A systems biology design and implementation of novel bioinformatics software tools for high throughput gene expression analysis

Mohsin Amir Faiz Khan

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor of Philosophy

Brunel University

March 2009

Institute of Cancer Genetics & pharmacogenomics,
School of Health Sciences

Declaration

The research reported in this thesis is my own work except where otherwise stated, and has not been submitted for any other degree.

Mohsin Amir Faiz Khan

Dedication

To my parents..

Acknowledgements

I will preface this by saying that this doctorate degree has given me much more than I ever imagined. I can clearly remember the day when I began my journey as a research student at Brunel University, and can't help but laugh at my naivety at the time because I subscribed to a belief that the PhD would not be significantly different to any other degree program. I have never been so wrong in my life! What made this experience so unique for me revolves around the fact that this degree has prepared me for life, academically, spiritually, emotionally and psychologically. In addition, it has had a tremendously positive impact on my character. I can remember being short tempered and idealistic, and now that I look back at those times, I have come to realize that research has taught me much more than publishing papers, reading articles and delivering presentations. It has taught me humility and resilience. How could it not? After all, anybody who has gone through this experience knows very well that in research, things seldom happen when you want them to. The unprecedented up's and down's make us realize that the results are not within our control, but more importantly that we must still continue to do our best regardless of how the situation may seem at the time. There are some very important people in my life who have helped me reach my goal of completing this doctorate degree. Even though I may have severed my relations with a few of them, I feel the need to acknowledge them.

I would like to firstly thank my dear supervisor Dr Su-Ling Li, who has been a fantastic support system to me throughout my studies as a research student. I sometimes feel that she offered much more support than what was required of her. Not only did she teach me about the good, the bad, and the ugly sides of research but she was always there for me at time of need. My success as a doctor is reflective of her success as a supervisor. I would also like to sincerely thank my second supervisor, Professor XiaoHui Liu who encouraged me to do well.

Another important person in my life without whom I could not have reached where I am, is my father Dr Wazir Ali Khan who was a tremendous support to me. He never hesitated to offer his financial support when required and was always supportive of my ambitions. I hope I have made you proud Dad! I would also like to acknowledge my mother Nasim Akhtar Khan who kept me healthy (or fat) with her delicious cuisines during the course of my studies. Thanks Mum! Moving on to my terrific colleagues, I would like to firstly thank Dr Lylia Ouboussad who was, still is and always will be, a great friend to me. I would also like to acknowledge Dr Sabah Khalid who collaborated with me on certain aspects of my research. And of course, I owe a sincere thank you to my good friend and collaborator Chandrasekhar Babu Gorle, who contributed well towards my projects. Also, a big thank you to Rana Hassan, Ricardo Mouro Pinto, Chiranjeevi Sandi and Dr Sahar Al-Mahdawi.

Siblings are indeed a pain, but they can sometimes be a pain and be supportive concurrently. This holds true for mine. So a massive thank you to my siblings, Saira Yasmin Khan, Dr Abid Raza Khan, Freda Mushtaq Khan, and Sadia Shahnaz Khan. Also a big thank you to my brother-in-laws, Khalid Farooq Khan, Ashraf El-Shazly and Mohsin Ullah Khan. Thanks for being there for me guys!

Tackling the PhD hurdle requires an escape from work, especially at times when we are on the verge of “burning out”. Hence, I would like to acknowledge my nephews and nieces who played a major part in helping me escape from research when it was necessary. So a massive thank you goes to Shabana Qurban Khan from University College London for using her excellent critical reading skills to proofread this thesis. Also, an equally massive thank you goes to Irfan Mustafa Khan, Saba Farooq Khan, Salman Akbar Khan, Aishah Qurban Khan, Omar El-Shazli, Maryam Khan, Ziad Al-Shazli, Hassan Farooq Khan and Malaika Khan.

Last but not least, I would like to express my sincere thank you to Dr Mark Stahr Taylor for offering his support and words of encouragement.

If I have missed anybody out, please forgive me and know that it was not intentional.

Abstract

Microarray technology has revolutionized the field of molecular biology by offering an efficient and cost effective platform for the simultaneous quantification of thousands of genes or even entire genomes in a single experiment. Unlike southern blotting, which is restricted to the measurement of one gene at-a-time, microarrays offer biologists with the opportunity to carry out genome-wide experiments in order to help them gain a systems level understanding of cell regulation and control. The application of bioinformatics in the milieu of gene expression analysis has attracted a great deal of attention in the recent past due to specific algorithms and software solutions that attempt to illustrate complex multidimensional microarray data in a biologically coherent fashion so that it can be understood by the biologist. This has given rise to some exciting prospects for deciphering microarray data, by helping us refine our comprehension pertinent to the underlying physiological dynamics of disease.

Although much progress is being made in the development of specialized bioinformatics software pipelines with the purpose of decoding large volumes of gene expression data in the context of systems biology, several loopholes exist. Perhaps most notable of these loopholes is the fact that there is an increasing demand for software solutions that specialize in automating the comparison of multiple gene expression profiles, derived from microarray experiments sharing a common biological theme. This is no doubt an important challenge, since common genes across different biological conditions having similar expression patterns are likely to be involved in the same biological process and hence, may share the same regulatory signatures. The potential benefits of this in refining our understanding of the physiology of disease are undeniable.

The research presented in this thesis provides a systematic walkthrough of a series of software pipelines developed for the purpose of streamlining gene expression analysis in a systems biology context. Firstly, we present BiSAn, a software tool that deciphers expression data from the perspective of transcriptional regulation. Following this, we present Genome Interaction Analyzer (GIA), which analyzes microarray data in the integrative framework of transcription factor binding sites, protein-protein interactions and molecular pathways. The final contribution is a software pipeline called MicroPath, which analyzes multiple sets of gene expression profiles and attempts to extract common regulatory signatures that may be implicating the biological question.

“Problems worthy of attack prove their worth by fighting back”

Paul Erdos

Contents

List of Figures

List of Tables

Supporting Publications

Journal Papers (Published or in Press):

I. **Khan, M**; Gorle, CB; Fraser, K; Wang, P; Liu, X. and Li, SL. MicroPath - A pathway-based pipeline for the comparison of multiple gene expression profiles to identify common biological signatures. *(In Press)*

II. **Khan, M**; Gorle, CB; Fraser, K; Wang, P; Liu, X. and Li, SL. BiSAn: A software for efficient computation of transcription factor binding motifs for high throughput gene expression analysis. *(In Press)*

III. Khalid, S., Fraser, K., **Khan, M.,** Wang, P., Liu, X. and Li, S. Analysing Microarray Data using the Multi-functional Immune Ontologiser. *Journal of Integrative Bioinformatics* 3: 2006

IV. Khalid, S., **Khan, M.,** Symonds, A., Fraser, K., Wang, P., Liu, X. and Li, S. CIDA: An integrated software for the design, characterisation and global comparison of microarrays. *Journal of Integrative Bioinformatics* 4(3):78: 2007

V. Khalid, S., **Khan, M.,** Gorle, CB; Fraser, K., Wang, P., Liu, X. and Li, S. MaXlab: A novel application for the cross comparison and integration of biological signatures from microarray studies. *In Silico Biology* 8, 0029: 2008

Manuscripts (Submitted or In Preparation):

I. **Khan, M**; Khalid, S; Gorle, CB; Fraser, K; Wang, P; Liu, X. and Li, SL. Genome Interactions Analyser: a systems biology approach for the global analysis of transcriptional networks in microarray data. *(Re-submission)*

II. **Khan, M**; Khalid, S; Gorle, CB; Fraser, K; Wang, P; Liu, X. and Li, SL. MiNer: A tool for the comparison of multiple gene expression studies to identify cellular transcriptional states. *(Re-submission)*

III. **Khan, M.,** Khalid, S., Symonds, A., Gorle, C., Wang, P., Liu, X. and Li, S. Deciphering microarray data to reveal transcription factors involved in regulating immune tolerance. *(Manuscript)*

Refereed conferences:

I. Khalid, S., **Khan, M.,** Wang, P., Liu, X. and Li, S. (2006) Application of Bioinformatics in the design of gene expression microarrays. *IEEE proceedings,* 163-177 (*ISOLA Conference)*

II. **Khan, M**; Khalid, S; Symonds, A; Gorle, CB; Fraser, K; Wang, P; Liu, X; Li, SL. *A systems biology approach for the global analysis of transcriptional networks in gene expression data.* International Symposium on Integrative Bioinformatics, 5th annual meeting, Leucorea, Lutherstadt Wittenberg, Germany, August 2008

III. **Khan, M**; Khalid, S; Gorle, CB; Fraser, K; Wang, P; Liu, X; Li, SL. *MiNeR: A tool for the comparison of multiple gene expression studies to identify cellular transcriptional states.* International Symposium on Integrative Bioinformatics, 5th annual meeting, Leucorea, Lutherstadt Wittenberg, Germany, August 2008

Chapter 1
Introduction

1

1.1 Overview

The completion of the Human Genome Project (HGP) in 2003 successfully resulted in sequencing the first human genome, which is representative of the complete set of genes present in the cell. Initiated by James D. Watson at the National Institutes of Health – USA in 1990, the goal of the HGP was to determine the precise sequence of base pairs in DNA that represented genes within the genome. Furthermore, the focal point of the project was to identify an anticipated 25,000 genes pertinent to the human genome from both a physical and functional perspective. Although the HGP is now complete from the sequencing standpoint, one is left to wonder whether this initiative was just a preparation for a massive biological jigsaw puzzle. After all, the resulting sequenced human genome has spawned challenges to relate the genomic sequences to specific biological functions in order to better understand biological systems and to enhance our comprehension of the underlying dynamics of disease. Furthermore, although biological functions have already been allocated to thousands of genes, merely knowing their biological roles is not sufficient to warrant an understanding of the biological networks that govern cellular behaviour. This is because gene function and behaviour is governed by complex molecular interactions and dependencies, which are intertwined in a cellular network that determines the overall fate of the cell. For this reason, gene function and behaviour needs to be understood in the context of the organism as a single unit, which at the same time, is capable of hosting complex inter-dependent biological interactions.

As means to provide a platform for molecular biologists to further understand the biological functions of known genes and discover those of unknown genes, the scientific and research communities have embraced a valuable high throughput laboratory based technique called microarray technology. Using this technology, molecular biologists can simultaneously quantify gene expression levels for thousands of genes or even entire genomes at once for any given biological question. Vital to its widespread popularity, microarray technology is regarded as a superior technology due to its biological usefulness and high throughput nature. Unlike traditional low throughput molecular biology techniques such as southern blotting, which are limited to the quantification of gene expression on a gene-by-gene basis, microarray technology offers a platform for putative biological discoveries to be made at an efficient pace, making it feasible to readily identify the potential involvement of genes in several biological processes. Although the benefits of microarray technology cannot be denied, it is also undeniable that the technology is not perfect, which is largely due to the colossal amounts of data that it generates, attributed to the *knowledge Vs data paradox*. To elaborate, the goal of microarray technology is to speed up the process of quantifying gene expression in order to extract meaningful biological knowledge, which paradoxically becomes a problem when too much raw data is generated from the use of the technology. With vast magnitudes of generated raw biological data, how does one go about mining the data to find answers to the original biological question of the experiment? The answer lies in bioinformatics.

Bioinformatics is a fast growing inter-disciplinary science, which applies computer technology to biological data in order to manage it. More specifically, it can be defined as a science that integrates informatics, mathematics, computer algorithms and statistics together to solve complex biological problems. The application of bioinformatics in the milieu of gene expression analysis has attracted a great deal of attention in the recent past due to specific algorithms and software solutions that attempt to illustrate complex multidimensional microarray data in a biologically coherent fashion so that it can be understood by the biologist (Slonim, 2002; Spang, 2003). This has indeed given rise to some exciting prospects for deciphering microarray data in order to aid the enhancement of our comprehension pertinent to the underlying physiological dynamics of disease. However, the advent of such an approach has also highlighted some important challenges. Perhaps the most notable challenge stems from the reality that biological data are disseminated in different laboratories, each having different management systems, file formats and methods for representing their biological data (Battistella *et al*, 2005). The heterogeneity of the data poses a problem for researchers due to a lack of cohesive vocabulary, which in turn hampers the progress of novel biological discoveries. For this reason, the attention is now shifting towards biological data integration.

The premise of this PhD study is based on the design, development and implementation of bioinformatics software using a data integration approach, all in the context of systems biology. Scrutinizing the relevant scientific literature reveals a soaring need for specialised software tools that cater for the high throughput analysis of microarray data in the perspective of systems biology. The approach proposed in this study is to integrate data at the DNA transcriptional level (transcription factor related data) with protein and molecular pathway data under a series of automated pipelines designed to facilitate microarray data analysis according to the users' specific needs.

1.2 Systems biology

The paradigm of systems biology is motivated by the fact that biological sciences is no longer a self sufficient discipline because it has become highly reliant on informatics, mathematics, computer algorithms, software development and statistics. It is hence no surprise that biology has amalgamated with the aforementioned disciplines, giving rise to a new era of opportunities for novel biological discoveries. Systems biology is a new research area that employs a multidisciplinary approach to look beyond the function of a single gene. As stated by Butcher *et al* (2004), *"The goal of modern systems biology is to understand physiology and disease from the level of molecular pathways, regulatory networks, cells, tissues, organs and ultimately the whole organism".* It is hence clear that systems biology is a holistic approach (Chong and Ray, 2002), which aims to facilitate a systems level understanding of biology by integrating knowledge pertaining to various different components that collectively determine the overall biology of the organism. This is an effective strategy since it encourages a scrutiny of the structure and dynamics of cellular function in the context of the organism as a whole rather than examining the characteristics of isolated parts of the cell or organism (Kitano, 2002).

With the advent of microarray technology, the focus of biological data interpretation has led to a rapid paradigm shift in molecular biology. Because a single microarray experiment typically generates several thousands of gene expression data points, it is becoming clear that there is great potential for novel biological discoveries provided the data is deciphered in the context of systems biology. In order to progress in this endeavour, it is essential to move to the systemic picture whilst gaining a deeper understanding at the molecular and biological levels (Kitano, 2002). Although systems biology is considered a new research discipline, the notion of integrative thinking is not new to molecular biology since the first molecular regulatory circuits were mapped out over 40 years ago (Westerhoff, 2004) (Figure 1.1).

