reconstructed.

This shows that, under the network model used in this work, it is possible to reconstruct networks, depending on what kind of dynamics the network produces. In other words, there has to be enough information in the time series.

Discussed below is future work designed to address the problem of including more information in the time series.

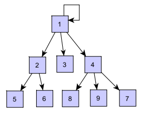
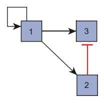
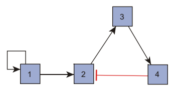
	Monotic increase to steady state	Activation followed by negative feedback	Oscillations
Time series reproduced by evolutionary algorithm	yes	yes	no
Network topology reproduced by evolutionary algorithm	no	partially	no
Typical network structure			

Figure 5.7: The table summarizes the results of this chapter. For the network category ‘monotonic increase’, the evolutionary algorithm produced networks that could reproduce the time series of the target network. However, the topology of the target network was a series of cascades with little interaction between the cascades which meant that many network topologies could produce the time series. Thus, there was little consistency between the solutions. For the network category ‘negative feedback’, the target networks usually had a feed-forward motif, and the algorithm had considerable success in reconstructing this part of the network, as well as nodes immediately connected to this structure. For the network category ‘oscillations’, the time series could not be reproduced by the algorithm and, thus, nor the network topology, which is apparently due to the highly specific subnetwork structure needed to produce oscillations.

5.5 Future work

There are several immediate possibilities for improving the reconstruction process. One is to use the idea of D’Haeseleer *et al.* (2000) where external inputs is modeled, such as artificially changing the expression of a specific gene at a specific time step, corresponding to an environmental signal. This would add more information to the network time series and constrain the number of possible networks topologies, making accurate reconstruction of the target network more likely.

Experimentally, this would involve an artificial or natural gene construct that responds to some added signal.

Another possibility is to use ‘knockout’ experiments where each gene is sequentially destroyed and the time series recorded. Knockout experiments are a standard experimental procedure and would be an ideal way to further test the reconstruction procedure presented here.

Further evaluating the reconstruction method presented here with other forms of experimental data will aid in designing a procedure for evaluating other gene-network reconstruction methods such as those discussed in Chapter 1. The overarching question of this entire project concerns the unknown regulators in a network. The biggest problem using microarray data to build models of gene-expression networks is not the noise of the sample, nor the intrinsic noise of the biological phenomena. It is the fact that microarrays only measure “fingerprints” of the real network. In addition, building models using only such data makes the assumption that measured mRNA levels correspond approximately to protein levels in the cell that in turn regulate mRNA levels. While this may be a reasonably valid assumption in prokaryotes, for eukaryotes it is not (Gygi *et al.*, 1999).

Crucial measurements that are not made are the actual concentration of the protein for which the mRNA is translated, the resulting variant if different exon combinations are possible, post-translational modification, protein localisation, its current state (for example bound by an inhibitor protein), and many others, as e.g. the recent discovery of the impact of small RNA strands (*microRNA*) on all levels of genomic regulation (Ambros, 2001). That is, the RNA world likely plays a very important role in gene regulation that currently is not well understood. Noise starts to look unimportant against all the missing data.

The result is a network with many unknown nodes and edges (Fig. 5.8). This is the direction of future work: assuming it is *possible* for a network to be reconstructed from experimental data, what affect do unmeasured nodes have on the reconstructed networks? The first step, undertaken in this section, is to investigate the limitations of reconstruction methods when **all** genes (nodes) in the network are visible, and to set up a framework for comparing different reconstruction methods.

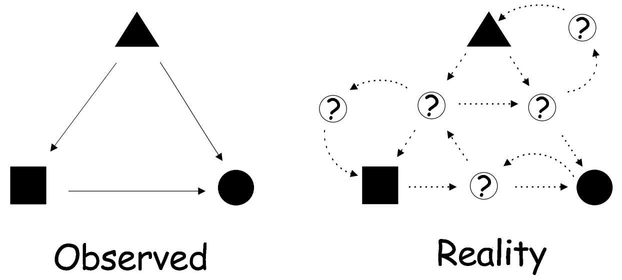


Figure 5.8: Microarrays only measure a fraction of the real gene regulatory network. Not measured are protein concentrations, post translational modification and localisation, etc. Future work will be directed at addressing this question using previously published gene regulatory reconstruction algorithms and the algorithm presented here.

Chapter 6

Closing comments

This thesis has regarded the genomic machinery of a cell as a network. A simplification, but a useful one, as (in this thesis) it partitions the task of understanding the cellular interaction network into two distinct problem areas (Figure 6.1).

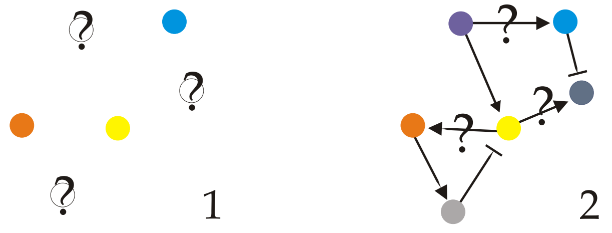


Figure 6.1: This thesis has addressed two problems in piecing together a network of cellular interactions: 1) Reconstructing the function of genes (coloured circles) that are uncharacterised; and 2) Investigating the limits of gene regulatory networks reconstruction methods using microarray experiments. This can be be thought of as 1) reconstructing the nodes in a network, and 2) reconstructing the edges in a network.

Chapters 2 and 3 dealt with understanding the nodes of the network, in this case, by predicting the function of uncharacterised genes using homology and phenotype information. This approach is akin to classical genetics in that we go from phenotype to genotype, instead of the other way around that has become more common in the molecular age. It is an approach that exploits evolution, in that the statistical significance, or 'surprise' value, becomes greater the longer the divergence time for similar genes or proteins. In other words, genes remaining similar over geologically long time periods in phylogenetically distant organisms implies a preservation of function. And with techniques exploiting evolution in this manner, the more genomes (or other data sets) one has, the more robust and useful the predictions. This general approach has been used recently on microarray data. Normally microarray data require several statistical processing steps, and the conclusions drawn are usually tentative. However, in Stuart *et al.* (2003) the authors gathered as much microarray data for a single species as was feasible and looked for co-occurrence of gene pairs. This technique has been done before on single experiments, and the conclusions were always tentative due to the relatively small data sets. However, by looking for co-occurrence of gene expression levels that are the same in many disparate experiments from many different research groups dealing with different stresses and developmental conditions, the chance of finding such co-occurrences by chance alone diminishes rapidly. In other words, to detect a weak signal you just need a bigger data set. Approaches of this nature are expected to become more common as biological data sets become larger and more diverse.

In this author's opinion, the most interesting results from Genetrawler were from investigating the assimilatory sulfur reduction phenotype. Here, several genes were detected, not from adjusting the thresholds, but from adjusting the e-value threshold. All the genes detected in such a manner (excluding the known core pathway genes) all relied on the sulfur reduction pathway as all the proteins encoded by these genes used a sulfur-metal group as a co-factor. These genes are dependent on the pathway, and the dependency has been preserved enough over the course of evolution for this method to detect.

As with any prediction method, the predictions require experimental verification. This was not performed in this work due to lack of resources and time. However, one experimental protocol for validating the functional predictions made by Genetrawler are gene knockouts, which are a standard laboratory exercise. In fact, they have been used previously to confirm the usefulness of a similar method (Levesque *et al.*, 2003). A limitation is that the validation requires the effect of the gene knockout to be observable, which will not be the case for every gene and phenotype. And in cases such as pathway dependency discussed above, removal of such a pathway is highly likely to be lethal.

Chapters 4 and 5 tackled the problem of reconstructing the topology of gene regulatory networks from microarray experiments. Actually, it investigated the limitations of this kind of approach. Since the microarray experiments were idealised (we assumed no experimental error), and the simulated time series rather long compared to (currently) realistic time series, the hope for network reconstruction on real experimental data seems limited. There are, however, some extra points worth discussing that both offer reasons for optimism and give more reasons for skepticism. As discussed previously in Section 5.5, microarray experiments only measure a fraction of the real interaction network. We don't know how much information processing occurs via ncRNAs in eukaryotes, but it appears substantial. That is why this work is directed at prokaryotes, where the correlation between transcription concentration and protein concentration is higher. However protein-protein interactions in prokaryotes are not measured. On the plus side, the networks used in this work had only one continuous input signal. Providing multiple input signals would constrain the networks further and improve the reconstruction capability of this approach. Using artificial gene knockouts and measuring the transcription time series could provide extra useful data and also constrain the possible numbers of networks that could fit the time series. Thus, both methods presented here can use the same experimental verification: gene knockouts. For prokaryotes, this procedure is well established and reasonably reliable.