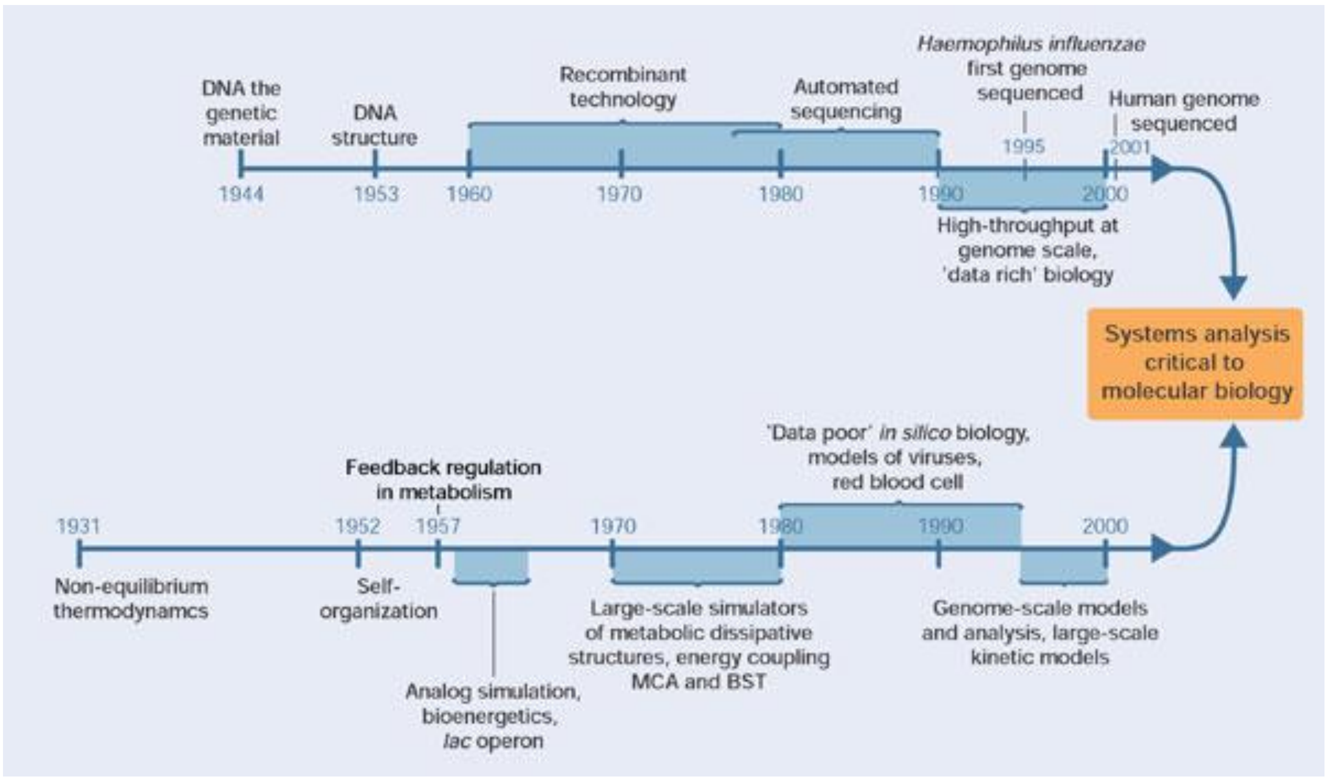

DNA the genetic material
DNA structure
Recombinant technology
Automated sequencing
Haemophilus influenzae first genome sequenced
Human genome sequenced
1944
1953
1960
1970
1980
1990
1995
2000
2001
High-throughput at genome scale, 'data rich' biology
Systems analysis critical to molecular biology
'Data poor' *in silico* biology, models of viruses, red blood cell
Feedback regulation in metabolism
1931
1952
1957
1970
1980
1990
2000
Non-equilibrium thermodynamcs
Self-organization
Large-scale simulators of metabolic dissipative structures, energy coupling MCA and BST
Genome-scale models and analysis, large-scale kinetic models
Analog simulation, bioenergetics, *lac* operon

Figure 1.1 An illustration of how two lines of enquiry led to the birth of present-day systems biology. The upper timeline reflects progress in molecular biology over the past years leading to systems biology. The lower timeline shows the formal analysis of functional states that arise when several molecules interact concurrently. (Taken from Westerhoff, 2004).

The importance of systems biology becomes apparent when we examine the current situation with drug discovery. Despite the fact that an astronomical amount of investment has taken place over the past 20 years towards screening technologies and genomics, the truth remains that the costs associated with new drug discovery continue to rise while approval rates fall. This is because the desire to effectively mine the genome has met paths with the realization that merely knowing a target is not sufficient to warrant an understanding of what the target does, let alone knowing the effects of a chemical inhibitor in diverse disease settings (Butcher, 2004). This emphasizes the need for an effective approach to integrate biological data in the form of automated pipelines that can be used to facilitate biological predictions in the context of high throughput gene expression data analysis.

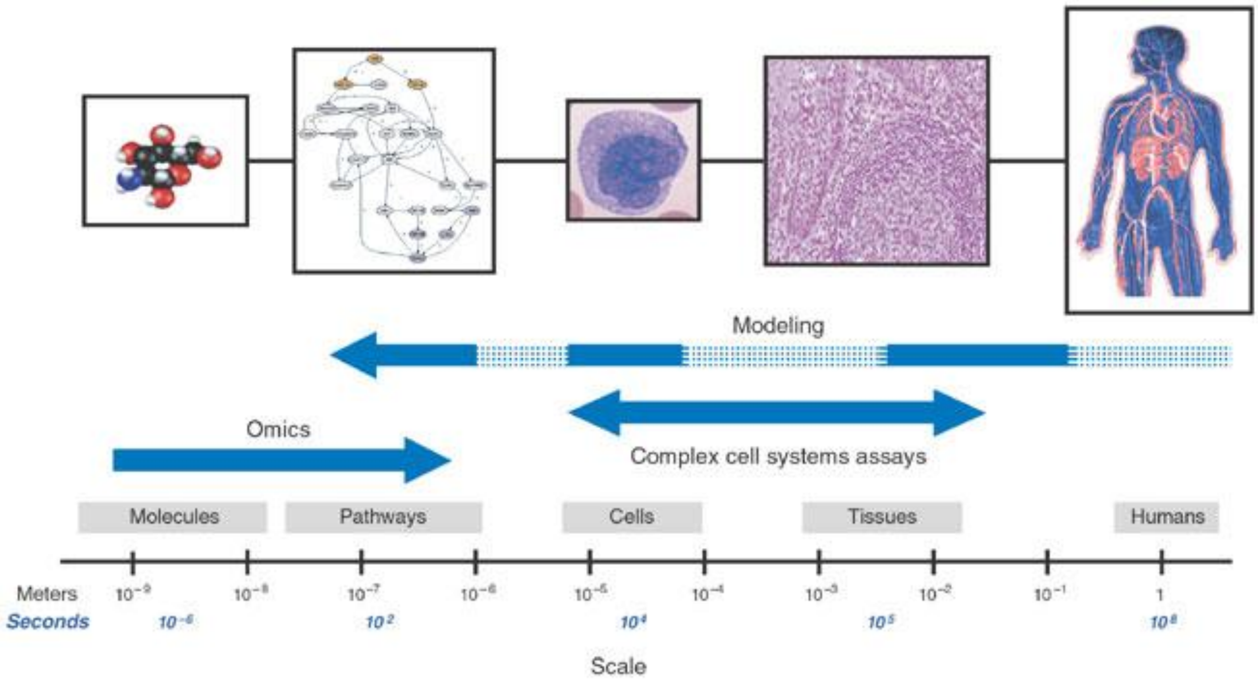


Figure 1.2 The application of systems biology in the pharmaceutical industry. Omics represents a bottom-up approach towards the identification of components at the molecular level (molecules and pathways). Modelling takes a top-down approach by starting from human physiology and disease. (Taken from Butcher, 2004).

The study of Omics plays a central role in systems biology and drug discovery. Omics is a specific term used to describe a broad research discipline focusing on the analysis of biological interactions that take place within various *Omes* or biolayers. Omics approaches to systems biology have been wholeheartedly accepted by the pharmaceutical industry in order to complement traditional approaches to target identification, generate hypotheses and for experimental analysis (Butcher, 2004). The strength of Omics lies in its use in generating potentially important hypothesis. For instance, it can be used to answer specific biological questions pertinent to high throughput experimental studies, which may be conducted in order to correlate a given disease state to the expression of specific genes or proteins (Figure 1.2). This has given rise to some very exciting prospects for treating disease and this is precisely the reason why there is a soaring desire for effective computational solutions capable of deciphering colossal amounts of gene expression data in the milieu of systems biology.

1.3 Loopholes in existing works and contribution to knowledge

The current situation is such that there is a great deal of research being conducted in the area of systems biology, where several software tools have been generated for the purpose of deciphering high throughput gene expression data using a number of different methodologies. However, there is much to be desired and several loopholes are yet to be accommodated. The intensity of such attempts is motivated by the fact that a paradigm shift has occurred from a reductionist approach to an integrative one. As stated by Aggarwal (2003), *"The reductionist approach to biological research, which has been extremely important to the development of a basic understanding of living system, is geared towards identifying the individual components (genes, proteins, metabolites etc) responsible for a particular phenomenon in an organism (e.g. metabolic activity, response to external stimuli etc). This approach has proven effective at elucidating key molecular components of living systems, leading to a variety of important applications in agriculture and medicine. It is now clear however, that information at only one level (the genome or the proteome, for example) by itself cannot fully explain the behaviour of any particular biological system."* In light of this, the field of systems biology is facing a great deal of pressure to devise effective integrative strategies to enhance our understanding of the physiology of disease states.

In the past, a number of software have been developed for the purpose of automating a systems level analysis of microarray data such as Expander (Shamir, 2005), INCLUSive (Thijs, 2002), Genesis (Sturn, 2002), CONFAC (Karanam, 2004), and GEPAS (Herrero, 2003). These pipelines specialise in streamlining the analysis flow by performing specific functions such as K-means clustering, promoter and functional analysis etc. This is no doubt an important step towards deciphering high throughput expression data in a systems biology context. However, further integrative strategies are required to move forward in this endeavour, which should complement a biologically valid understanding of how a cell functions as a whole.

It is important to understand that the overall fate of a cell is determined by three principle classes of interactions, 1) the binding between genomic DNA and specific DNA-binding proteins called transcription factors (TF's), via regulatory binding sites, 2) participation of Protein-protein interactions between two separate protein molecules in the cellular cytoplasm and finally, 3) the cascade of molecular and signalling events that dictate the behaviour of a given molecular pathway. A detailed explanation of these classes of interactions is provided in Chapter 2 of this thesis. There is currently a need for specialised software tools that cater for the analysis of microarray data under the integrative framework of the aforementioned classes of interactions. This is an important step since the goal of systems biology is to understand molecular interactions in the perspective of the organism itself.

Furthermore, because high throughput technologies such as microarrays are rapidly gaining popularity at a global scale due to the prospect of efficiently quantifying gene expression in a high throughput fashion in order to identify previously unknown transcriptome states, expression data related to various different biological questions are being readily generated by scientists worldwide. Such data sets are continuously being uploaded to public data repositories such as ArrayExpress (Sarkans, 2005) and the Gene Expression Omnibus (GEO) (Edgar, 2002), which are subsequently made accessible to the public. This has opened a gateway for biologists to utilise these sets of data in an attempt to investigate common regulatory signatures that may be responsible for dictating biological questions sharing a similar biological theme. One of the most common methods of comparison is based on the assumption that genes across different biological conditions exhibiting similar expression patterns are likely to be involved in the same biological process (Barrett, 2006) and may therefore, share common regulatory signatures. By using this method of comparison, which is one of the most successful methods to date, coupled with the availability of publicly available data repositories offering gene expression data, biologists have been granted the opportunity to answer complex biological questions with regards to biological phenomena underlying various different disease states. When one scrutinizes current literature relevant to automated solutions of gene expression analysis, it becomes apparent that there is an increasing demand for software applications that offer an efficient pipeline to the analysis of multiple gene expression profiles.

In light of this, the research presented in this thesis is based on the following contributions to knowledge:

- *Identification of existing loopholes in systems biology from a software development perspective*
- *Design, development and implementation of a bioinformatics software tool specializing in deciphering gene expression data at the transcriptional level, using Position Frequency Matrices (PFM's)*
- *Design, development and implementation of a bioinformatics software pipeline to facilitate the automation of gene expression analysis in the context of transcription factors, protein-protein interactions and molecular pathway analyses*

- *Further development of a novel bioinformatics software pipeline to incorporate the analysis of multiple gene expression profiles in order to identify common regulatory signatures*
- *Proposition of a novel blueprint for future prospects of treating disease*

1.4 Roadmap

In this section, the structure of the thesis is described together with a brief summary of what each chapter comprises of.

Chapter 2 is concerned with providing a review of microarray technology covering aspects such as laboratory procedures, raw data generation, differential gene analysis etc. Prior to this, the three principal classes of molecular interactions are explained, which involves the science behind how transcription factors interact with genomic DNA, how proteins interact with one another in the cytoplasm and how molecular pathways are regulated in order to collectively govern overall cellular behaviour. Finally, current literature pertaining to the subject of bioinformatics software for microarray data analysis is scrutinized to act as the premise for the subsequent chapters.

Chapter 3 presents a bioinformatics software called BiSAn, which has been specifically designed, developed and implemented for high throughput gene expression analysis by using Position Frequency Matrices (PFMs) to scan promoter sites for the presence of Transcription Factor Binding Motifs (TFBMs). This work represents the first contribution of this PhD study.

Chapter 4 introduces an integrative strategy to microarray data analysis in the form of a software pipeline called Genome Interactions Analyzer (GIA), which deciphers gene expression data in the context of transcription factor binding sites, protein-protein interactions and molecular pathways.

Chapter 5 highlights an important loophole in the area of systems biology by emphasizing on the necessity to analyze biologically related multiple gene expression profiles, which is an essential requirement for better understanding the physiology of disease. The contribution presented in this chapter takes the form of a novel algorithm and software pipeline, developed to facilitate multiple analyses of gene expression data in a user friendly and automated manner. Relevant statistical tests are also applied and the pipeline's faculty to demonstrate multiple gene expression analysis from the perspective of pathway analysis is presented.

Chapter 6 concludes this thesis by offering a rationale for effectively treating disease, from the perspective of *in silico* studies to *in vivo* laboratory experiments.

Chapter 2 Background

2

2.1 Introduction

Systems biology is a newly established multi-disciplinary science, which integrates Computer Science, Mathematics, Informatics, and Statistics in order to offer a holistic approach to facilitate a systems level understanding of biology by understanding physiology and disease from the level of molecular pathways, regulatory networks, cells, tissues, organs and ultimately the whole organism (Chong and Ray, 2002; Butcher, 2004). The premise of the research presented in this thesis is based on the development of specialized bioinformatics software in the context of systems biology. As this area is constantly evolving, a single definition does not suffice when attempting to specify its objectives in their entirety. Hence, the purpose of this chapter is to provide the reader with a systematic walkthrough of specific biological principles that form the backbone of systems biology, which are imperative to warrant an understanding of the contributions presented in this thesis. Firstly, the reader is provided with an explanation of DNA and the paradigm of gene expression. Next, the three principle classes of molecular interactions that collectively govern overall cellular behaviour is described followed by an overview of microarray technology. Finally, literature pertinent to microarray data analysis from the perspective of computational biology and bioinformatics is scrutinized to summarize the work that has been carried out till present day.

2.2 Explanation of biological terms

2.2.1 DNA

Deoxyribonucleic Acid or DNA is the genetic blueprint of life that contains precise instructions that dictate the structure and functions of cells. Abundantly located in the nucleus of the cell, DNA exists as a hereditary code comprising of four specific chemical bases namely, Adenine (A), Guanine (G), Thymine (T) and Cytosine (C). In humans, the total DNA within the nucleus of the cell consists of approximately 3 million of these bases and it is the order or the sequence of these bases that governs the accuracy of the information available for cell structure and function. To understand the importance of the precise sequence of DNA in controlling cell behaviour, consider a scenario where you are giving someone instructions in English to perform any given task. The precise sequence of letters from the English alphabet used to construct words will determine the clarity of the instructions. Similarly, the sequence of bases in DNA determines the accuracy of the instructions.

Almost every cell of a living organism contains the exact same DNA so why do cells behave differently to one another? This is because only a proportion of the overall DNA in a given cell is "expressed" (Refer to section 2.2.2 below for a review of gene expression) and the precise location and length of the sequence coding for a specific protein (genes) varies from cell to cell. Genes that are expressed in skin cells for instance may not all be expressed in other cell types although almost all cells contain the exact same DNA. First postulated in 1958 and subsequently published in 1970 (Crick, 1970) the central dogma of biology illustrates how information contained in DNA is used to synthesize polypeptides (primary structure of proteins) via the process of transcription and translation (Figure 2.1).

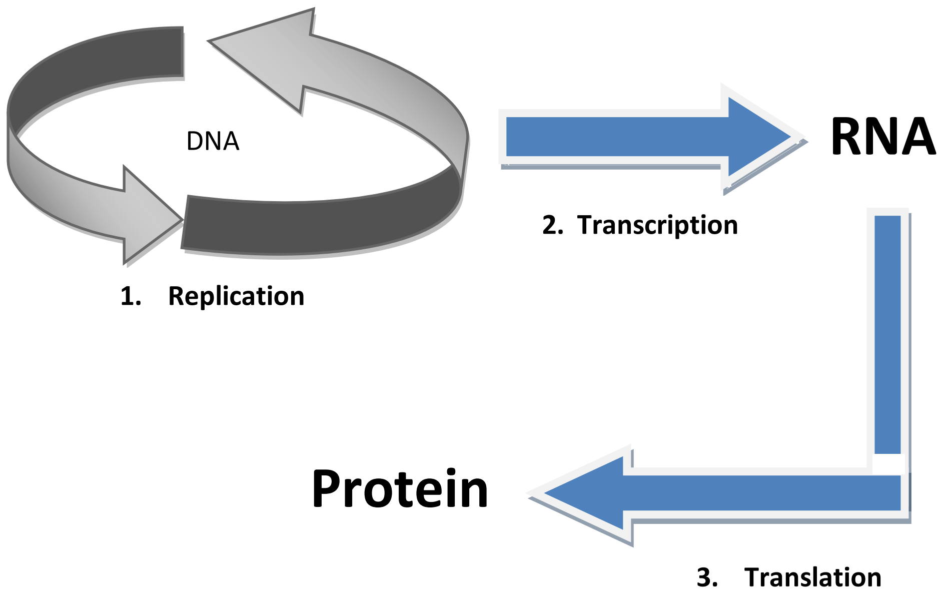


Figure 2.1 The central dogma of biology illustrating how information coded within DNA is transcribed into RNA via the process of transcription and subsequently translated into protein via translation. (Image constructed in Microsoft Word).

The central dogma of biology postulates that *"information cannot be transferred back from protein to either protein or nucleic acid*. For proteins to be synthesized, DNA acts as a template for messenger RNA (mRNA) synthesis via the process of transcription. This is necessary since DNA is located in the nucleus and is bound there. The newly synthesized mRNA then exits the nucleus and travels to the ribosomes located in the cytoplasm of the cell, where the information contained within mRNA is used to dictate the assembly of amino acids into polypeptides during the process of translation. The production of protein from DNA is therefore, an indirect process mediated by mRNA synthesis and the coding information cannot be transferred from protein to any other molecule (i.e. protein, RNA, or DNA), although the reverse holds true.

2.2.2 Gene expression

Gene expression is a term exclusively used to describe the process by which coding information within a gene leads to the production of a protein specific to that gene. Because a gene is a specific length of DNA coding for one or more polypeptides, it can be said that the gene is “expressed” when it leads to the synthesis of polypeptides that it codes for via the processes of transcription and translation. In other words, gene expression is the end result of translation and post-translational modifications. The relationship of genes and proteins is important in accurately understanding gene expression. Genes exist in the form of DNA sequences in the nucleus of the cell, which are programmed to synthesize proteins. Proteins on the other hand, are the product of genes and it is proteins that directly implicate cell behaviour by regulating various different cellular processes. Gene expression is hence, the production of one or more proteins as dictated by a given gene (Figure 2.2).

The fact that the human genome is being successfully sequenced has given birth to the goal of identifying all protein coding genes. However, in order to understand their functions in different physiological contexts, it is imperative to understand how their expression is regulated (Mintseris, 2006). In addition, gene expression is regulated by a number of different factors such as genes themselves, their products and even the products product (Stryer *et al.,* 2006). To aid in this, several high throughput techniques are available to efficiently facilitate the quantification of gene expression, most notable being Microarray technology (Schena *et al.,* 1996). This is discussed in more detail in section 2.3 of this thesis.

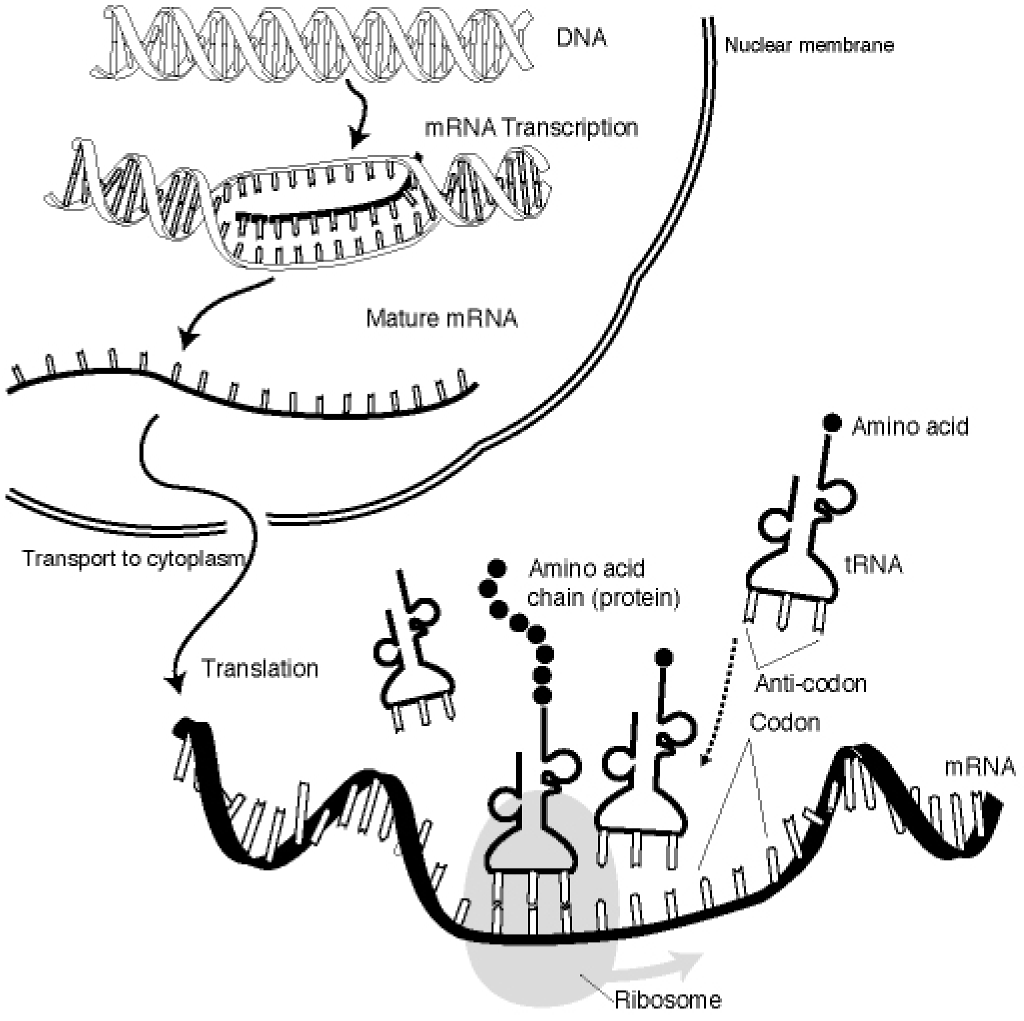


Figure 2.2 The complete process of gene expression involving transcription of DNA into mRNA, transport of mRNA into cytoplasm where it interacts with ribosomes to facilitate translation, and participation of tRNA (transfer RNA) in the synthesis of amino acid chains (proteins). Image taken from http://www.genome.gov

2.2.3 Transcription Factors

Transcription Factors (sometimes called sequence-specific DNA binding factors) are specialized proteins that bind to specific DNA sequences (called transcription factor binding sites) thereby regulating transcription (Latchman, 1997). These DNA-binding proteins can act alone or they can form a complex with other protein molecules and they control transcription by activating or suppressing the recruitment of RNA polymerase. Since RNA polymerase is an enzyme that controls transcription of DNA to mRNA in the nucleus of the cell, it is not surprising that transcription factors are able to impact gene expression to such a level. A prominent characteristic of transcription factors is that they contain one or more binding domains, which attach to their respective transcription factor binding sites adjacent to the genes that they regulate (Mitchell, 1989., Ptashne, 1997) and the binding occurs on either the promoter or enhancer sites of the genome. It has been estimated that there are approximately 2600 proteins that contain DNA binding domains in the human genome and a majority of these proteins are assumed to function as transcription factors (Babu, 2004).

The faculty of transcription factors to interact with DNA sequences in the genome to control transcription and ultimately the amount of gene expression has spawned a great deal of interest among researchers. This has motivated them to carefully examine promoter regions of genes to identify the involvement of certain transcription factors in the regulation of the genes (Kel, 2006). In the context of high throughput gene expression analysis such as microarrays, it is becoming common practice to employ this methodology to help answer complex biological questions since such technology allows biologists to efficiently quantify mRNA transcript levels for thousands of genes or even entire genomes concurrently. The application of promoter analysis to identify putative transcription factor binding sites in the perspective of whole genome analysis has hence, gained widespread popularity.

2.2.4 Protein-protein interactions

Once proteins have been synthesized by DNA via transcription and translation, they participate in regulating cellular behaviour in several different ways. An important trait of proteins is their ability to interact with other proteins to regulate many biological functions. For instance, protein-protein interactions play a fundamental role in signal transduction by mediating signals from the exterior of the cell to the inside of the cell. Sometimes a protein may form a complex with another protein in order to carry the protein it binds to from the cytoplasm to the nucleus and vice versa. Others may participate in protein-protein interactions to modify the structure of a protein, which may in turn alter the function of that protein. In any event, these interactive capabilities of proteins are vital to many biological processes that occur within the cell and it is crucial to gain a better understanding of them in order to enhance our comprehension of the underlying dynamics of disease. For example, signal transduction plays a major role in many diseases including Cancer and since proteins mediate signal transduction via protein-protein interactions, it becomes of utmost priority to study them in more detail.

There are a number of laboratory based methods commonly used to detect protein-protein interactions, such as Bimolecular Fluorescence Complementation (BiFC) (Lu *et al.*, 2008), Co-Immunoprecipitation and Fluorescence Resonance Energy Transfer (FRET) to name a few.

2.2.5 Molecular Pathways

In order to maintain the biological processes of life, cells must have a well orchestrated cascade of events that allow them to function properly. Molecular pathways form the cellular machinery that precisely regulates cell function by organizing proteins and inorganic molecules so that they can participate in various mechanisms central to the sustenance of the cell. Proteins together with their substrates are typically involved in highly complex interconnected chain of events and collectively, they constitute what is known as a molecular pathway. Because a cell hosts a vast number of biological processes and mechanisms, there are generally several active molecular pathways for a given cell. These pathways tend to work together to ensure that the organism continuously responds to changes in the internal and external stimuli.

However, sometimes the normal behaviour of pathways is affected as a result of mutations, leading to the production of oncogenes. For instance, the cell cycle pathway is vital to the normal functioning of a cell because it controls cell proliferation (Figure 2.3). On the other hand, cancer is a disease which results from uncontrolled cell proliferation and hence, the relationship between cell cycle and cancer is undeniable. The proliferative capability of cancer cells leads to an excess of cell number, which is a consequence of a reduction in sensitivity to signals that normally tell a cell to adhere, differentiate or die (Collins, 1997). Because cell proliferation is expected in a healthy cell, it becomes problematic to identify the exact causality of uncontrolled proliferation attributed to cancer. Similarly, the overproduction of a specific signalling protein called Epidermal Growth Factor receptor (EGF) has been implicated in many breast cancers because this overproduction has been known to cause uncontrolled cell division, leading to the development of tumours (Weinstein *et al.*, 1997).

From a systems biology point of view, the study of molecular pathways is vital if we are to improve our understanding of various different states of disease due to the fact that when a disease occurs, several key molecular pathways may be affected negatively. Gene expression studies from the use of microarrays has made it possible for us to correlate gene expression in a given disease phenotype to the affected area of several key pathways, hence providing us with the opportunity to decipher gene expression in a systems biology context. It is therefore, not surprising that a common challenge faced by all researchers is to translate gene expression data points into a better understanding of the underlying biological phenomena by putting this in the context of the whole organism as a complex system (Dragichi *et al.*, 2007). A study of biochemical pathways in particular is an important focal point in drug discovery and it is widely accepted as an important strategy by many biopharmaceutical and genomic companies (Dhillon *et al.*, 2007).