A reason for optimism is that, in our case, we assume no interactions are previously known. However, for most regulatory networks there is at least a small part characterised. If these known interactions were included, it may be enough to substantially constrain the possible networks and thus provide useful network topology predictions.

In conclusion, we have presented a useful tool for working biologists for predicting the function of bacterial genes, and demonstrated some limitations on gene regulatory network reconstruction methods using microarray data.

Appendix

A Genetrawler species classifications

Query: *Bacillus subtilis,subsp. subtilis str. 168*

Exclude genomes (without flagella):	Include genomes (with flagella):
Archaeoglobus fulgidus	*Aquifex aeolicus*
Bifidobacterium longum NCC2705	*Borrelia burgdorferi*
Chlamydia trachomatis D/UW-3/CX	*Campylobacter jejuni*
Deinococcus radiodurans	*Escherichia coli UPEC-CFT073*
Fusobacterium nucleatum subsp. nucleatum ATCC 25586	*Helicobacter pylori 26695*
Haemophilus influenzae Rd	*Pseudomonas aeruginosa PAO1*
Methanobacterium thermoautotrophicum Delta H	*Thermotoga maritima*
Methanococcus jannaschii	*Vibrio cholerae*
Methanopyrus kandleri AV19	
Methanosarcina acetivorans str. C2A	
Mycobacterium tuberculosis CDC 1551	
Mycoplasma genitalium	
Mycoplasma penetrans HF-2	
Mycoplasma pneumoniae	
Mycoplasma pulmonis	
Neisseria meningitidis MC58	
Pyrococcus abyssi	
Pyrococcus furiosus DSM 3638	
Pyrococcus horikoshii	
Rickettsia prowazekii	
Sulfolobus solfataricus	
Sulfolobus tokodaii	
Synechocystis sp PCC6803	
Thermoplasma acidophilus	
Thermoplasma volcanium	
Thermosynechococcus elongatus BP-1	
Xylella fastidiosa	

Table A1: Species in Flagella analysis (Chapter 3.3.1).

Query: *Mesorhizobium loti*

Exclude genomes (without nitrogen fixation):	Include genomes (with nitrogen fixation):
Aeropyrum pernix K1	*Anabaena sp.*
Archaeoglobus fulgidus	*Bradyrhizobium japonicum USDA 110*
Bacillus halodurans	*Chlorobium tepidum TLS*
Bacillus subtilis,subsp. subtilis str. 168	*Clostridium acetobutylicum ATCC824*
Bifidobacterium longum NCC2705	*Mesorhizobium loti*
Borrelia burgdorferi	*Methanobacterium thermoautotrophicum Delta H*
Brucella melitensis 16M	*Methanosarcina acetivorans str. C2A*
Brucella melitensis biovar suis 1330	*Methanosarcina mazei Goe1*
Buchnera aphidicola str. Sg (Schizaphis graminum)	*Nostoc sp PCC7120 (Cyanobacteria)*
Campylobacter jejuni	*Rhodopseudomonas palustris CGA009*
Chlamydia trachomatis D/UW-3/CX	*Sinorhizobium meliloti 1021*
Chlamydia trachomatis MoPn Nigg (replace muridum)	
Chlamydophila pneumoniae AR39	
Chlamydophila pneumoniae J138	
Clostridium perfringens 13	
Corynebacterium efficiens YS-314T	
Corynebacterium glutamicum ATCC 13032	
Deinococcus radiodurans	
Escherichia coli UPEC-CFT073	
Haemophilus influenzae Rd	
Halobacterium sp. NRC-1	
Helicobacter pylori 26695	
Helicobacter pylori J99	
Lactococcus lactis subsp. lactis	
Leptospira interrogans serovar lai str. 56601	
Listeria innocua	
Listeria monocytogenes	
Mycobacterium leprae	
Mycobacterium tuberculosis CDC 1551	
Mycobacterium tuberculosis H37Rv	
Mycoplasma genitalium G37	
Mycoplasma penetrans HF-2	
Neisseria meningitidis MC58	
Neisseria meningitidis Z2491	
Oceanobacillus iheyensis HTE 831	
Pasteurella multocida Pm70	
Pseudomonas aeruginosa PAO1	
Pseudomonas putida KT2440	
Pyrobaculum aerophilum	
Pyrococcus abyssi	
Pyrococcus furiosus DSM 3638	
Pyrococcus horikoshii OT3	
Ralstonia solanacearum GM1000	
Rickettsia conorii Malish 7	
Rickettsia prowazekii strain Madrid E	
Salmonella typhimurium LT2	
Shewanella oneidensis MR-1	
Shigella flexneri 2a str. 301	
Staphylococcus aureus subsp. aureus Mu50	
Staphylococcus aureus subsp. aureus MW2	
Staphylococcus aureus subsp. aureus N315	
Streptococcus agalactiae 2603V/R	
Streptococcus agalactiae NEM316	
Streptococcus mutans UA159	
Streptococcus pneumoniae R6	
Streptococcus pneumoniae TIGR4	
Streptomyces coelicolor A3(2)	
Sulfolobus solfataricus	
Sulfolobus tokodaii	
Thermoanaerobacter tengcongensis	
Thermoplasma acidophilus	
Thermoplasma volcanium	
Thermotoga maritima	
Treponema pallidum	
Ureaplasma urealyticum	
Vibrio cholerae	
Wigglesworthia brevipalpis	
Xanthomonas axonopodis pv. citri str. 306	
Xanthomonas campestris pv. campestris str. ATCC 33913	
Xylella fastidiosa	
Yersinia pestis C092	
Yersinia pestis KIM5 P12 (Biovar Mediaevalis)	

Table A2: Species in Nitrogen Fixation analysis (Chapter 3.3.2).

Query: *Synechocystis sp.*

Exclude genomes (without photosynthesis):	Include genomes (with photosynthesis):
Aquifex aeolicus	*Anabaena sp.*
Archaeoglobus fulgidus	*Chlorobium tepidum TLS*
Bacillus subtilis,subsp. subtilis str. 168	*Nostoc sp PCC7120 (Cyanobacteria)*
Borrelia burgdorferi	*Prochlorococcus marinus str. MIT 9313*
Campylobacter jejuni	*Rhodopseudomonas palustris CGA009*
Chlamydia trachomatis D/UW-3/CX	*Synechocystis sp PCC6803*
Chlamydia trachomatis MoPn Nigg (replace muridum)	*Thermosynechococcus elongatus BP-1*
Escherichia coli UPEC-CFT073	
Haemophilus influenzae Rd	
Helicobacter pylori 26695	
Helicobacter pylori J99	
Methanobacterium thermoautotrophicum Delta H	
Methanococcus jannaschii	
Mycobacterium tuberculosis CDC 1551	
Mycobacterium tuberculosis H37Rv	
Mycoplasma genitalium G37	
Neisseria meningitidis MC58	
Neisseria meningitidis Z2491	
Pseudomonas aeruginosa PAO1	
Pyrococcus horikoshii OT3	
Rickettsia prowazekii strain Madrid E	
Thermotoga maritima	
Vibrio cholerae	
Xylella fastidiosa	

Table A3: Species in Photosysnthesis analysis (Chapter 3.3.3).

Query: *Corynebacterium glutamicum ATCC 13032*

Exclude genomes (without ass. sulfur reduction):	Include genomes (with ass. sulfur reduction):
Bifidobacterium longum NCC2705	*Brucella melitensis 16M*
Chlorobium tepidum TLS	*Corynebacterium efficiens YS-314T*
Mycoplasma penetrans HF-2	*Corynebacterium glutamicum ATCC 13032*
Mycoplasma pneumoniae M129	*Leptospira interrogans serovar lai str. 56601*
Pasteurella multocida Pm70	*Mycobacterium tuberculosis H37Rv*
Pyrococcus furiosus DSM 3638	*Pseudomonas aeruginosa PAO1*
Staphylococcus aureus subsp. aureus Mu50	*Salmonella typhimurium LT2*
Staphylococcus aureus subsp. aureus MW2	*Sinorhizobium meliloti 1021*
Staphylococcus aureus subsp. aureus N315	*Streptomyces coelicolor A3(2)*
	Xanthomonas axonopodis pv. citri str. 306
	Xanthomonas campestris pv. campestris str. ATCC 33913

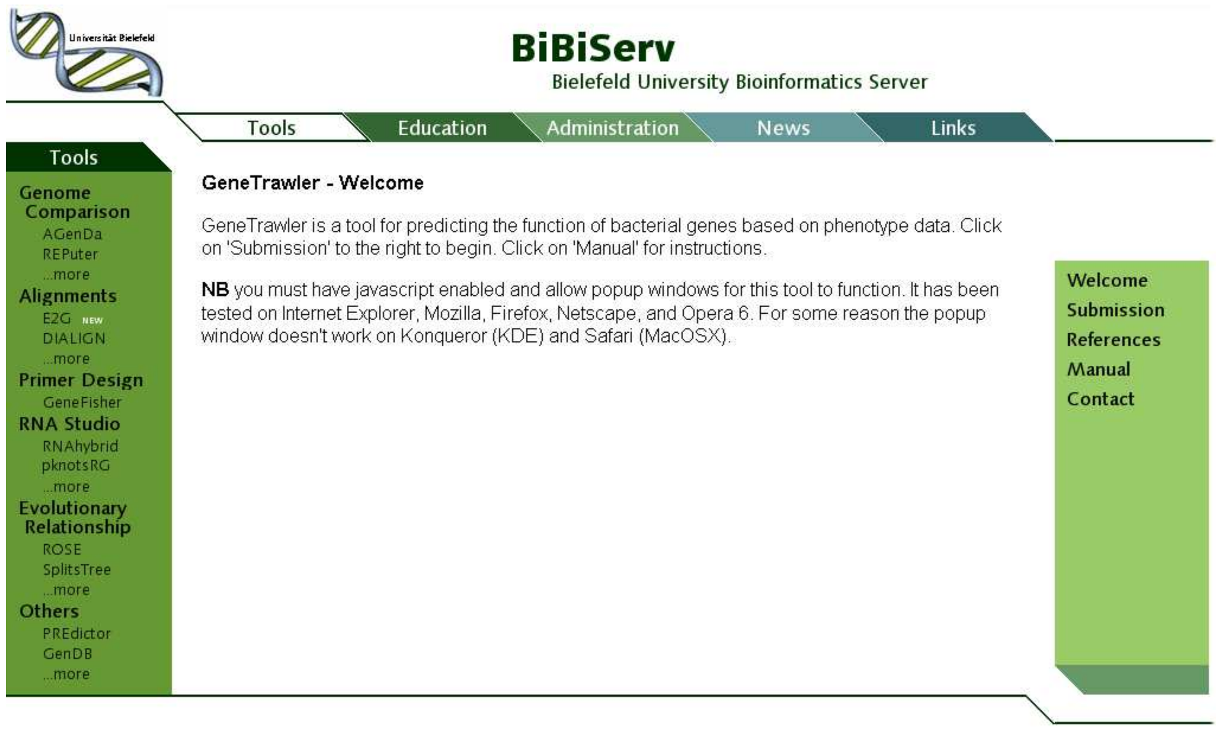
Table A4: Species in Assimilatory Sulfur Reduction analysis (Chapter 3.3.4).

B GeneTrawler Manual

This manual describes in a step by step manner how to predict genes functionally involved in a phenotype using the methods from Chapter 2.

Step 1:

Open a web browser and go to the link: `http://bibiserv.techfak.uni-bielefeld.de/genetrawler/`. The screen in Figure B1 should greet you.



GeneTrawler - Welcome

GeneTrawler is a tool for predicting the function of bacterial genes based on phenotype data. Click on 'Submission' to the right to begin. Click on 'Manual' for instructions.

NB you must have javascript enabled and allow popup windows for this tool to function. It has been tested on Internet Explorer, Mozilla, Firefox, Netscape, and Opera 6. For some reason the popup window doesn't work on Konqueror (KDE) and Safari (MacOSX).

Figure B1: The GeneTrawler welcome screen at http://bibiserv.techfak.uni-bielefeld.de/genetrawler/

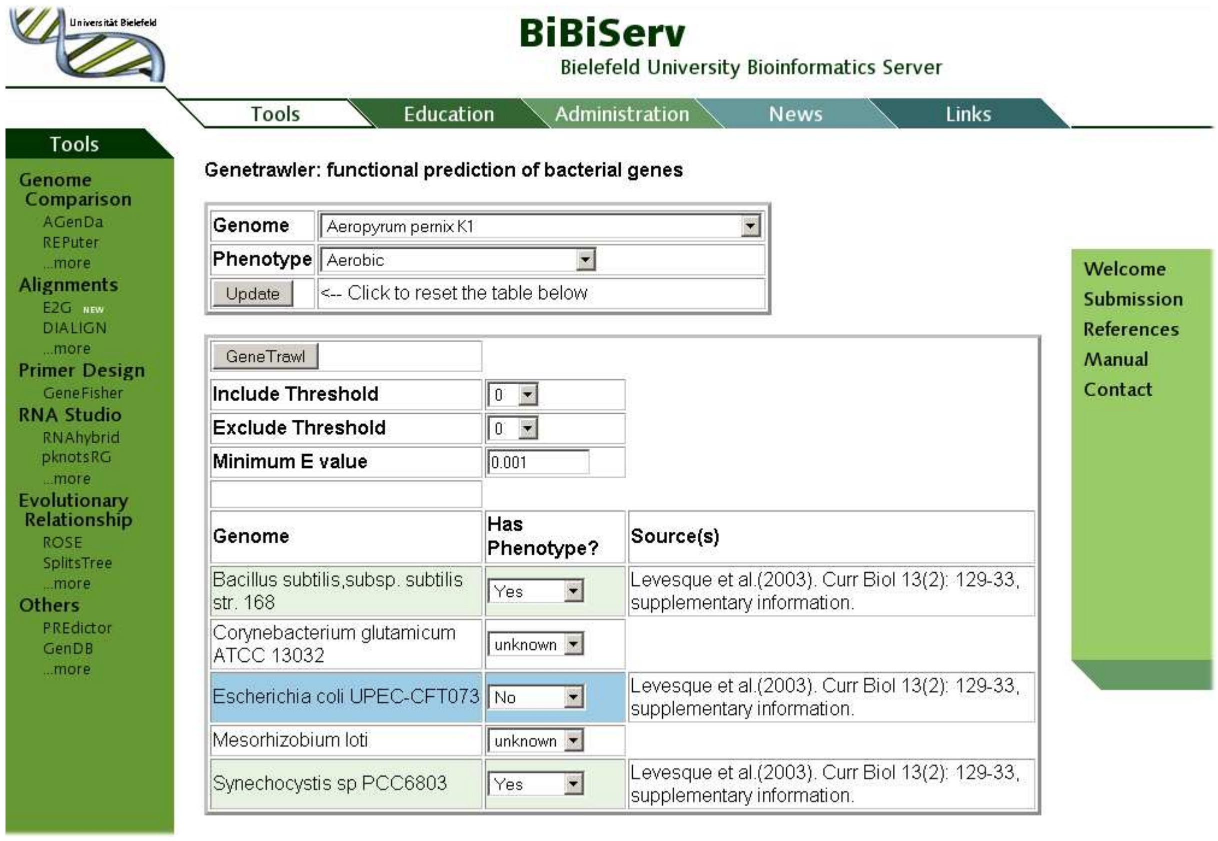
Step 2:

Click on "Submission" on the panel on the right. The screen in Figure B2 should then appear.

Click on the field "Genome" and choose an organism from the scrolling chooser. An an example here we will choose *"Mesorhizobium loti"* as our genome. Then choose a phenotype from the scrolling chooser below the Genome chooser, in our example here we will choose "Nitrogen_fixation" (Figure B3). Once the organism and the phenotype have been chosen, push the "Update" button. The selections will be submitted to the database and the table of phenotype information will be updated.