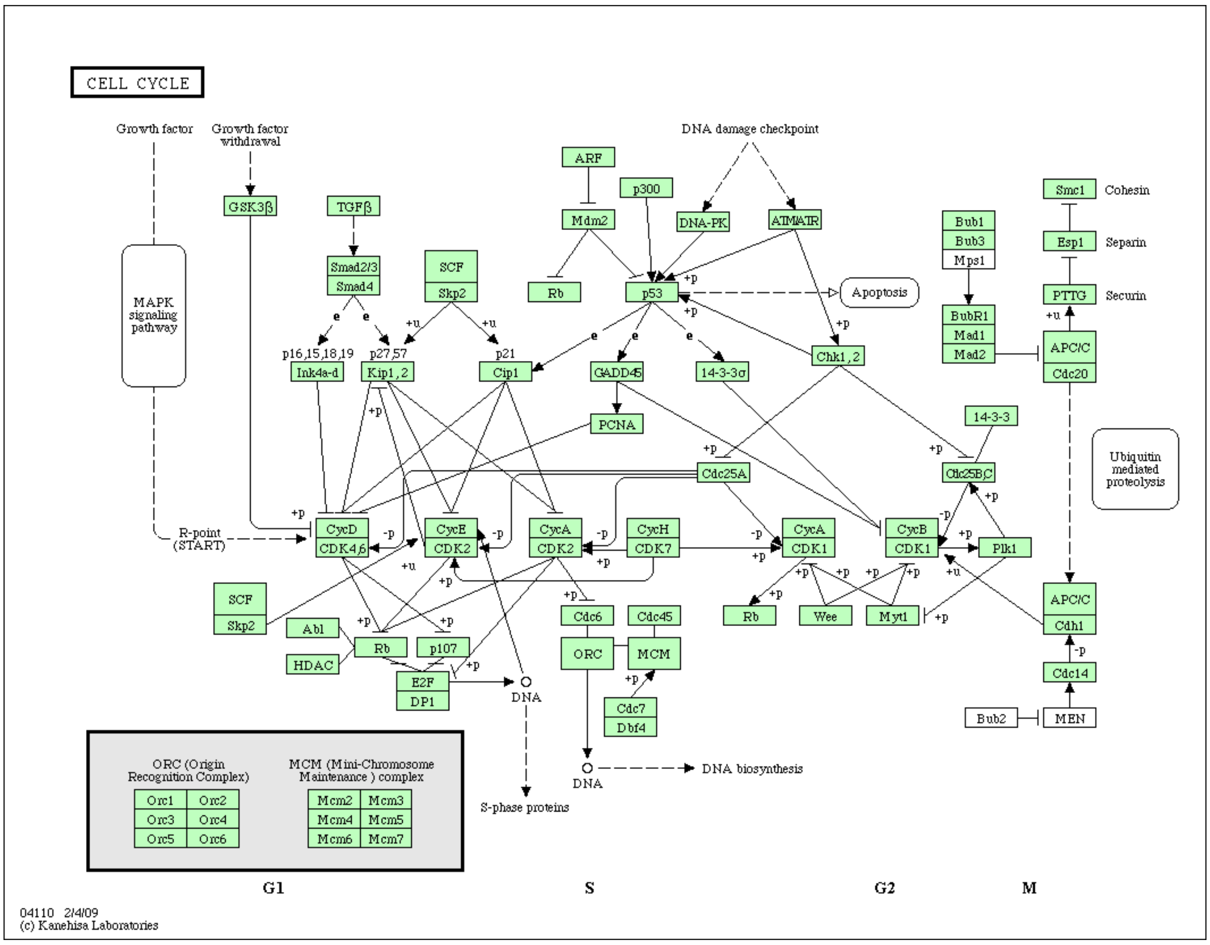


Figure 2.3 The cell cycle pathway illustrating how the complexity of molecular interactions and collaboration with other pathways results in controlled cell proliferation. Image taken from http://www.genome.jp

2.3 Microarray Technology

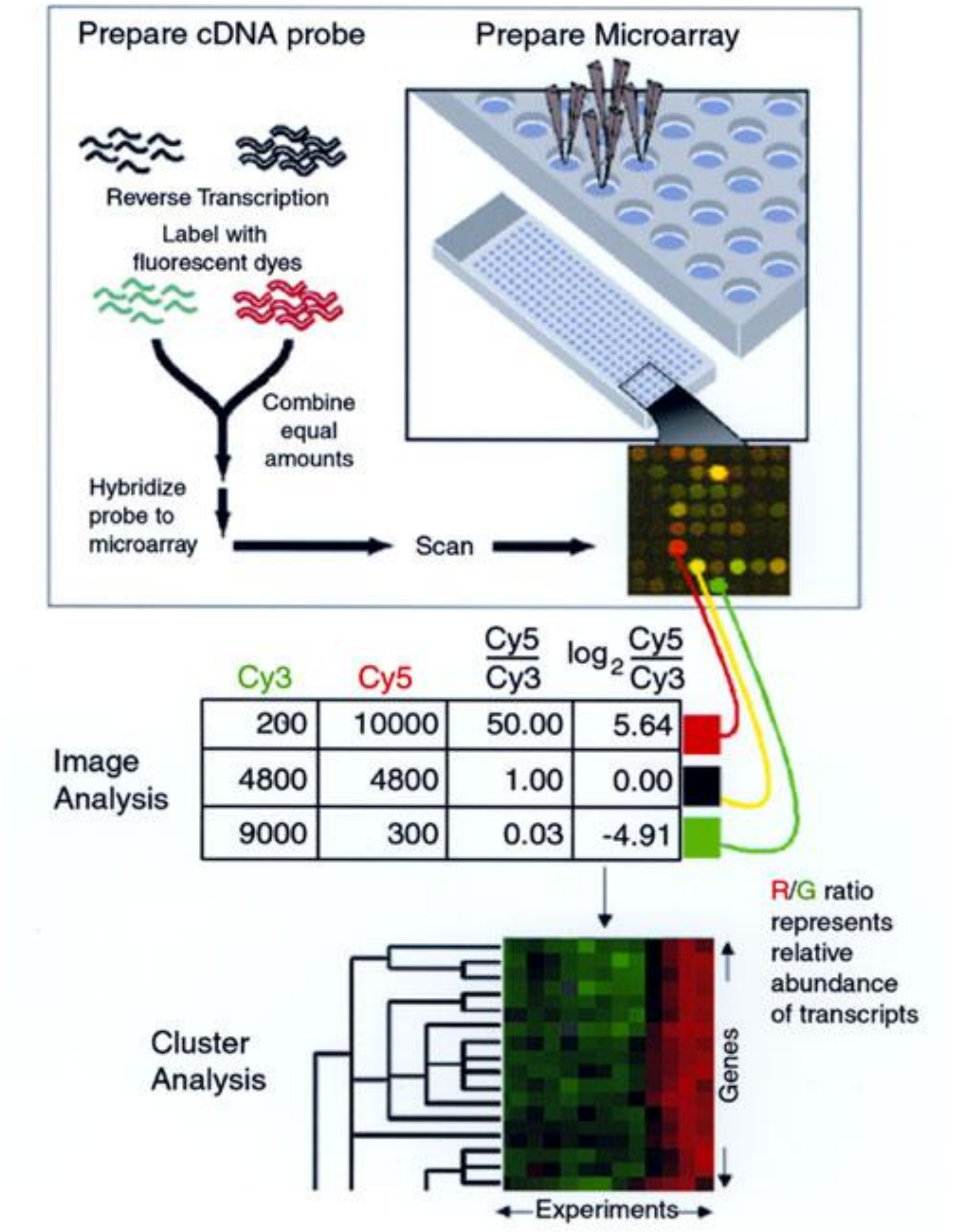
The history of microarray technology dates back to the 1980's, when it evolved from a well established laboratory technique called southing blotting (Southern *et al.*, 1999) used to attach a fragmented DNA to a substrate followed by probing it with a known target gene. It was not until 1987 when a collection of DNA sequences were used in arrays for the purpose of expression profiling (Kulesh *et al.*, 1987).

Microarray technology has revolutionized the field of molecular biology by offering an efficient and cost effective platform for the simultaneous quantification of thousands of genes or even entire genomes in a single experiment. Unlike southern blotting, which is restricted to the measurement of one gene at-a-time, microarrays offer biologists with the opportunity to carry out genome-wide experiments in order to help them gain a systems level understanding of cell regulation and control. Central to the technology, microarrays are a major breakthrough due to the use of a DNA chip (made of glass, plastic or silicon biochip) on which specific sequences of DNA (or entire genomes) are robotically printed using a microarrayer. Labelled DNA or RNA samples are then applied to the chip in order to detect complementary sequences. A typical gene expression profiling experiment is motivated by a specific biological question asked by the investigator. Expression levels of thousands of genes are then simultaneously monitored in order to study the effects of certain biological conditions, disease types and treatments on gene expression (Adomas *et al*., 2008).

2.3.1 The Microarray Procedure

The high throughput nature of microarray technology makes it a fundamental asset to the field of molecular biology because it circumvents the restrictions set by one-gene-by-one-experiments (i.e. Southern blotting) as before (Schena *et al.*, 1995; 1996). With the advent of this technological breakthrough, several thousands of genes can be scrutinized concurrently in a single experiment (Spellman *et al.*, 1998), making it possible to observe a global snapshot of the entire cell in a specific time point and/or following a particular treatment.

DNA microarrays are constructed by attaching fragments of DNA (each representing a specific gene at its known position) such as library clones or PCR (Polymerase Chain Reaction) products to a solid substrate (Schena *et al.*, 1995). A robotic microarrayer is used to print the fragments on to the substrate, which can typically spot more than 20,000 fragments per substrate. RNA is then extracted from the untreated cell sample (control) and treated sample (test), which are then each reverse transcribed into cDNA. During reverse transcription, as each cDNA sample is being synthesized from the RNA template by the enzyme Reverse Transcriptase (RT), they are each labelled with a specific fluorescent dye (Cy3 and Cy5 for the control and test samples respectively), which becomes incorporated into the newly synthesized cDNA. Following the synthesis of labelled cDNA, equal amounts of each cDNA sample (test and control) are combined and subsequently hybridized on to the microarray chip (containing DNA fragments of known genes). By labelling both samples with different fluorescent dyes (Cy3 and Cy5), relative abundance of mRNA transcript levels can be measured by determining the fluorescence ratio for each spot on the substrate when scanned. Figure 2.4 illustrates the complete microarray process.



Cy3	Cy5	$\frac{Cy5}{Cy3}$	$\log_2\frac{Cy5}{Cy3}$	
200	10000	50.00	5.64	🟥
4800	4800	1.00	0.00	⬛
9000	300	0.03	-4.91	🟩

Figure 2.4 DNA microarrays. DNA fragments are spotted on to a glass slide (top right). RNA is then extracted from the two samples to be compared and then fluorescently labelled cDNA is prepared by reverse transcription, leading to the incorporation of Cy3 dye (green) in the control cDNA sample and Cy5 dye (red) in the test cDNA sample (top left). Both labelled cDNA samples are then mixed and hybridized on to the microarray and the slide is scanned to generate the image. In the image, green spots reflect down-regulated (under expressed) genes, red represent up-regulated (over expressed) genes and yellow show an equal expression of both red and green. Special image analysis software is used to determine signal intensities for each dye at each array position and logarithm of the ratio of

Cy5 intensity to Cy3 is calculated to determine the strength of expression. Once intensity ratios are calculated for each array position, positive Cy5/Cy3 ratios indicate over expressed genes and negative Cy5/Cy3 ratios indicate under expressed genes. The datasets can then be analyzed by cluster analysis (bottom). (Image taken from Cummings & Relman, 2000).

2.3.2 Pre-processing microarray data & differential expression analysis

Underlying every microarray experiment is a specific biological question being addressed and in order to effectively answer the biological question, intensity ratios of all genes on the microarray chip must be carefully examined to accurately identify those genes that exhibit a significant difference of expression between the test and control samples. By narrowing down the genes of interest, the focus of data interpretation becomes concentrated on these potentially meaningful genes, hence providing the biologist with clues as to where the answer to the biological question may lie. Furthermore, because microarray experiments typically generate colossal amounts of raw biological data, inconsistencies may arise in the gene expression measurements due to sources of non-experimental variations. Preprocessing microarray data, including image analysis, normalization and data transformation is an active research area and how to suitably quantify spots on microarrays remains a topic of great enquiry (Allison *et al*., 2006).

Because gene expression measurements are derived from the microarray image itself (when the microarray chip is scanned and subsequently generated on screen by specialized analysis software tools), a great deal of attention is dedicated to effectively process microarray images and many image processing approaches have been developed in the past to cater for this need (Chen *et al*., 1997; Schadt, *et al*., 2001; Ekstrom *et al*., 2004; Yang *et al*., 2001; Steinfath *et al*., 2001). An important preprocessing step is normalization, which allows comparisons to be made between microarray experiments by controlling extraneous variation among experimental datasets (Allison *et al.*, 2006). Normalization is carried out by applying statistical methods to the entire experimental datasets (Lee *et al.*, 2000) so that gene expression comparisons can be made in a manner that minimizes systematic bias. Gene expression measurements are derived from measuring the distribution of pixels in each spot on the microarray image and the intensity of pixilation of each spot can be used to assess the degree of similarity of the distribution in relation to the normal distribution. Researchers can also manually flag groups of spots or individual spots as good, bad or absent in order to help them differentiate between valid and invalid measurements of expression. Other statistical methods such as Standard Deviation (SD) can also be used to observe the spread of the entire data. The purpose of these statistics is to act as an aid for researchers by allowing them to estimate which spots are statistically usable so that they can be incorporated into the final dataset (Dudoit *et al.*, 2002).

Pre-processing and normalizing microarray data with the purpose of minimizing the effects of systematic error while retaining full biological variation are critically important goals that need to be given consideration if valid results are to be obtained from the experiment (Zahurak *et al.*, 2007). During the normalization process, the dual dye measurements of each spot are used as a basis to estimate the relative abundance of expression for each gene by comparing the colour intensity of the test dye (Cy5) relative to the control (Cy3) at each spot on the array. The measurements are Log transformed (Logarithm to the base of 2) and from the resulting values over and under expressed genes can be readily identified, since positive values represent genes that are over expressed in the first condition compared to the second and negative values represent genes that are under expressed in the first condition compared to the second (Dudoit *et al.*, 2002). Fold change is one of the most widely used approaches employed for the purpose of comparing the expression levels of genes across different biological conditions. Consider two biological conditions, X and Y, where each expression value of each gene has been log transformed. Fold change is calculated simply by subtracting each gene expression value of the biological condition exhibiting a lower expression measurement from the condition exhibiting a higher gene expression value. In other words, *if $X > Y$, then $X - Y$, else $Y - X$.* The result of this is that positive fold change values will indicate genes that are over expressed in biological condition X in comparison to Y, while negative values will indicate under expressed genes in condition X compared to Y. A fold change threshold is used to define the extent of biological variation that is considered to be significant by the researcher performing the experiment, which generates a narrowed down set of genes considered to be highly significant in answering the biological question. These genes of interest are referred to as differentially expressed genes.

The fold change approach was the first method used to assess whether genes are differentially expressed by adequately measuring effect size and its popularity predominantly stems from its simplicity (Allison *et al.*, 2006). However, fold change calculations should not be used as a sole basis to measure biological variation primarily because it is widely considered to be an inadequate test statistic (Hsiao *et al.*, 2004; Miller *et al.*, 2001). This is because it does not offer an associated level of confidence and hence, it fails to incorporate variance (Hsiao *et al.*, 2004; Budhraja *et al.*, 2003). This is not to say that the fold change approach should not be used because it is conceptually useful due to the fact that it assumes a constant variance across transcripts (Allison *et al.*, 2006). Nevertheless, if valid biological inferences are to be made, its use should be coupled with other test statistics such as the T test statistic.

2.3.3 Clustering gene expression data

High throughput experiments such as microarrays no doubt generate vast amounts of biological data, which is not surprising considering the fact that they are designed to simultaneously quantify thousands of genes. In other words, the voluminous nature of the output is directly proportional to the volume of genes on the microarray chip (the input). For this reason, making valid biological inferences from the generated data can be a daunting task for the common biologist. An important objective of a microarray experiment is to identify groups of genes that exhibit similar behaviour of expression, since this is indicative that they may share common regulatory signatures and may ultimately be involved in the same biological processes (Barrett *et al*., 2006). This objective is fundamentally important because it has the potential to help answer the specific biological question motivating the microarray experiment. One of the most common methods of analyzing expression data in this context is clustering (Hand & Heard, 2005).

Clustering is a very effective technique to extract interesting patterns of gene expression across different biological conditions, whilst circumventing the problematic nature of expression data attributable to the complexity of biological networks that lie hidden within the data. The simplest explanation of clustering is based on the paradigm of grouping subsets of genes (into clusters) that show similar expression measurements so that interesting biological patterns can be easily visualized. However, in order to take maximum benefit from the use of clustering, care must be taken to ensure that trivial subsets of genes are not included in the clustering analysis while employing the more interesting ones. When performing any sort of clustering, the distance metric is used to define the measure of similarity, which is a function that uses two points in a dimensional space where symmetry lies (Shay, 2003). There is not a single clustering solution universally applicable to all biological problems and several different clustering algorithms have been developed, each suitable for specific needs (Everitt, 1980). Some examples include Hierarchical clustering (Eisen *et al.*, 1998; Heyer *et al.*, 1999; Qin *et al.*, 2003), Self Organizing Maps (SOMS) (Ressom *et al.*, 2003; Kohonen, 1995), K-means clustering (Tavazoie *et al.*, 1999), and Principal Component Analysis (PCA). Since Hierarchical clustering and K-means clustering are most widely applicable to the analysis of gene expression data, they ought to be discussed here in some detail.

2.3.3.1 Hierarchical Clustering

Hierarchical clustering (Eisen *et al.*, 1998) is usually the first clustering method of choice when analyzing gene expression data. The method itself is based on a single layered neural network and when performed, it hierarchically groups genes located near each other and with similar expression patterns together across a series of samples. The algorithm computes distance relationships between genes and experiments in a pair-wise manner and joins pairs of expression values that are most similar in expression to form what is called, a node. Upon each iteration, clusters of genes with similar expression patterns are then merged together into a single cluster until the desired number of clusters is obtained (Figure 2.5). The hierarchical arrangement of clusters takes the form of a dendrogram, which is a classical tree structure that appears when the clusters are graphically represented (Hand & Heard, 2005). The dendrogram illustrates the relationship between the clusters, making it easy to visualize different segments of similarly expressed genes.

Although Hierarchical clustering is one of the most common clustering approaches used widely for the analysis of gene expression data, there are some limitations of its use. Firstly, it fails to yield meaningful results as the number and size of datasets to compare grow. It is also notorious for not being very robust.

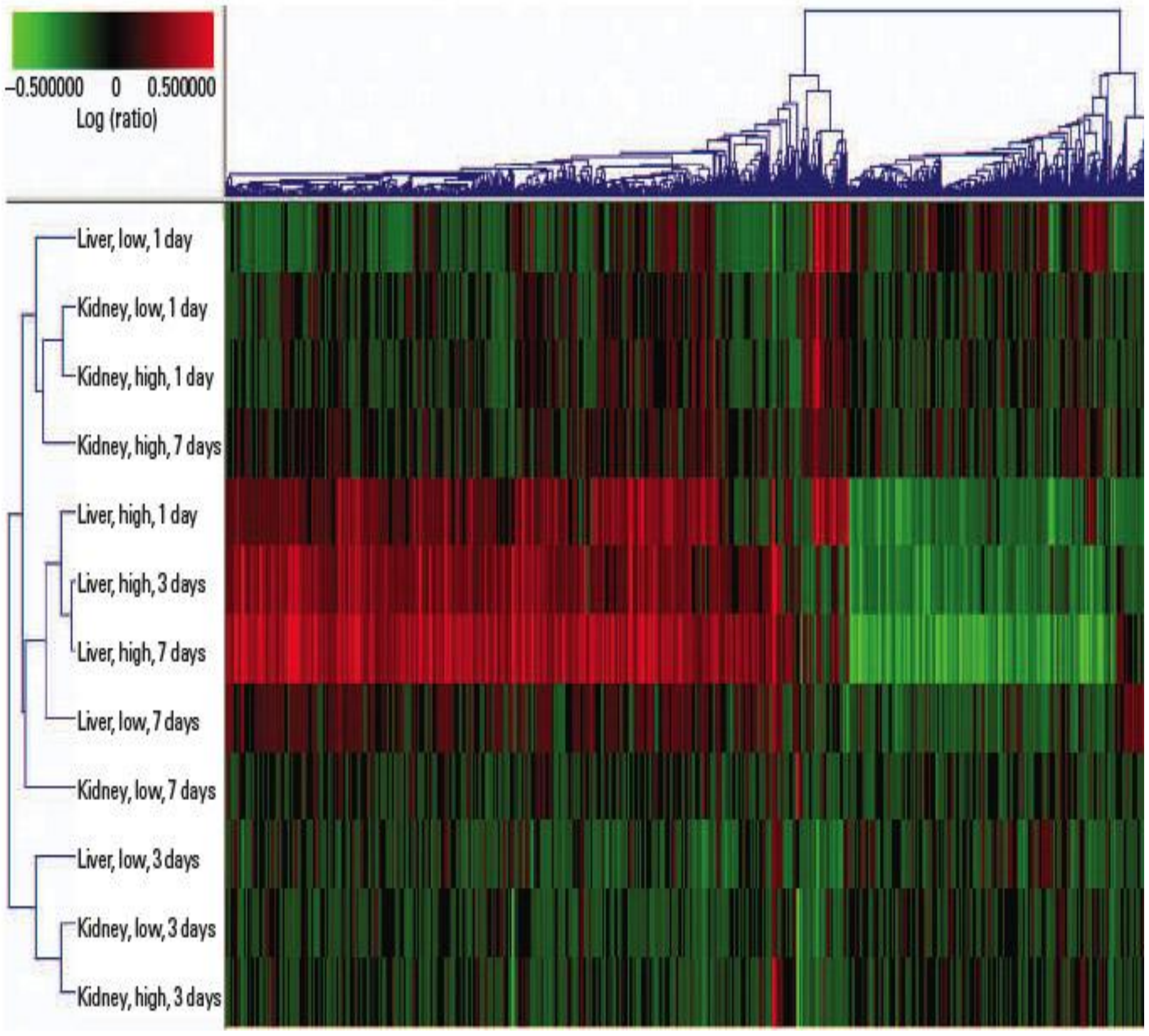


Figure 2.5 Hierarchical clustering. The dendrogram on the top represents the relationships between clusters of genes and the dendrogram on the left illustrates the relationships between the experimental conditions. Clusters of similarly expressed genes are grouped together, where red represents over-expressed genes, green represents under-expressed genes and black represents no change in expression. Image taken from Auman *et al*., 2007.

2.3.3.2 K-means Clustering

K-means (Tavazoie *et al.*, 1999) is a straight forward and efficient clustering method that arbitrarily divides the entire dataset into k number of disjoint subsets (Figure 2.6). The criterion used to define the number of clusters to divide the data into is user-defined and so as soon as the user inputs the number of clusters that they intend the data to be divided into (k), k-means efficiently segregates the dataset into k number of partitions, the centre of each partition or cluster being called a *centroid*. The k-means algorithm groups data in an optimal manner by calculating the distance between each gene and the centroid of each cluster. This ensures that each gene is assigned to the centroid (and subsequently grouped into that cluster) with the closest Euclidean distance whilst ensuring maximal distance between genes belonging to different clusters.

The main drawback of k-means clustering is its arbitrary nature of clustering data, which generates different results each time it is performed. This is because the algorithm assumes no knowledge of the number of clusters in a given gene expression dataset, resulting in an alteration of results after each successive run.

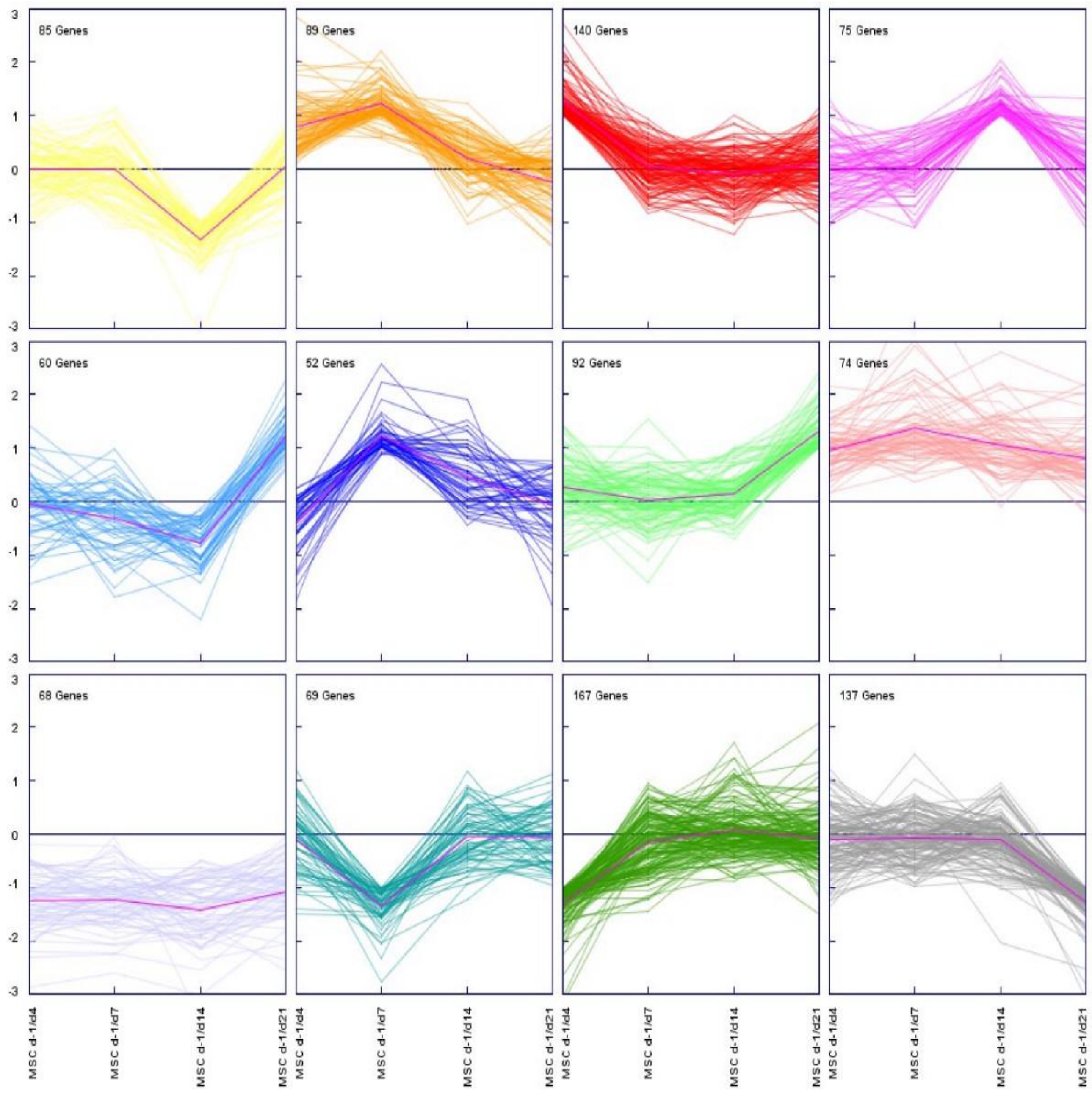


Figure 2.6 A graphical representation of results generated from a typical K-means clustering analysis. In this example, the entire expression dataset is divided into 12 clusters, each consisting of genes grouped together showing their expression levels across different time points. The magenta line represents the mean expression level of each cluster. Image taken from Kulterer *et al*., 2007.

2.4 Analyzing microarray data in a systems biology context

With the advent of bioinformatics, a whole new generation of opportunities to decipher complex biological data have come into existence. This is especially true in the case of microarrays. Since the use of this valuable high throughput technology compels biologists to be faced with vast magnitudes of raw complex biological data, bioinformatics has opened up a gateway for exciting opportunities in the context of systems biology. We are no longer living in an era where the field of biological sciences is self sufficient. Testament to this is the fact that the field of biology has become heavily reliant on informatics, mathematics, computer algorithms, software development and statistics in order to solve biological problems. In the midst of this interdisciplinary matrix, a realization has been born from which, the field of systems biology has emerged. It is becoming abundantly apparent that a scrutiny of characteristics of isolated parts of the cell or organism is not sufficient to warrant a global understanding of how a given organism functions as a whole unit (Kitano, 2002). The motivation of systems biology is hence, to understand physiology and disease from the integrative perspective of molecular pathways, regulatory networks, cells, tissues, organs, and ultimately the whole organism (Butcher *et al.*, 2004).

The marriage of systems biology with microarrays is undeniable. On one hand we have a high throughput technology offering a platform to measure mRNA transcript levels for entire genomes simultaneously in a cost effective and efficient fashion, and on the other hand, we have a interdisciplinary science specializing in the application of mathematics, computer science and biology with the fundamental purpose of unravelling the underlying functional dynamics of the cell in the perspective of the entire organism. In theory, the possibilities are endless. In practice, systems biology is still at its infancy and consequently, there is much to be desired.