Step 3:

Now that the genome and phenotype has been selected, choose the include threshold, exclude threshold, and e value threshold, described in Section 2.1.3. For this example, leave the



Genetrawler: functional prediction of bacterial genes

Genome	Aeropyrum pernix K1
Phenotype	Aerobic
Update	<-- Click to reset the table below

GeneTrawl	
Include Threshold	0
Exclude Threshold	0
Minimum E value	0.001

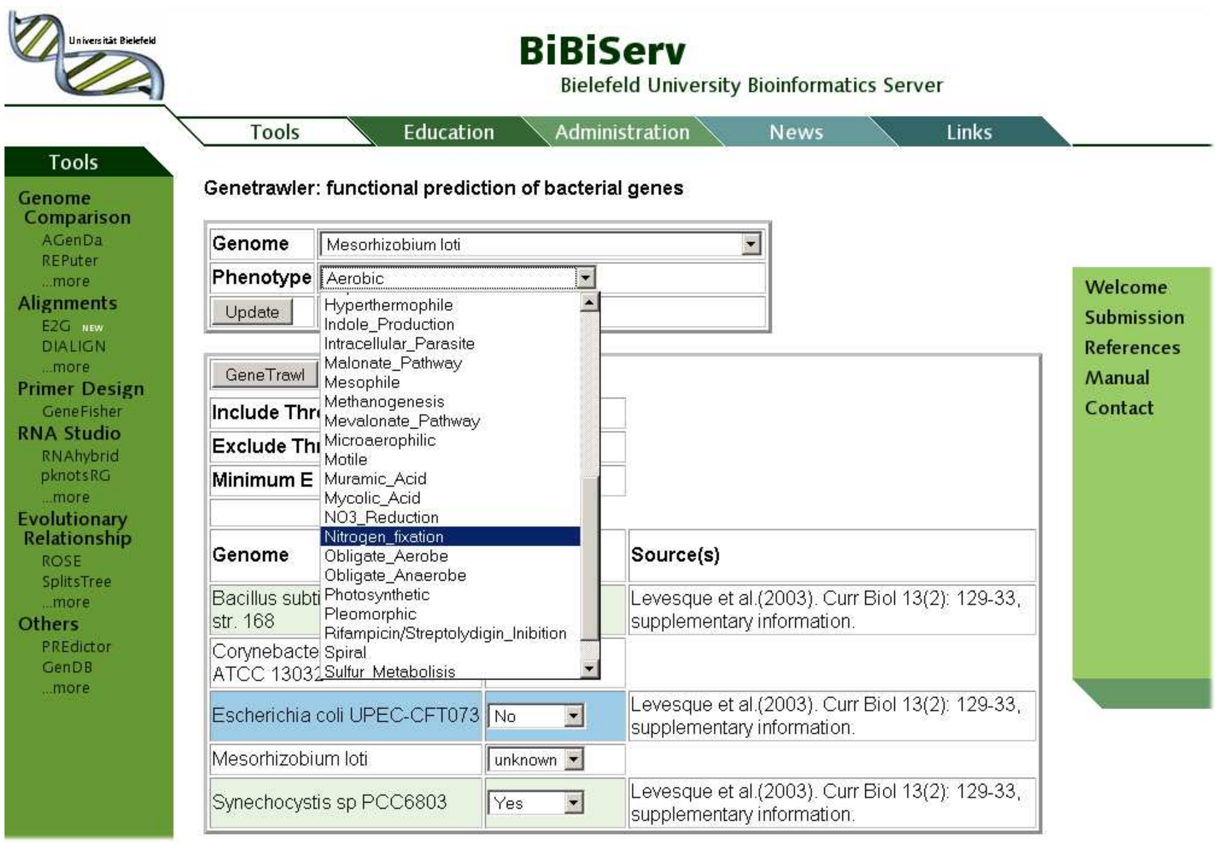
Genome	Has Phenotype?	Source(s)
Bacillus subtilis,subsp. subtilis str. 168	Yes	Levesque et al.(2003). Curr Biol 13(2): 129-33, supplementary information.
Corynebacterium glutamicum ATCC 13032	unknown	
Escherichia coli UPEC-CFT073	No	Levesque et al.(2003). Curr Biol 13(2): 129-33, supplementary information.
Mesorhizobium loti	unknown	
Synechocystis sp PCC6803	Yes	Levesque et al.(2003). Curr Biol 13(2): 129-33, supplementary information.

Figure B2: The submission screen.

parameters at the default setting.

The next step is to check the phenotype data. If the phenotype classifications are satisfactory, push the “GeneTrawler” button at the top of the phenotype data table. A new window will open, and after about 10 seconds (depending on you internet connection and how busy the database is) the results will be displayed (Figure B4).

From the demonstration example here, 5 genes are predicted to be involved in nitrogen fixation in “*Mesorhizobium loti*”, and all five have a nitrogen fixation annotation. A list of the genomes classified as possesing and not possessing the phenotype are also shown.

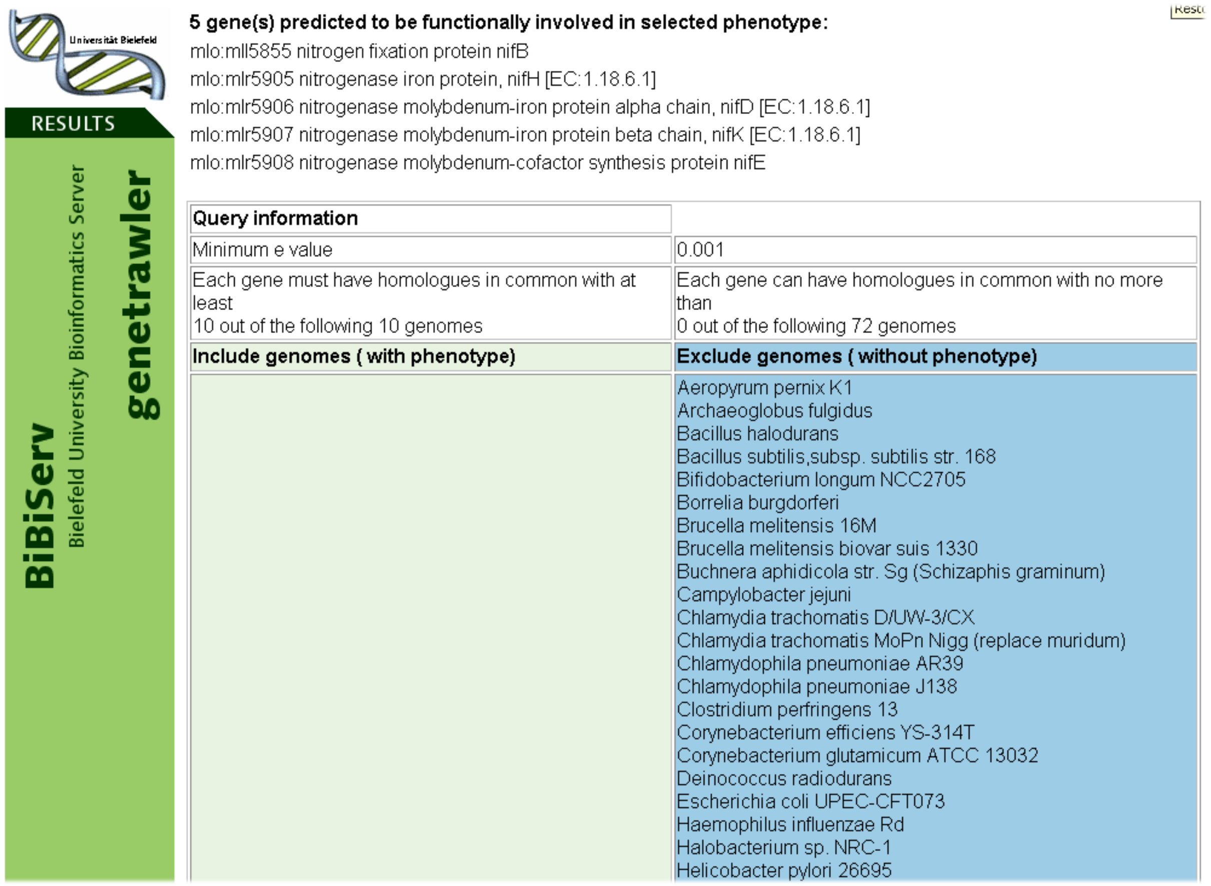


Genetrawler: functional prediction of bacterial genes

Genome	Mesorhizobium loti
Phenotype	Aerobic

Genome		Source(s)
Bacillus subti str. 168		Levesque et al.(2003). Curr Biol 13(2): 129-33, supplementary information.
Corynebacte ATCC 13032		
Escherichia coli UPEC-CFT073	No	Levesque et al.(2003). Curr Biol 13(2): 129-33, supplementary information.
Mesorhizobium loti	unknown	
Synechocystis sp PCC6803	Yes	Levesque et al.(2003). Curr Biol 13(2): 129-33, supplementary information.

Figure B3: Choose the genome and phenotype to analyse.



5 gene(s) predicted to be functionally involved in selected phenotype:
mlo:mll5855 nitrogen fixation protein nifB
mlo:mlr5905 nitrogenase iron protein, nifH [EC:1.18.6.1]
mlo:mlr5906 nitrogenase molybdenum-iron protein alpha chain, nifD [EC:1.18.6.1]
mlo:mlr5907 nitrogenase molybdenum-iron protein beta chain, nifK [EC:1.18.6.1]
mlo:mlr5908 nitrogenase molybdenum-cofactor synthesis protein nifE

Query information	
Minimum e value	0.001
Each gene must have homologues in common with at least 10 out of the following 10 genomes	Each gene can have homologues in common with no more than 0 out of the following 72 genomes
Include genomes (with phenotype)	**Exclude genomes (without phenotype)**
	Aeropyrum pernix K1 Archaeoglobus fulgidus Bacillus halodurans Bacillus subtilis,subsp. subtilis str. 168 Bifidobacterium longum NCC2705 Borrelia burgdorferi Brucella melitensis 16M Brucella melitensis biovar suis 1330 Buchnera aphidicola str. Sg (Schizaphis graminum) Campylobacter jejuni Chlamydia trachomatis D/UW-3/CX Chlamydia trachomatis MoPn Nigg (replace muridum) Chlamydophila pneumoniae AR39 Chlamydophila pneumoniae J138 Clostridium perfringens 13 Corynebacterium efficiens YS-314T Corynebacterium glutamicum ATCC 13032 Deinococcus radiodurans Escherichia coli UPEC-CFT073 Haemophilus influenzae Rd Halobacterium sp. NRC-1 Helicobacter pylori 26695

Figure B4: The results window. 5 genes are predicted to be involved in nitrogen fixation in *Mesorhizobium loti* with the default parameters. A list of genomes and their phenotype classification is also given.

Bibliography

Aboa, E. E., Bairoch, A., Barker, C. W., Beck, S., Benson, D. A., Berman, H., Cameron, G., Cantor, C., Doubet, S., Hubbard, T. J. P., Jones, T. A., Kleywegt, G. J., Kolaskar, A. S., Kuik, A. V., Lesk, A. M., Mewes, H.-W., Neuhaus, D., Pfeiffer, F., TenEyck, L. F., Simpson, R. J., Stosser, G., Sussman, J. L., Tatento, Y., Tsugita, A., Ulrich, E. L., & Vliegenthart., J. F. G. (2000): Quality control in databanks for molecular biology. *Bioessays*, **22** 11: 1024–34.

Adams, M., Kelley, J., Gocayne, J., Dubnick, M., Polymeropoulos, M., Xiao, H., Merril, C., Wu, A., Olde, B., & Moreno, R. (1991): Complementary DNA sequencing: expressed sequence tags and human genome project. *Science*, **252** 5013: 1651–1656.

Akutsu, T., Miyano, S., & Kuhara, S. (1999): Identification of genetic networks from a small number of gene expression patterns under the Boolean network model. *Pac Symp Biocomput*, pages 17–28.

Akutsu, T., Miyano, S., & Kuhara, S. (2000): Inferring qualitative relations in genetic networks and metabolic pathways. *Bioinformatics*, **16** 8: 727–734.

Albert, R., Jeong, H., & Barabasi, A. L. (2000): Error and attack tolerance of complex networks. *Nature*, **406** 6794: 378–82.

Altschul, S. F., Madden, T. L., Schaffer, A. A., Zhang, J., Zhang, Z., Miller, W., & Lipman, D. J. (1997): Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res*, **25** 17: 3389–402.

Ambros, V. (2001): microRNAs: tiny regulators with great potential. *Cell*, **107** 7: 823–6.

Arkin, A., Ross, J., & McAdams, H. H. (1998): Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected escherichia coli cells. *Genetics*, **149** 4: 1633–48.

Arkin, A., Shen, P.-D., & Ross, J. (1997): A test case of correlation metric construction of a reaction pathways from measurements. *Science*, **277** 5330: 1275–1279.

Arkin, A. P. (2000): *Self-Organized Biological Dynamics and Nonlinear Control: Toward Understanding Complexity, Chaos and Emergent Function in Living Systems*, chapter 5: Signal Processing in Biochemical Reaction Networks, pages 112–145. Cambridge University Press.

Armstrong, W. (1998): California's pink salt lakes. *Wayne's Word Noteworthy Plants*. Http://waynesword.palomar.edu/plsept98.htm (11 October 2004).

Banzhaf, W. (2003): On the dynamics of an artificial regulatory network. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J. T., & Ziegler, J., editors, *Advances in Artificial Life, Proceedings of European Conference of Artificial Life (ECAL) 2003, LNAI 2801*, pages 217–227. Springer–Verlag.

Banzhaf, W., Nordin, P., Keller, R. E., & Francone, F. D. (1998): *Genetic Programming An Introduction*. Morgan Kaufmann Pulishers, Inc.

Becskei, A. & Serrano, L. (2000): Engineering stability in gene networks by autoregulation. *Nature*, **405** 6786: 590–3.

Benson, D. A., Karsch-Mizrachi, I., Lipman, D. J., Ostell, J., & Wheeler, D. L. (2004): GenBank: update. *Nucleic Acids Res*, **32 Database issue**: D23–6.

Bergey, D. H., Krieg, N. R., & Holt, J. G. (1984): *Bergey's manual of systematic bacteriology*. Williams and Wilkins, Baltimore.

Blake, W. J., M, K. A., Cantor, C. R., & Collins, J. J. (2003): Noise in eukaryotic gene expression. *Nature*, **422** 6932: 633–7.

Blocker, A., Komoriya, K., & Aizawa, S. (2003): Type III secretion systems and bacterial flagella: insights into their function from structural similarities. *Proc Natl Acad Sci U S A*, **100** 6: 3027–30.

Brutlag, D., Galper, A., & Millis, D. (1991): Knowledge-based simulation of DNA metabolism: prediction of enzyme action. *Comput Appl Biosci*, **7** 1: 9–19.

Burge, C. & Karlin, S. (1997): Prediction of complete gene structures in human genomic dna. *J Mol Biol*, **268** 1: 78–94.

Chen, T., He, H. L., & Church, G. M. (1999): Modeling gene expression with differential equations. *Pac Symp Biocomput*, pages 29–40.

Collado-Vides, J., Gutirrez-Ros, R., & Bel-Enguix, G. (1998): Networks of transcriptional regulation encoded in a grammatical model. *Biosystems*, **47** 1-2: 103–18.

Conant, G. C. & Wagner, A. (2003): Convergent evolution of gene circuits. *Nat Genet*, **34** 3: 264–6.

Cornish-Bowden, A. (1979): *Fundamentals of enzyme kinetics*. Butterworths, London ; Boston.

Covert, M. W., Knight, E. M., Reed, J. L., Herrgard, M. J., & Palsson, B. O. (2004): Integrating high-throughput and computational data elucidates bacterial networks. *Nature*, **429** 6987: 92–6.

Crick, F. & Watson, J. (1953): Molecular structure of nucleic acids. *Nature*, **171** 4356: 737–738.

Cui, X., Kerr, M. K., & Churchill, G. A. (2003): Data transformations for cDNA microarray data. *Statistical Applications in Genetics and Molecular Biology*, **2** 1.

Dandekar, T., Snel, B., Huynen, M., & Bork, P. (1998): Conservation of gene order: a fingerprint of proteins that physically interact. *Trends Biochem Sci*, **23** 9: 324–328.

Davidson, S. B., Overton, C., & Buneman, P. (1995): Challenges in integrating biological data sources. *J Comput Biol*, **2** 4: 557–72.

de Hoon, M. J. L., Imoto, S., Kobayashi, K., Ogasawara, N., & Miyano, S. (2003): Inferring gene regulatory networks from time-ordered gene expression data of Bacillus subtilis using differential equations. *Pac Symp Biocomput*, pages 17–28.

de Jong, H. (2002): Modeling and simulation of genetic regulatory systems: a literature review. *J Comput Biol*, **9** 1: 67–103.

de la Fuente, A., Brazhnik, P., & Mendes, P. (2002): Linking the genes: inferring quantitative gene networks from microarray data. *Trends Genet*, **18** 8: 395–8.

Delcher, A., Harmon, D., Kasif, S., White, O., & Salzberg, S. (1999): Improved microbial gene identification with GLIMMER. *Nucleic Acids Res*, **27** 23: 4636–41.

D'Haeseleer, P., Liang, S., & Somogyi, R. (2000): Genetic network inference: from co-expression clustering to reverse engineering. *Bioinformatics*, **16** 8: 707–26.

D'Haeseleer, P., Wen, X., Fuhrman, S., & Somogyi, R. (1999): Linear modeling of mrna expression levels during cns development and injury. *Pac Symp Biocomput*, pages 41–52.

Eisen, J. (1998): Phylogenomics: improving functional predictions for uncharacterized genes by evolutionary analysis. *Genome Res*, **8** 3: 163–7.