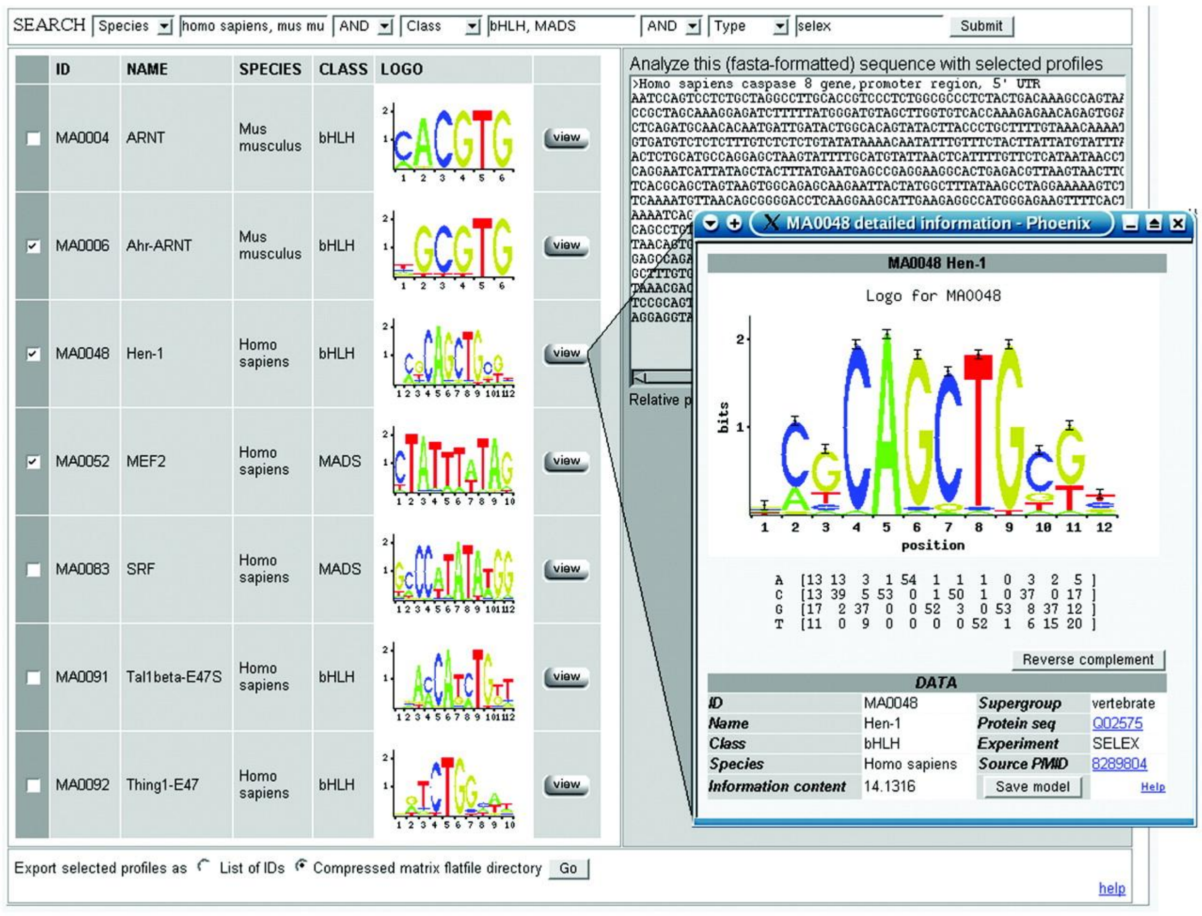
Nevertheless, research in the area of systems biology is intense. There are generally three main areas that are progressing steadily. The first is the analysis of high throughput data such as microarrays using statistical algorithms and compiling reference databases encompassing information pertaining to genes, proteins, enzymes, and entire genomes (Moore, 2007). The second is the development of specialized bioinformatics software pipelines to automate gene expression analysis in the context of systems biology, and the third is the implementation of computational approaches to help answer biological questions. For instance, artificial intelligence in the form of neural networks is constantly being applied for pattern recognition purposes (Statnikov *et al.*, 2005). Although the intensity of such efforts is undeniable, the truth remains that biologists are generally reluctant to employ bioinformatics applications/tools, especially if they are not able to comprehend how the analysis is carried out (Kulyk & Wassink, 2006). This is acting as a rate limiting factor and with time it is becoming more apparent that in order for novel biological discoveries to be made, true collaboration must exist between biologists and bioinformaticians. It is also becoming apparent that bioinformatics software programs need to be developed in a way in which biologists can use them without having problems understanding how the underlying analysis is carried out. This is an important step required to reinforce this necessary collaboration.

The current predicament is such that there is a variety of biological data scattered everywhere. Several databases are available, each storing crucial biological knowledge. Furthermore, much work is being conducted in the field of systems biology by developing software applications to facilitate high throughput analysis. The aforementioned situation shall be considered in the sections that follow.

2.4.1 Transcription Factor databases

The faculty of transcription factors to interact with DNA sequences in the genome to control transcription and ultimately the amount of gene expression has spawned a great deal of interest among researchers. This has motivated them to carefully examine promoter regions of genes to identify the involvement of certain transcription factors in the regulation of the genes (Kel, 2006). Because these regulatory DNA sequences bind to transcription factors in order to control transcription and ultimately gene expression, a great deal of effort is being placed on understanding these sequences. Attention is now shifting towards representing these sequences (called transcription factor binding motifs or TFBMs) in the form of Position Frequency Matrices (PFMs) and Position Weight Matrices (PWMs). A detailed explanation of these terminologies is explained in the subsequent chapter of this thesis.

Currently, there are publicly available transcription factor databases, such as JASPAR (Sandelin, *et al.*, 2004) and TRANSFAC (Matys *et al.*, 2003). JASPAR catalogues matrix-based transcription factor binding profiles for the purpose of warehousing non redundant representations of high quality transcription factor binding site (TFBS) profiles (Sandelin, *et al.*, 2004) (Figure 2.7). The profiles are derived as a result of collecting experimentally verified TFBMs (for multicellular eukaryotes) from published data, where some of the binding sites were determined by SELEX experiments (Pollock *et* al., 1990). Such databases have given rise to opportunities to analyze these crucial regulatory sequences against promoter regions of genes in a high throughput context.



	ID	NAME	SPECIES	CLASS	LOGO
☐	MA0004	ARNT	Mus musculus	bHLH	
✓	MA0006	Ahr-ARNT	Mus musculus	bHLH	
✓	MA0048	Hen-1	Homo sapiens	bHLH	
✓	MA0052	MEF2	Homo sapiens	MADS	
☐	MA0083	SRF	Homo sapiens	MADS	
☐	MA0091	Tal1beta-E47S	Homo sapiens	bHLH	
☐	MA0092	Thing1-E47	Homo sapiens	bHLH	

Analyze this (fasta-formatted) sequence with selected profiles

```
>Homo sapiens caspase 8 gene,promoter region, 5' UTR
AATCCAGTCCTCTGCTAGGCCTTGCACCGTCCCTCTGGCGCCCTCTACTGACAAAGCCAGTA
CCGCTAGCAAAGGAGATCTTTTTATGGGATGTAGCTTGGTGTCACCAAAGAGAACAGAGTGG
CTCAGATGCAACACAATGATTGATACTGGCACAGTATACTTACCCTGCTTTTGTAAACAAAA
GTGATGTCTCTCTTTGTCTCTCTGTATATAAAACAATATTTGTTTCTACTTATTATGTATTT
ACTCTGCATGCCAGGAGCTAAGTATTTTGCATGTATTAACTCATTTTGTTCTCATAATAACC
CAGGAATCATTATAGCTACTTTATGAATGAGCCGAGGAAGGCACTGAGACGTTAAGTAACTT
TCACGCAGCTAGTAAGTGGCAGAGCAAGAATTACTATGGCTTTATAAGCCTAGGAAAAAGTC
TCAAAATGTTAACAGCGGGGACCTCAAGGAAGCATTGAAGAGGCCATGGGAGAAGTTTTCAC
AAAATCAG
CAGCCTG
TAACAGTG
GAGCCAGA
GCTTTGTG
TAAACGAG
TCCGCAGT
AGGAGGTA
```

MA0048 Hen-1

```
A  [13 13  3  1 54  1  1  1  0  3  2  5 ]
C  [13 39  5 53  0  1 50  1  0 37  0 17 ]
G  [17  2 37  0  0 52  3  0 53  8 37 12 ]
T  [11  0  9  0  0  0  0 52  1  6 15 20 ]
```

DATA

ID	MA0048	Supergroup	vertebrate
Name	Hen-1	Protein seq	Q02575
Class	bHLH	Experiment	SELEX
Species	Homo sapiens	Source PMID	8289804
Information content	14.1316		

Figure 2.7 The User Interface of the JASPAR database, showing the results from a typical query. Taken from Sandelin, *et al.*, 2004

2.4.2 Molecular pathway databases

There are a number of publicly available databases that store information relevant to biological pathways and genes. The most prominent pathway database is the KEGG database (Kanehisa & Ogata *et al.*, 1999) since it contains a comprehensive collection of known biochemical pathways, providing information pertinent to genes and their relationships in the pathway networks (Figure 2.8). Furthermore, each pathway is represented as a seemingly static pathway map where each member or genes in a given pathway are hyperlinked to the underlying comprehensive gene database. Biocarta (http://www.biocarta.com) is another online database, containing a collection of biological pathways. MetaCyc (Karp *et al*., 2002) is a more specialized and specific pathway database, which primarily focuses on the dissemination of metabolic pathways. The advantage of MetaCyc is that unlike KEGG, which does not allow users to carry out pathway engineering due to a lack of interactive functions, MetaCyc facilitates the use of metabolic engineering by offering the capability to interact with the pathways in order to add, remove or replace genes. Both KEGG and MetaCyc have an active Application Programming Interface (API), giving rise to the opportunity to dynamically interact with them by writing code for specific needs.

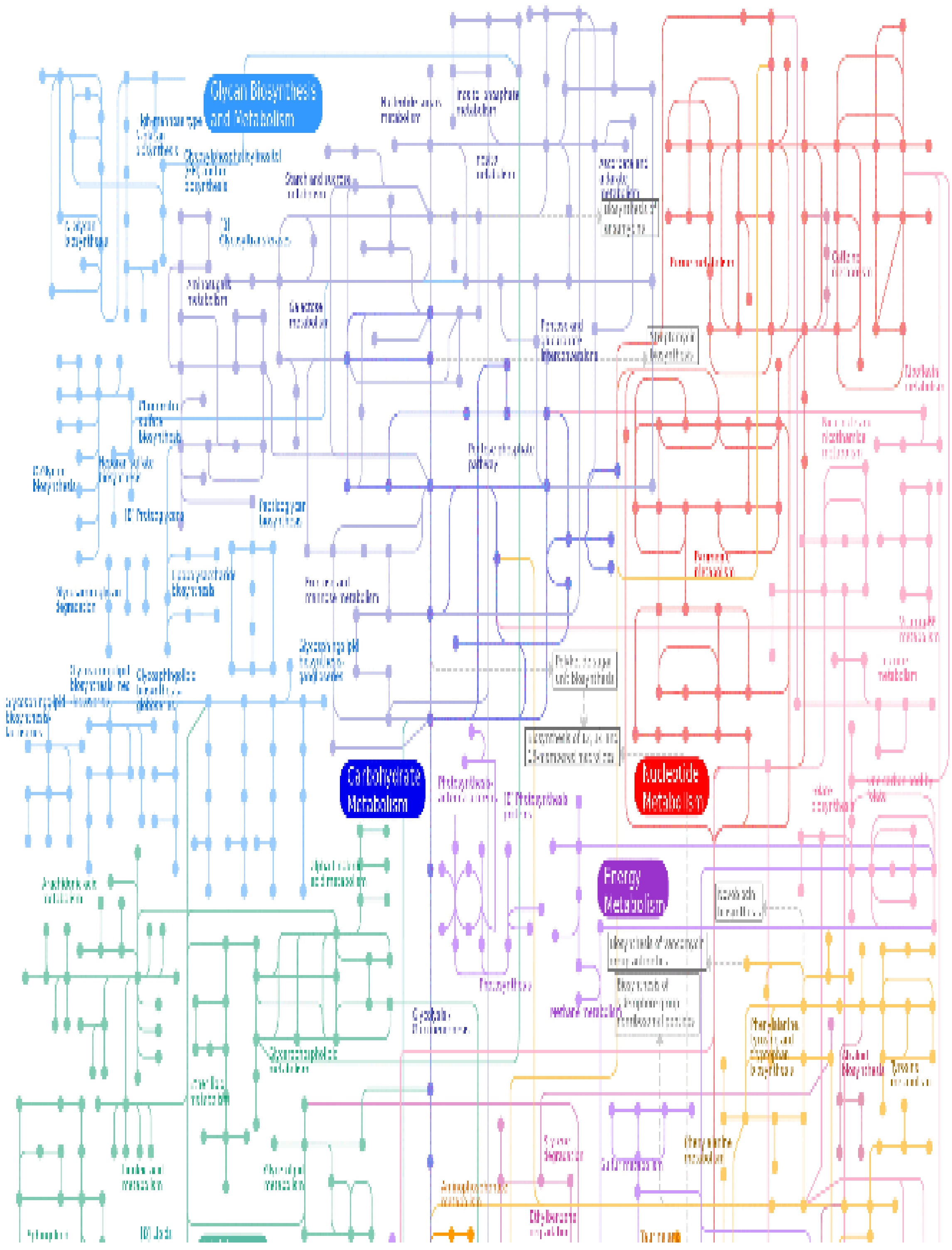


Figure 2.8 The Metabolic Atlas of KEGG, displaying proteins in the form of nodes, and associations as lines. Each coloured box represents a separate metabolic pathway and the atlas shows how pathways and proteins are interconnected. Taken from http://www.genome.jp/kegg/atlas

2.4.3 Available software

In an attempt to decipher gene expression data in a systems biology context, several bioinformatics software tools have been developed. Some examples of such software include Expander (Shamir *et al.*, 2005), INCLUSive (Thijs *et al.*, 2002), Genesis (Sturn *et al.*, 2002), CONFAC (Karanam *et al.*, 2004) and GEPAS (Herrero *et al.*, 2003). Most of these software focus on streamlining the analysis flow of microarray data analysis by implementing specific functions such as k-means clustering, bi-clustering, promoter analysis to identify transcription factor binding sites and functional analysis. In addition, there are various software applications that attempt to extract a systems level understanding by facilitating data analysis from the perspective of molecular pathways. Pathfinder (Goesmann *et al.*, 2002) is one of such tools, which facilitates pathway engineering from annotated data. GenMAPP (Salomonis *et al.*, 2007) is another software relevant to pathway analysis, which is more specific for the analysis of microarray data. From the perspective of transcription factor analysis, several programs such as MATCH (Kel *et al.*, 2003) and P-MATCH (Chekmenev *et al.*, 2005) from TRANSFAC (Matys *et al.*, 2006), PRODORIC (Muench *et al.*, 2003), and PoSSuMsearch (Beckstette *et al.*, 2006) are available that can scan upstream sequences of putative target genes to identify TFBMs. All of these software and others alike are no doubt important in the milieu of systems biology. However, there are several loopholes that are yet to be accommodated, some of which are addressed in the subsequent chapters of this thesis.

2.5 Conclusion

The purpose of this chapter was to provide the reader with a systematic walkthrough of the main principles that collectively form the backbone of systems biology. Because the relationship between systems biology and microarray technology is a fundamentally important one, an overview of microarray technology was given from the microarray paradigm itself to the need for clustering solutions to solve biological problems. Finally, current works in the area of systems biology was summarized in order to give an idea of how much progress has been made in the field and the array of opportunities available for the future. The subsequent chapters of this thesis will now present contributions made during the course of pursuing this PhD.