Eisen, J. A., Nelson, K. E., Paulsen, I. T., Heidelberg, J. F., Wu, M., Dodson, R. J., Deboy, R., Gwinn, M. L., Nelson, W. C., Haft, D. H., Hickey, E. K., Peterson, J. D., Durkin, A. S., Kolonay, J. L., Yang, F., Holt, I., Umayam, L. A., Mason, T., Brenner, M., Shea, T. P., Parksey, D., Nierman, W. C., Feldblyum, T. V., Hansen, C. L., Craven, M. B., Radune, D., Vamathevan, J., Khouri, H., White, O., Gruber, T. M., Ketchum, K. A., Venter, J. C., Tettelin, H., Bryant, D. A., & Fraser, C. M. (2002): The complete genome sequence of chlorobium tepidum tls, a photosynthetic, anaerobic, green-sulfur bacterium. *Proc Natl Acad Sci U S A*, **99** 14: 9509–14.

Enard, W. & Pbo, S. (in press): Comparative primate genomics. *Ann Rev Genomics*.

Enright, A., Iliopoulos, I., Kyrpides, N., & Ouzounis, C. (1999): Protein interaction maps for complete genomes based on gene fusion events. *Nature*, **402** 6757: 86–90.

Erdös, P. & Renyi, A. (1960): On the evolution of random graphs. *Publ Math Inst Hung Acad Sci*, **5**: 17–61.

Estes, J. & Palmisano, J. (1974): Sea otters: their role in structuring nearshore communities. *Science*, **185**: 1058–1060.

Finlay, B. J., Curds, C. R., Bamforth, S. S., & Bafort, J. M. (1987): Ciliated protozoa and other microorganisms from 2 African soda lakes (Lake Nakuru and Lake Simbi, Kenya). *Archiv fr Protistenkunde*, **133**: 81–91.

Fisher, C. (1997): http://www.science.psu.edu/iceworms/iceworms.html (11 October 2004).

Fleischmann, R. D., Adams, M. D., White, O., Clayton, R. A., Kirkness, E. F., Kerlavage, A. R., Bult, C. J., Tomb, J. F., Dougherty, B. A., Merrick, J. M., & et al. (1995): Whole-genome random sequencing and assembly of haemophilus influenzae rd. *Science*, **269** 5223: 496–512.

Fogel, L. J., Owens, A. J., & Walsh, M. J. (1966): *Artificial Intelligence Through Simulated Evolution*. Wiley Publishing, New York.

Forterre, P. (2002): A hot story from comparative genomics: reverse gyrase is the only hyperthermophile-specific protein. *Trends Genet*, **18** 5: 236–7.

Friedman, N. (2004): Inferring cellular networks using probabilistic graphical models. *Science*, **303** 5659: 799–805.

Friedman, N., Linial, M., Nachman, I., & Pe'er, D. (2000): Using Bayesian networks to analyze expression data. *J Comput Biol*, **7** 3-4: 601–620.

Garrell, J. & Campuzano, S. (1991): The helix-loop-helix domain: a common motif for bristles, muscles and sex. *Bioessays*, **13** 10: 493–8.

Gillespie, D. (1977): Exact stochastic simulation of coupled chemical reactions. *J Phys Chem*, **81** 25: 23402361.

Gillespie, D. (2000): The chemical Langevin equation. *J Chem Phys*, **113** 1: 297306.

Gilman, A. & Arkin, A. P. (2002): Genetic ”code”: representations and dynamical models of genetic components and networks. *Annu Rev Genomics Hum Genet*, **3**: 341–69.

Goldberg, D. E. (1989): *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesly.

Goss, P. & Peccoud, J. (1998): Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. *Proc Natl Acad Sci U S A*, **95** 12: 6750–5.

Gregory, T. (2001): http://www.genomesize.com/ (20th October 2004).

Gygi, S., Rochon, Y., Franza, B., & Aebersold, R. (1999): Correlation between protein and mRNA abundance in yeast. *Mol Cell Biol*, **19** 3: 1720–30.

Heidtke, K. & Schulze-Kremer, S. (1998): Design and implementation of a qualitative simulation model of lambda phage infection. *Bioinformatics*, **14** 1: 81–91.

Heinrich, R. & Schuster, S. (1996): *The regulation of cellular systems*. Chapman & Hall, New York.

Hill, T. L. (1985): *Cooperativity theory in biochemistry : steady-state and equilibrium systems*. Springer series in molecular biology. Springer-Verlag, New York.

Hofestaedt, R. & Thelen, S. (1998): Quantitative modeling of biochemical networks. *In Silico Biol*, **1** 1: 39–53.

Holland, J. (1975): *Adaption in Natural and Artificial Systems*. MIT Press.

Huynen, M., Dandekar, T., & Bork, P. (1998): Differential genome analysis applied to the species-specific features of Helicobacter pylori. *FEBS Lett*, **426** 1: 1–5.

Jacob, F. & Monod, J. (1961): Genetic regulatory mechanisms in the synthesis of proteins. *J Mol Biol*, **3**: 318–356.

Jain, R., Rivera, M. C., & Lake, J. A. (1999): Horizontal gene transfer among genomes: the complexity hypothesis. *Proc Natl Acad Sci U S A*, **96** 7: 3801–6.

Jeong, H., Tombor, B., Albert, R., Oltvai, Z. N., & Barabasi, A. L. (2000): The large-scale organization of metabolic networks. *Nature*, **407** 6804: 651–4.

Jorgensen, P., Nishikawa, J. L., Breitkreutz, B. J., & Tyers, M. (2002): Systematic identification of pathways that couple cell growth and division in yeast. *Science*, **297** 5580: 395–400.

Kanehisa, M. & Goto, S. (2000): Kegg: Kyoto encyclopedia of genes and genomes. *Nucleic Acids Res*, **28** 1: 27–30.

Karp, P., Riley, M., Paley, S., & Pelligrini-Toole, A. (1996): EcoCyc: an encyclopedia of Escherichia coli genes and metabolism. *Nucleic Acids Res*, **24** 1: 32–9.

Kashtan, N., Itzkovitz, S., Milo, R., & Alon, U. (2004): Topological generalizations of network motifs. *Phys Rev E Stat Nonlin Soft Matter Phys*, **70** 3 Pt 1: 031909.

Kauffman, S. (1969): Homeostasis and differentiation in random genetic control networks. *Nature*, **224** 215: 177–8.

Kauffman, S. A. (1993): *The origins of order : self-organization and selection in evolution*. Oxford University Press, New York.

Kepler, T. & Elston, T. (2001): Stochasticity in transcriptional regulation: origins, consequences, and mathematical representations. *Biophys J*, **81** 6: 3116–36.

Kotliar, Baker, Whicker, & Plumb (1999): A Critical Review of Assumptions About the Prairie Dog as a Keystone Species. *Environ Manage*, **24** 2: 177–192.

Koza, J. R. (1992): *Genetic Programming: On the Programming of Computers by Natural Selection*. MIT Press, Cambridge, MA.

Krumholz, L., McKinley, J., Ulrich, G., & Suflita, J. (1997): Confined subsurface microbial communities in cretaceous rock. *Nature*, **386**.

Kunin, V. & Ouzounis, C. A. (2003): The balance of driving forces during genome evolution in prokaryotes. *Genome Res*, **13** 7: 1589–94.

Lambert, J. D. (1991): *Numerical Methods for Ordinary Differential Systems : The Initial Value Problem*. John Wiley & Sons.

Larimer, F. W., Chain, P., Hauser, L., Lamerdin, J., Malfatti, S., Do, L., Land, M. L., Pelletier, D. A., Beatty, J. T., Lang, A. S., Tabita, F. R., Gibson, J. L., Hanson, T. E., Bobst, C., Torres, J. L., Peres, C., Harrison, F. H., Gibson, J., & Harwood, C. S. (2004): Complete genome sequence of the metabolically versatile photosynthetic bacterium Rhodopseudomonas palustris. *Nat Biotechnol*, **22** 1: 55–61.

Lashkari, D., DeRisi, J., McCusker, J., Namath, A., Gentile, C., Hwang, S., Brown, P., & Davis, R. (1997): Yeast microarrays for genome wide parallel genetic and gene expression analysis. *Proc Natl Acad Sci U S A*, **94** 24: 13057–13062.

Lawrence, J. G. & Roth, J. R. (1996): Selfish operons: horizontal transfer may drive the evolution of gene clusters. *Genetics*, **143** 4: 1843–60.

Lee, T. I., Rinaldi, N. J., Robert, F., Odom, D. T., Bar-Joseph, Z., Gerber, G. K., Hannett, N. M., Harbison, C. T., Thompson, C. M., Simon, I., Zeitlinger, J., Jennings, E. G., Murray, H. L., Gordon, D. B., Ren, B., Wyrick, J. J., Tagne, J.-B., Volkert, T. L., Fraenkel, E., Gifford, D. K., & Young, R. A. (2002): Transcriptional regulatory networks in Saccharomyces cerevisiae. *Science*, **298** 5594: 799–804.

Levesque, M., Shasha, D., Kim, W., Surette, M. G., & Benfey, P. N. (2003): Trait-to-gene: a computational method for predicting the function of uncharacterized genes. *Curr Biol*, **13** 2: 129–33.

Liang, S., Fuhrman, S., & Somogyi, R. (1998): Reveal, a general reverse engineering algorithm for inference of genetic network architectures. *Pac Symp Biocomput*, pages 18–29.

Lukashin, A. V. & Borodovsky, M. (1998): Genemark.hmm: new solutions for gene finding. *Nucleic Acids Res*, **26** 4: 1107–15.

Luscombe, N., Babu, M., Yu, H., Snyder, M., Teichmann, S., & Gerstein, M. (2004): Genomic analysis of regulatory network dynamics reveals large topological changes. *Nature*, **431** 7006: 308–312.

Mangalam, H. (2002): The Bio* toolkits–a brief overview. *Brief Bioinform*, **3** 3: 296–302.

Mangan, S. & Alon, U. (2003): Structure and function of the feed-forward loop network motif. *Proc Natl Acad Sci U S A*, **100** 21: 11980–5.

Mangan, S., Zaslaver, A., & Alon, U. (2003): The coherent feedforward loop serves as a sign-sensitive delay element in transcription networks. *J Mol Biol*, **334** 2: 197–204.

Marcotte, E. (2000): Computational genetics: finding protein function by nonhomology methods. *Curr Opin Struct Biol*, **10** 3: 359–65.

Marcotte, E., Pellegrini, M., Ng, H., Rice, D., Yeates, T., & Eisenberg, D. (1999): Detecting protein function and protein-protein interactions from genome sequences. *Science*, **285** 5428: 751–3.

Martin, A. P. & Palumbi, S. R. (1993): Body size, metabolic rate, generation time, and the molecular clock. *Proc Natl Acad Sci U S A*, **90** 9: 4087–91.

Mathe, C., Sagot, M.-F., Schiex, T., & Rouze, P. (2002): Current methods of gene prediction, their strengths and weaknesses. *Nucleic Acids Res*, **30** 19: 4103–17.

Matsuno, H., Doi, A., Nagasaki, M., & Miyano, S. (2000): Hybrid Petri net representation of gene regulatory network. *Pac Symp Biocomput*, pages 341–52.

Mattick, J. (2001): Non-coding RNAs: the architects of eukaryotic complexity. *EMBO Rep*, **2** 11: 986–91.

McAdams, H. & Arkin, A. (1997): Stochastic mechanisms in gene expression. *Proc Natl Acad Sci U S A*, **94** 3: 814–9.

Mendes, P., Sha, W., & Ye, K. (2003): Artificial gene networks for objective comparison of analysis algorithms. *Bioinformatics*, **19 Suppl 2**: II122–II129.

Mendoza, L., Thieffry, D., & Alvarez-Buylla, E. (1999): Genetic control of flower morphogenesis in Arabidopsis thaliana: a logical analysis. *Bioinformatics*, **15** 7-8: 593–606.

Meng, T., Somani, S., & Dhar, P. (2004): Modeling and simulation of biological systems with stochasticity. *In Silico Biol*, **4** 2: 0024–0024.

Meyers, S. & Friedland, P. (1984): Knowledge-based simulation of genetic regulation in bacteriophage lambda. *Nucleic Acids Res*, **12** 1 Pt 1: 1–9.

Milo, R., Itzkovitz, S., Kashtan, N., Levitt, R., Shen-Orr, S., Ayzenshtat, I., Sheffer, M., & Alon, U. (2004): Superfamilies of evolved and designed networks. *Science*, **303** 5663: 1538–42.

Mjolsness, E., Sharp, D. H., & Reinitz, J. (1991): A connectionist model of development. *J Theor Biol*, **152** 4: 429–53.

Murphy, K. & Mian, S. (1999): Modeling gene expression data using dynamic bayesian networks. Technical report, Computer Science Division, University of California, Berkeley, CA.

O'Brien, S. J., Menotti-Raymond, M., Murphy, W. J., Nash, W. G., Wienberg, J., Stanyon, R., Copeland, N. G., Jenkins, N. A., Womack, J. E., & Marshall Graves, J. A. (1999): The promise of comparative genomics in mammals. *Science*, **286** 5439: 458–62, 479–81.

Olsen, G., Woese, C., & Overbeek, R. (1994): The winds of (evolutionary) change: breathing new life into microbiology. *J Bacteriol*, **176** 1: 1–6.

Osman, A. (2004): Yeast two-hybrid assay for studying protein-protein interactions. *Methods Mol Biol*, **270**: 403–22.

Overbeek, R., Fonstein, M., D'Souza, M., Pusch, G., & Maltsev, N. (1999): The use of gene clusters to infer functional coupling. *Proc Natl Acad Sci U S A*, **96** 6: 2896–2901.

Overbeek, R., Larsen, N., Pusch, G., D'Souza, M., Selkov, E., Kyrpides, N., Fonstein, M., Maltsev, N., & Selkov, E. (2000): WIT: integrated system for high-throughput genome sequence analysis and metabolic reconstruction. *Nucleic Acids Res*, **28** 1: 123–5.

Papp, B., Pal, C., & Hurst, L. D. (2004): Metabolic network analysis of the causes and evolution of enzyme dispensability in yeast. *Nature*, **429** 6992: 661–4.

Parra, G., Agarwal, P., Abril, J. F., Wiehe, T., Fickett, J. W., & Guigo, R. (2003): Comparative gene prediction in human and mouse. *Genome Res*, **13** 1: 108–17.

Pe'er, D., Regev, A., Elidan, G., & Friedman, N. (2001): Inferring subnetworks from perturbed expression profiles. *Bioinformatics*, **17 Suppl 1**: S215–24.

Pellegrini, M., Marcotte, E. M., Thompson, M. J., Eisenberg, D., & Yeates, T. O. (1999): Assigning protein functions by comparative genome analysis: protein phylogenetic profiles. *Proc Natl Acad Sci U S A*, **96** 8: 4285–8.

Pennisi, E. (2001): The human genome. *Science*, **291** 5507: 1177–80.

Qian, J., Lin, J., Luscombe, N. M., Yu, H., & Gerstein, M. (2003): Prediction of regulatory networks: genome-wide identification of transcription factor targets from gene expression data. *Bioinformatics*, **19** 15: 1917–26.

Rao, C. V., Wolf, D. M., & Arkin, A. P. (2002): Control, exploitation and tolerance of intracellular noise. *Nature*, **420** 6912: 231–7.

Raymond, J., Zhaxybayeva, O., Gogarten, J. P., Gerdes, S. Y., & Blankenship, R. E. (2002): Whole-genome analysis of photosynthetic prokaryotes. *Science*, **298** 5598: 1616–20.

Rechenberg, I. (1973): *Evolutionsstrategie; Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, [Stuttgart-Bad Cannstatt].

Regev, A., Silverman, W., & Shapiro, E. (2001): Representation and simulation of biochemical processes using the pi-calculus process algebra. *Pac Symp Biocomput*, pages 459–70.

Reichard, K. & Kaufmann, M. (2003): Epps: mining the cog database by an extended phylogenetic patterns search. *Bioinformatics*, **19** 6: 784–5.

Repsilber, D. & Kim, J. T. (2003): Developing and testing methods for microarray data analysis using an artificial life framework. In Banzhaf, W., Christaller, T., Dittrich, P., Kim, J. T., & Ziegler, J., editors, *Advances in Artificial Life, Proceedings of European Conference of Artificial Life (ECAL) 2003*, pages 686–695. Springer–Verlag.

Rieger, G., Rachel, R., Hermann, R., & Stetter, K. (1995): Ultrastructure of the hyperthermophilic archaeon pyrodictium abyssi. *J Struct Biol*, **115** 1: 78 – 87.

Rivera, M., Jain, R., Moore, J., & Lake, J. (1998): Genomic evidence for two functionally distinct gene classes. *Proc Natl Acad Sci U S A*, **95** 11: 6239–44.

Roughgarden, J. (1977): Density dependent natural selection. In *Theory of Population Genetics and Evolutionary Ecology*, pages Chapter 17, pp 311–319. MacMillan, NY.

Rueckert, C. (2004): Personal communication. At the time of writing, Chistian Rueckert was a molecular biologist working on the sulfur metabolism pathway in *C. glutamicum*.

Salazar-Ciudad, I., Garcia-Fernandez, J., & Sole, R. V. (2000): Gene networks capable of pattern formation: from induction to reaction-diffusion. *J Theor Biol*, **205** 4: 587–603.