Chapter 3

3

BiSAn: A software for efficient computation of transcription factor binding motifs for high throughput gene expression analysis

3.1 Introduction

DNA-binding transcription factors (TF's) are important determinants of transcriptional regulation and control that bind to specific recognition sites of operator sequences of target genes to activate or suppress transcription. Environmental factors coupled with the internal conditions of the organism determine the proportion of the complete set of transcription factors that are active at a given point in time, making it possible to observe specific states of the transcriptome. Although each cell of an organism contains an exact copy of its genome, the expression of genes can vary due to different biological conditions, giving rise to different transcriptome states. Furthermore, some transcription factors may only dictate the expression of a single gene, whereas others may organize the activation or suppression of several genes at once (Teichmann *et al.*, 2004). It is this precise molecular ability of TFs to interact with specific recognition sites called transcription factor binding motifs (TFBMs) that makes it possible for cells to control reproduction, growth and death. It is hence no surprise that a lot of attention has shifted towards better understanding these powerful DNA-binding proteins together with their target TFBMs. These binding motifs are usually represented as matrices and are referred to as position weight matrices (PWMs), position frequency matrices (PFMs) or alignment matrices in the scientific literature.

Position frequency matrices (PFMs) represent the most widely employed model for TFBMs and have been catalogued in web-based databases such as JASPAR (Sandelin, *et al.*, 2004), where each matrix consists of nucleotide counts per position for a given TF (Stormo *et al.*, 1982). Given a set of TFBMs, regulatory networks can be predicted through *in silico* methods by using cognate binding sequences to construct models. Using a given PFM pertinent to a given collection of TFBMs, consensus binding sites specific to a particular TF can be readily derived that represent evolutionarily conserved regions. Scanning such putative binding sites in promoter regions of genes has been recognised as an important step in determining the potential binding affinity of a transcription factor to its respective binding site(s) and several programs such as MATCH (Kel *et al.*, 2003) and P-MATCH (Chekmenev *et al.*, 2005) from TRANSFAC (Matys *et al.*, 2006), PRODORIC (Muench *et al.*, 2003), and PoSSuMsearch (Beckstette *et al.*, 2006) are available that can scan upstream sequences of putative target genes to identify TFBMs. However, careful scrutiny of current literature reveals a growing demand for automating this search strategy for the analysis of high throughput expression data such as microarrays. Although the aforementioned programs can be used to automate high throughput search, it is apparent that there is a lack of available pipelines specifically designed for the common biologist who may wish to put a set of differentially expressed genes into an automated pipeline in order to scan promoter sequences pertinent to these genes for the presence of TFBMs (using PFMs) at the click of a button.

This chapter introduces BiSAn, a software pipeline specifically designed for microarray data analysis that uses PFMs to scan promoter sites for the presence of TFBMs enriched in the user's set of data. Regardless of the biological question, a microarray experiment usually results in one or more sets of differentially expressed genes, which represent genes that have significant difference in expression ratios between the control and test samples. An important challenge is to decipher these expression data points in the context of transcriptional regulation to better understand the biological phenomenon under investigation. Once a set of differentially expressed genes has been imported by the biologist, BiSAn automatically fetches their promoter sequences and subsequently scans them for TFBMs in an efficient and high throughput fashion. It then carries out TFBM enrichment analysis to detect binding sites that are enriched within the user's set of genes.

The subsequent sections of this chapter will focus on the following. Firstly, the concept of Position Frequency Matrices and the required terminology is explained. Next in the Methods, the implementation of the algorithm and software is described for high throughput TFBM detection. The efficiency and biological usefulness of BiSAn is then described by subjecting it to microarray data generated from our in-house studies.

3.2 Terminology

Position frequency matrices (PFMs) represent the most widely employed model for TFBMs where each PFM, F = ($f\sigma j$) consists of a set of TFBMs of length m over an alphabet Σ = {A,T,C,G}. A PFM is hence defined as a $|\Sigma| \times m$ matrix, where $f\sigma j$ is the frequency of symbol σ at position j

The information content, Ij (1) and mean information content, I(F) (2) for column j of a position frequency matrix, F can be defined as follows:

$$Ij = \log 2\, |\Sigma| + \sum_{\sigma \in \Sigma} f\sigma j \cdot \log 2\, f\sigma j \text{ [bits]}. \qquad [3.1]$$

$$I(F) = 1/m \sum_{j=1}^{m} Ij\,. \qquad [3.2]$$

When measuring the quality of a given PFM, the mean information content is used.

3.3 High Throughput scanning of PFMs for promoter analysis

For a given set of differentially expressed genes, PFMs for *x* number of TFs can be used to scan promoter sequences pertaining to these interesting genes. Because each PFM, F = (*fσj*), the frequency of symbol *σ* for each nucleotide base A, T, G or C at position *j* over the length of the entire matrix can be used to determine all possible consensus binding sites that have the potential to bind to the transcription factors which they belong to. Each consensus can then be given a score based on 1) the frequency of each base A, T, G or C at position *j* of the matrix and 2) the total frequency of all bases at position *j* of the matrix. The subsequent scores of each consensus site can then be used to reflect the putative binding affinities of these sites to their respective TF, which can facilitate *in silico* biological predictions.

The purpose of BiSAn is to provide a simple yet effective medium for biologists to efficiently identify putative TFBMs from promoter sites belonging to their genes of interest at the click of a button.

3.4 Methods

This section describes how the TF data was collected and organized followed by a description of the algorithm used to scan promoter sites for the presence of consensus TFBMs.

3.4.1 Data collection and storage

The following sets of data were downloaded from the specified sources:

TF Binding site data: A total of 23 *mus musculus* PFMs were obtained from the JASPAR database (Sandelin *et al*., 2004). Because each PFM consists of frequencies of each symbol σ at position j of the matrix, we derived a consensus TFBM for each matrix as follows. For each position j of the matrix, the highest frequency of the symbol σ (A, T, G or C) was taken as the first nucleotide base at position j of the consensus. The next most frequent symbol from the remaining 3 was then added to the same position of the consensus (separating each symbol at that position of the consensus with the / delimiter) until the least frequent symbol was added. Symbols with a frequency count of 0 were not added to the consensus. Once consensus binding sites were derived from all 23 PFMs, they were stored in an excel file together with the PFMs, names, classes, species and accession identifiers of the matrices.

Promoter data: The complete set of 22,549 mouse promoter sequences were obtained from PromoSer (http://biowulf.bu.edu/zlab/PromoSer/), where each promoter sequence consisted of a length of 2000 bases upstream and 100 bases downstream of the Transcription Start Signal (TSS). MatchMiner (http://discover.nci.nih.gov/matchminer) and the gene Id converter (http://idconverter.bioinfo.cnio.es/) tools were used to convert their locus id's into genbank accession Id's where necessary. The promoter sequences and their corresponding identifiers were then stored in an excel file as two separate columns.

3.4.2 The Scan and Score algorithm

We have developed an algorithm to scan promoter sequences in search for each consensus site representing a PFM of a specific TF. Before the algorithm is executed, the user must provide the system with a set of differentially expressed genes as a text file and in the format

"Genbank ID", "Gene name/description" and "Fold Change/Expression value", each separated by a tab delimiter. Following this, BiSAn fetches the promoter sequences using the genbank Ids of the input genes at which point the algorithm is executed as shown below.

```
One: Algorithm: Scanning promoter sequences for consensus sites

1)  Require: $x$ number of differentially expressed genes
2)  For $X = 1$ to $X = X^{th}$ gene, do
3)  Pull promoter sequence $Y$ of length $L$ for gene $X$
4)    For position $i = 1$ to $i = i^{th}$ of $Y$, do
5)      For position $j = 1$ to $j = j^{th}$ of consensus binding site $C$ from matrix $M$, do
6)        Search consecutive symbols $\sigma$ {A, T, G, C} from position $j$ to $j^{th}$ of consensus
    binding site $C$ in promoter sequence $Y$ (from $i = 1$ to $i = i^{th}$)
7)          If exact match found, calculate similarity score, $S$ of matched consensus $C$ from
    matrix $M$
8)          End if
9)      End for
10)   End for
11) End for
```

Calculating similarity scores, S

Similarity scores are calculated conditional upon a match found between a consensus binding site and the region of a given promoter site that contains the consensus. Once a match is found, the similarity score S, is calculated by taking the first base of the consensus, retrieving it's score from the matrix and dividing it by the total frequency count of all bases at that specific position of the matrix. This calculation is repeated for each and every subsequent base of the consensus and once all scores have been calculated, the overall value is derived by adding each resulting score followed by dividing the sum by the total length of the matrix and then multiplying by 100 to generate a percentage similarity score.

Two: Calculating Similarity scores, S

$$S = \sum \frac{f\sigma j}{f_j^{\text{Total}}} / m \quad X\,100$$

$$(\sigma \in \{A, T, G, C\})$$

Where:

$j = 1, 2, 3, 4, 5, 6. \quad m$

$f\sigma j$ = frequency of symbol σ at position j

f_j^{Total} = Sum of all frequencies at position j

3.4.3 Transcription Factor Binding Motif Enrichment Analysis

Following the identification of specific TFBMs within promoter regions belonging to the user's genes of interest, the next important step is to statistically determine which binding motifs are not occurring by chance alone. Therefore, BiSAn was developed to carry out Transcription Factor Binding Motif Enrichment Analysis (TFBMEA), which compares the number of differentially expressed genes that share a given consensus binding site, with the number of genes expected to share that binding site by chance alone. The difference in the observed and expected numbers can then be used to generate contingency tables, which can in turn be used as a basis to report P values for each binding site by using a hypergeometric statistical model. This work is motivated by CORNA, which is a package written in R to allow users to test their gene lists for enrichment of microRNA targets in the context of KEGG pathways and GO terms (Wu and Watson, 2009). There are also other software tools such as FatiGO (Al-Shahrour *et al*, 2004), GOStats (Falcon and Gentleman, 2007), GSEA (Subramanian *et al*, 2005), and MappFinder (Doniger *et al*, 2003) that use a hypergeometric distribution or a probability distribution to generate P values in the milieu of functional annotation.

BiSAn firstly generates a 2 x 2 contingency table for each of the 23 consensus binding sites, and then it uses the Fisher's exact test (hypergeometric model) to derive P values from them. In order to generate the contingency tables, each consensus binding site is scanned in the entire mouse genome (collection of 22,549 promoter sites) and their number of occurrences is calculated. Likewise, each consensus binding site is scanned in the promoter regions of the user's set of differentially expressed genes and subsequently counted for their number of occurrences. A contingency table for each consensus binding site is then derived as follows:

	Chosen	Not Chosen	Total
TFBM	***a***	***b***	***a + b***
Absent	***c***	***d***	***c + d***
Totals	***a + c***	***b + d***	***n***

Where: a = No of genes in user's gene list that contain the binding site (chosen from gene list)

b = No of genes in the genome that contain the binding site minus a (Not chosen from gene list)

c = No of genes in user's gene list that do not contain the binding site (chosen from gene list)

d = 22,549 (Total no of genes in genome) – (a + b + c)

n = Sum of each total (Grand total)

These contingency tables reflect the degree of enrichment of a given binding site in the user's set of differentially expressed genes (gene list) relative to the entire genome. Once contingency tables have been generated for all consensus binding sites, BiSAn then applies the Fisher's exact test to compute P values for each binding site from their contingency tables using the following hypergeometric distribution:

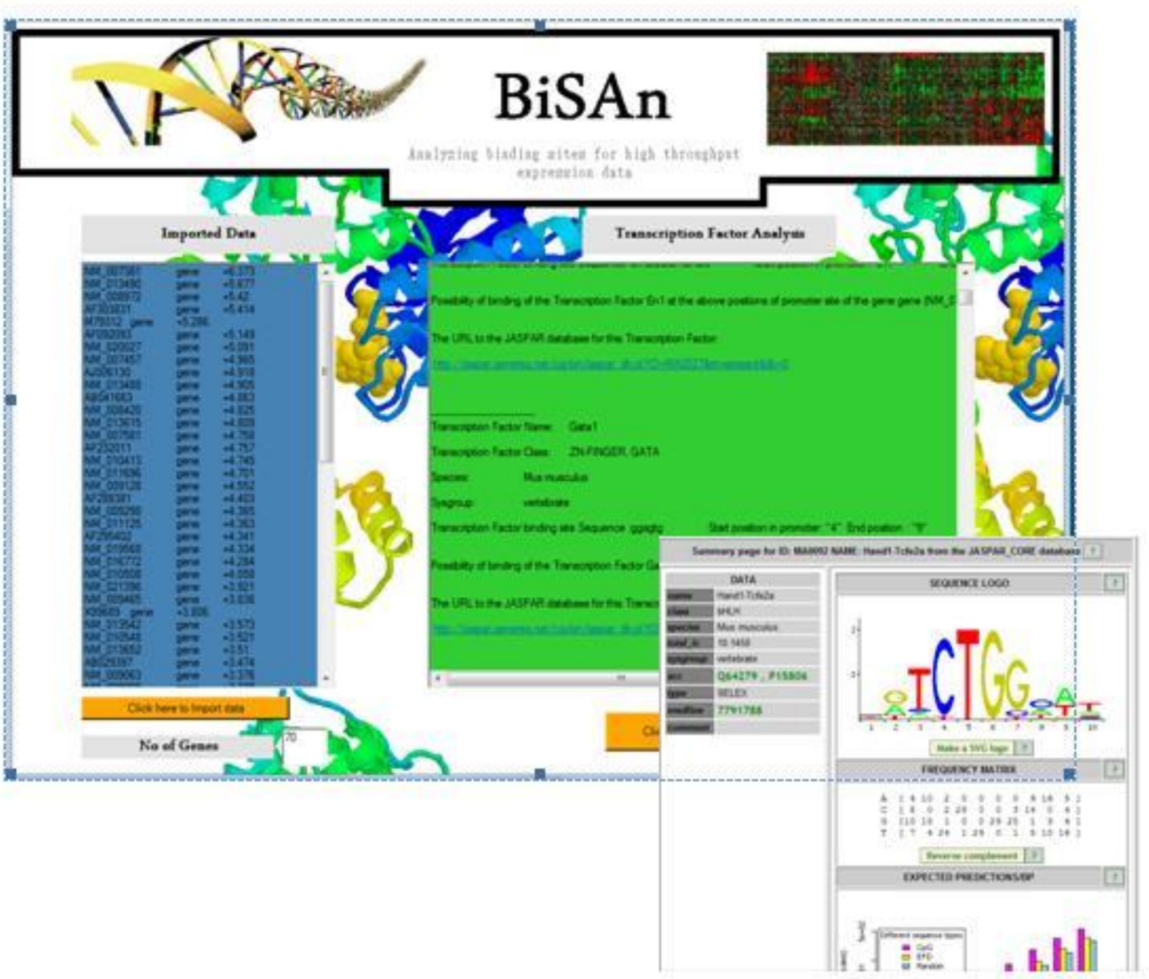
$$p = \binom{a+b}{a}\binom{c+d}{c} \Big/ \binom{n}{a+c} = \frac{(a+b)!(c+d)!(a+c)!(b+d)!}{n!a!b!c!d!}$$

Where $\binom{n}{k}$ = binomial coefficient and ! = factorial operator

The fundamental motivation behind the development of BiSAn is as follows. Although there are software available that specialize in promoter analysis for TFBM detection, complications arise for the common biologist who may wish to efficiently analyze their microarray data. The most noteworthy problem is related to a lack of consensus for the use of gene identifiers when representing genomic data. Because of this, the biologist is faced with the problem of ensuring that their microarray data representation complies with the type(s) of identifiers required by these software. BiSAn has been specifically designed to accept Genbank accession identifiers, which is by far the most commonly used to represent microarray data. Furthermore, TFBM detection is usually complicated by a low level of conservation; hence matrices are rapidly becoming the chosen methodology for TFBM representation. The goal of this work is to take advantage of this methodology for high throughput analysis specific for microarray data by providing a simple yet effective pipeline for biologists to detect TFBMs that may be implicating the biological phenomenon under investigation.