Salgado, H., Gama-Castro, S., Martnez-Antonio, A., Daz-Peredo, E., Snchez-Solano, F., Peralta-Gil, M., Garcia-Alonso, D., Jimnez-Jacinto, V., Santos-Zavaleta, A., Bonavides-Martnez, C., & Collado-Vides, J. (2004): RegulonDB (version 4.0): transcriptional regulation, operon organization and growth conditions in Escherichia coli K-12. *Nucleic Acids Res*, **32 Database issue**: D303–6.

Salzberg, S. L., Delcher, A. L., Kasif, S., & White, O. (1998): Microbial gene identification using interpolated markov models. *Nucleic Acids Res*, **26** 2: 544–8.

Savageau, M. A. (1998): Rules for the evolution of gene circuitry. *Pac Symp Biocomput*, pages 54–65.

Schena, M., Shalon, D., Davis, R. W., & Brown, P. O. (1995): Quantitative monitoring of gene expression patterns with a complementary dna microarray. *Science*, **270** 5235: 467–70.

Schubert, W. (2003): Topological proteomics, toponomics, MELK-technology. *Adv Biochem Eng Biotechnol*, **83**: 189–209.

Schwefel, H.-P. (1981): *Numerical optimization of computer models*. Wiley, Chichester ; New York.

Shen-Orr, S. S., Milo, R., Mangan, S., & Alon, U. (2002): Network motifs in the transcriptional regulation network of escherichia coli. *Nat Genet*, **31** 1: 64–8.

Snel, B., Bork, P., & Huynen, M. (1999): Genome phylogeny based on gene content. *Nat Genet*, **21** 1: 108–10.

Stark, J., Brewer, D., Barenco, M., Tomescu, D., Callard, R., & Hubank, M. (2003): Reconstructing gene networks: what are the limits? *Biochem Soc Trans*, **31** Pt 6: 1519–25.

Strogatz, S. H. (2001): Exploring complex networks. *Nature*, **410** 6825: 268–76.

Stuart, J., Segal, E., Koller, D., & Kim, S. (2003): A gene-coexpression network for global discovery of conserved genetic modules. *Science*, **302** 5643: 249–255.

Snchez, L. & Thieffry, D. (2003): Segmenting the fly embryo: a logical analysis of the pair-rule cross-regulatory module. *J Theor Biol*, **224** 4: 517–37.

Taher, L., Rinner, O., Garg, S., Sczyrba, A., & Morgenstern, B. (2004): AGenDA: gene prediction by cross-species sequence comparison. *Nucleic Acids Res*, **32** Web Server issue: W305–8.

Tamames, J., Casari, G., Ouzounis, C., & Valencia, A. (1997): Conserved clusters of functionally related genes in two bacterial genomes. *J Mol Evol*, **44** 1: 66–73.

Tamas, I., Klasson, L., Canback, B., Naslund, A. K., Eriksson, A. S., Wernegreen, J. J., Sandstrom, J. P., Moran, N. A., & Andersson, S. G. (2002): 50 million years of genomic stasis in endosymbiotic bacteria. *Science*, **296** 5577: 2376–9.

Tatusov, R. L., Fedorova, N. D., Jackson, J. D., Jacobs, A. R., Kiryutin, B., Koonin, E. V., Krylov, D. M., Mazumder, R., Mekhedov, S. L., Nikolskaya, A. N., Rao, B. S., Smirnov, S., Sverdlov, A. V., Vasudevan, S., Wolf, Y. I., Yin, J. J., & Natale, D. A. (2003): The COG database: an updated version includes eukaryotes. *BMC Bioinformatics*, **4** 1: 41.

Tatusov, R. L., Galperin, M. Y., Natale, D. A., & Koonin, E. V. (2000): The COG database: a tool for genome-scale analysis of protein functions and evolution. *Nucleic Acids Res*, **28** 1: 33–6.

Tatusov, R. L., Koonin, E. V., & Lipman, D. J. (1997): A genomic perspective on protein families. *Science*, **278** 5338: 631–7.

Tatusov, R. L., Natale, D. A., Garkavtsev, I. V., Tatusova, T. A., Shankavaram, U. T., Rao, B. S., Kiryutin, B., Galperin, M. Y., Fedorova, N. D., & Koonin, E. V. (2001): The COG database: new developments in phylogenetic classification of proteins from complete genomes. *Nucleic Acids Res*, **29** 1: 22–8.

Tavazoie, S., Hughes, J. D., Campbell, M. J., Cho, R. J., & Church, G. M. (1999): Systematic determination of genetic network architecture. *Nat Genet*, **22** 3: 281–5.

Thattai, M. & van Oudenaarden, A. (2001): Intrinsic noise in gene regulatory networks. *Proc Natl Acad Sci U S A*, **98** 15: 8614–9.

Thieffry, D. & Thomas, R. (1995): Dynamical behaviour of biological regulatory networks–II. Immunity control in bacteriophage lambda. *Bull Math Biol*, **57** 2: 277–97.

Thomas, R. (1991): Regulatory networks seen as asynchronous automata: A logical description. *J Theor Biol*, **153**: 1–23.

Thomas, R. & D'Ari, R. (1990): *Biological feedback*. CRC Press, Boca Raton.

Tian, T. & Burrage, K. (2004): Stochastic neural network models for gene regulatory networks. *Unpublished*.

Tominaga, D., Okamoto, M., Maki, Y., Watanabe, S., & Eguchi, Y. (1999): Nonlinear numerical optimization technique based on a genetic algorithm for inverse problems: Towards the inference of genetic networks. In *German Conference on Bioinformatics (GCB)*. http://www.bioinfo.de/isb/gcb99/talks/tominaga/.

Uetz, P., Giot, L., Cagney, G., Mansfield, T., Judson, R., Knight, J., Lockshon, D., Narayan, V., Srinivasan, M., Pochart, P., Qureshi-Emili, A., Li, Y., Godwin, B., Conover, D., Kalbfleisch, T., Vijayadamodar, G., Yang, M., Johnston, M., Fields, S., & Rothberg, J. (2000): A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae. *Nature*, **403** 6770: 623–7.

Velculescu, V., Zhang, L., Vogelstein, B., & Kinzler, K. (1995): Serial analysis of gene expression. *Science*, **270** 5235: 484–487.

Vilar, J. M., Kueh, H. Y., Barkai, N., & Leibler, S. (2002): Mechanisms of noise-resistance in genetic oscillators. *Proc Natl Acad Sci U S A*, **99** 9: 5988–92.

Voit, E. O. (2000): *Computational analysis of biochemical systems : a practical guide for biochemists and molecular biologists*. Cambridge University Press, New York.

von Dassow, G., Meir, E., Munro, E. M., & Odell, G. M. (2000): The segment polarity network is a robust developmental module. *Nature*, **406** 6792: 188–92.

Wahde, M. & Hertz, J. (2000): Coarse-grained reverse engineering of genetic regulatory networks. *Biosystems*, **55** 1-3: 129–36.

Weaver, D. C., Workman, C. T., & Stormo, G. D. (1999): Modeling regulatory networks with weight matrices. *Pac Symp Biocomput*, pages 112–23.

Williams, R. & Martinez, N. (2000): Simple rules yield complex food webs. *Nature*, **404** 6774: 180–3.

Wilson, E. O. (1998): *Consilience p.85*. Knopf, New York.

Woese, C. R. (2000): Interpreting the universal phylogenetic tree. *Proc Natl Acad Sci U S A*, **97** 15: 8392–6.

Woese, C. R. & Fox, G. E. (1977): Phylogenetic structure of the prokaryotic domain: The primary kingdoms. *Proc Natl Acad Sci USA*, **74**: 5088–5090.

Xiong, J., Fischer, W. M., Inoue, K., Nakahara, M., & Bauer, C. E. (2000): Molecular evidence for the early evolution of photosynthesis. *Science*, **289** 5485: 1724–30.

Yagil, G. (1975): *Current topics in cellular regulation*, chapter Quantitative aspects of protein induction, pages 183–237. Academic Press, New York. ISBN 0070-2137.

Yagil, G. & Yagil, E. (1971): On the relation between effector concentration and the rate of induced enzyme synthesis. *Biophys J*, **11** 1: 11–27.

Zielenski, J., Rozmahel, R., Bozon, D., Kerem, B., Grzelczak, Z., Riordan, J., Rommens, J., & Tsui, L. (1991): Genomic DNA sequence of the cystic fibrosis transmembrane conductance regulator (CFTR) gene. *Genomics*, **10** 1: 214–28.

Zweig, M. & Campbell, G. (1993): Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine. *Clin Chem*, **39** 4: 561–577.