3.5 Results and Discussion

BiSAn was written in Perl. Visual Basic.Net was used to construct the Graphical User Interface for the output to be displayed to the user. The software and associated data files are freely available and can be downloaded from the project website (http://www.1066technologies.co.uk/bisan). BiSAn requires users to provide their data as a text file, where the first column contains the genbank accession identifier, the second column contains the gene name or description, and the third column contains the expression value or fold change (each column separated by a tab delimiter). Once imported, the underlying algorithm of BiSAn (section 3.4.2) is executed and the gene identifiers from the input data are used to fetch their corresponding promoter sequences. These sequences are then fully



Possibility of binding of the Transcription Factor En1 at the above positions of promoter site of the gene gene (NM_[illegible]

The URL to the JASPAR database for the Transcription Factor

[illegible]

Transcription Factor Name: Gata1

Transcription Factor Class: ZN-FINGER, GATA

Species: Mus musculus

Sysgroup: vertebrate

Transcription Factor binding site Sequence: ggagtg Start position in promoter: "4" End position: "9"

Possibility of binding of the Transcription Factor Ga[illegible]

The URL to the JASPAR database for the Transc[illegible]

Summary page for ID: MA0092 NAME: Hand1-Tcfe2a from the JASPAR_CORE database

DATA	
name	Hand1-Tcfe2a
class	bHLH
species	Mus musculus
total_ic	10.1450
sysgroup	vertebrate
acc	Q64279 , P15806
type	SELEX
medline	7791788
comment	

Figure 3.1 The Graphical User Interface (GUI) of BiSAn. The left panel displays the users imported gene expression profile and the right panel displays the output from the analysis, including web links, which when clicked display the profiles of the relevant Transcription Factor in JASPAR.

3.5.1 Application of BiSAn

To demonstrate the benefits that can be derived from analysing microarray data using BiSAn, we utilized data generated from our in-house microarray studies to evaluate how well our pipeline could be used to predict new outcomes. The biological question pertinent to this study was to unravel the underlying molecular mechanisms dictating immune tolerance by analysing the role of Egr-2 in implicating T cell tolerance. Recently characterised as a candidate tolerance-inducing transcription factor, the Early Growth Response gene (Egr-2) is known to interact with specific genes to implicate the state of tolerance in T cells (Safford *et al*., 2005; Warner *et al*., 1999). The aim of this microarray experiment was to generate differentially expressed genes by comparing tolerant Vs activated CD4+ T cells taken from mice, using an oligonucleotide chip consisting of approximately10, 000 known mouse genes (the accession number of the array data is e-mexp-283, accessible via the ArrayExpress website (http://www.ebi.ac.uk/arrayexpress/)). From this microarray experiment, 70 differentially expressed genes were identified (fold change >= 1.5) from which 8 genes were specifically confirmed to be highly upregulated by Reverse Transcriptase PCR (Table 3.1).

Table 3.1 8/70 differentially expressed genes confirmed to be highly up-regulated in T cell tolerance by Reverse Transcriptase PCR (RT-PCR)

Gene ID	HUGO ID	Fold Change (Tolerance Vs activated)
NM_007457	Ap1s1	4.965
NM_009168	Shd	2.838
NM_009298	Surf6	4.365
NM_009510	Vil2	3.151
NM_010548	Il10	3.521
NM_013532	Lilrb4	2.111
NM_019507	Tbx21	1.595
NM_021396	Pdcd1lg2	3.921

We inputted these 70 differentially expressed genes into BiSAn in order to assess its ability to carry out TFBM detection under high throughput conditions, whilst determining how well it could be used to better understand the molecular mechanisms underlying immune tolerance. From the input, 47 genes were found by BiSAn that contained known mouse TFBMs in their promoter sequences. A total of 23 TF PFMs were scanned against these promoter sites by the algorithm with an overall runtime of less than 7 minutes (Table 3.2).

Table 3.2 Some statistics to show the efficiency of BiSAn when scanning promoter sites for high throughput TFBM detection

No of genes imported into BiSAn	No of gene promoters containing TFBMs	No of PWMs scanned	No of TFs identified	Runtime of algorithm (Mins)
70	47	23	23	<7

For each promoter site scanned, BiSAn detected TFBMs pertaining to several different TFs (such as GATA1, Klf4, Nobox, Pax4, Pax5, ELF5 and Myb) and where a consensus site was found in the promoter, it efficiently computed the similarity score of the consensus to its

Gene ID	TFBMs	Binding affinity (%)

PFM. Because 8/70 genes from our tolerance dataset were confirmed to be highly regulated, we decided to place special emphasis on understanding the results generated for these putatively interesting genes. Among several other TFs, BiSAn detected TFBMs for the GATA1 Transcription factor in promoter regions of these 8 tolerance-related genes with their binding affinities ranging from 47.17% to 32.08% (Table 3.3). Similarly, scores calculated for the Klf4 transcription factor ranged from 33.26% to 13.91% (Table 3.3).

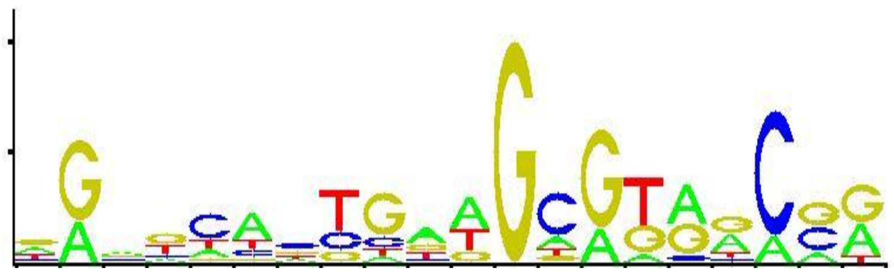
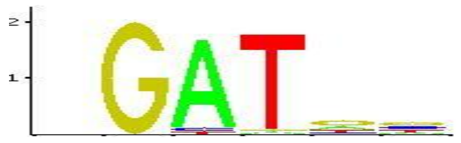
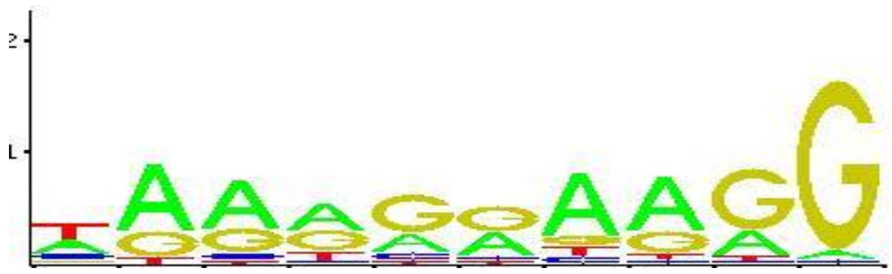
NM_007457 NM_009168 NM_009298 NM_009510 NM_010548 NM_013532 NM_019507 NM_021396	Transcription Factor = Pax5 	0 46.25 37.50 33.75 35.83 29.58 31.25 29.58
NM_007457 NM_009168 NM_009298 NM_009510 NM_010548 NM_013532 NM_019507 NM_021396	Transcription Factor = GATA1 	0 32.08 32.70 42.77 43.71 42.77 0 47.17
NM_007457 NM_009168 NM_009298 NM_009510 NM_010548 NM_013532 NM_019507 NM_021396	Transcription Factor = Klf4 	0 26.09 33.26 22.48 13.91 N/A 0 15.43

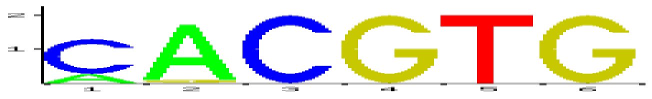
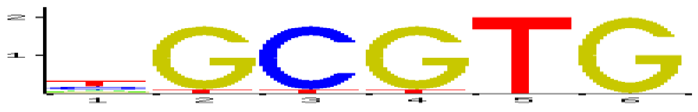
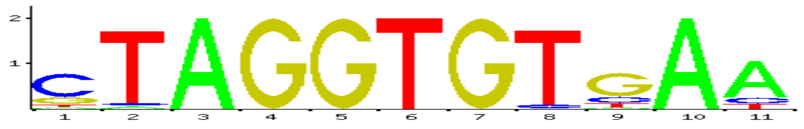
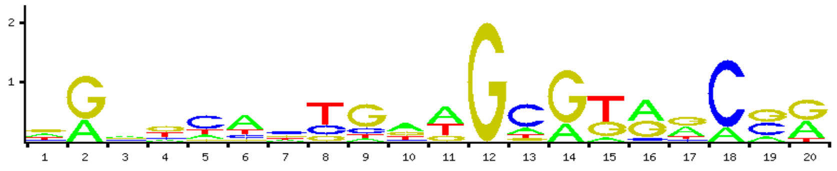
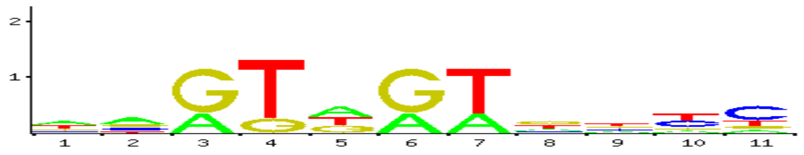
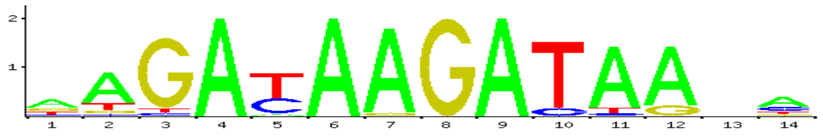
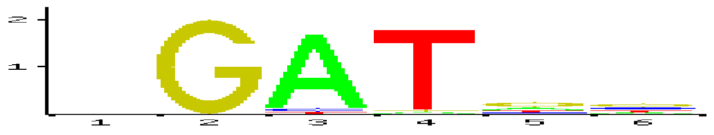
Table 3.3 Promoter analysis of 8/70 genes tolerance genes (confirmed by RT-PCR) for TFBM detection together with scores calculated for each consensus site found in their promoter regions.

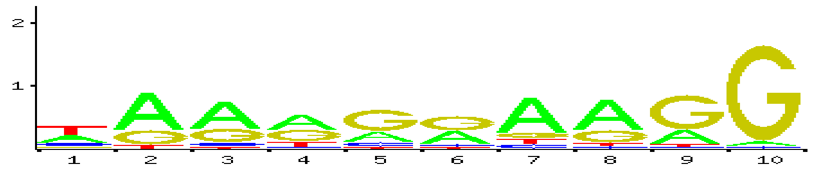
The GATA binding protein (GATA1) TF is known to differentially suppress the expression of CCR5 (a major HIV-1 co-receptor and critical determinant of HIV-1 infection) in stem cell derived dendritic cells and primary human T-cell subsets (Sundrud *et al*., 2005). This is indicative that GATA1 plays a role in the induction of immune tolerance. Klf4 has also been attributed to suppressing the immune response due to its ability to suppress B cell proliferation (Yusuf *et al*., 2008). Results generated from BiSAn show that these aforementioned TFs have the potential to bind to promoter sequences of our 8 genes confirmed to be over-expressed in immune tolerance. TFs do not necessarily bind to promoter sequences with a 100% binding efficiency in the cell. Hence, scores in the range of 30-50% may be significant and may reflect the putative ability of these genes to bind to these 2 TFs.

Once putative binding efficiencies of the motifs have been successfully calculated, the next logical step in the analysis is to identify consensus binding motifs that are enriched in the user's gene expression datasets in relation to the entire genome. Hence, BiSAn was developed to cater for this function. Solely relying on binding efficiencies of consensus sequences can pose a problem at the statistical level because such an approach lacks the ability to discern consensus binding sites that are not occurring in the user's gene list by chance alone. In light of this BiSAn implements the TFBM Enrichment Analysis function, which generates a 2 x 2 contingency table for each consensus binding site and subsequently applies the Fisher's Exact test to generate a P value for each site (refer to section 3.4.3 for a detailed description of the implementation of TFBM Enrichment Analysis). Results generated from this enrichment analysis showed that a very few consensus sites were enriched in our tolerance dataset since P values generated for most of the sites were quite high (Table 3.4). The most noteworthy consensus site belonged to the transcription factor T, with a P value of

0.065, which indicates a statistically significant degree of enrichment (Table 3.4). The T transcription factor is known to act as an activator of transcription, which binds to its target DNA as a homodimer. It is biologically interesting that the consensus binding site for a transcriptional regulator was found to be enriched in our immune tolerance dataset because there is a possibility that the transcription factor T may be hampered from interacting with these genes via the consensus site in order to induce the state of T cell tolerance. However, this is something that needs to be confirmed through wet lab experimentations.

Table 3.4 Promoter analysis of 8/70 genes tolerance genes (confirmed by RT-PCR) for TFBM detection together with scores calculated for each consensus site found in their promoter regions.

JASPAR ID	Transcription Factor Name	Consensus binding site	P Value
MA0004	Arnt		0.900
MA0006	Arnt-Ahr		0.994
MA0009	T		0.065
MA0014	Pax5		0.984
MA0027	En1		0.997
MA0029	Evi1		0.754
MA0035	Gata1		1.000

MA0039	Klf4		1.000

3.6 Concluding remarks

In this chapter, we have shown that utilizing PFMs for the purpose of scanning promoter sites for high throughput TFBM detection works well for deciphering microarray data. We have demonstrated this by developing BiSAn, which is a simple yet effective pipeline for the common biologist who may wish to input their gene expression data into a easy-to-use software to facilitate efficient TFBM detection. The algorithm underlying BiSAn is computationally fast and results generated from it can be used to make valid *in silico* predictions to aid the biologist in answering their research question.

Chapter 4

4

Genome Interactions Analyzer: a systems biology approach for the global analysis of transcriptional networks in microarray data

4.1 Introduction

Cis- regulatory sequences are frequently present in the genome, acting as recognition sites for the binding between DNA and RNA polymerase. Such regulatory sequences are recognised by and subsequently bind to specific transcription factors, which governs the increase or decrease of binding between RNA polymerase and genomic DNA. Consequently, the strength of the latter binding determines the expression of the specific gene being observed. Identifying the potential interactions between transcription factors and genes via these regulatory binding sequences is crucial towards enhancing our understanding of molecular mechanisms, such as cell differentiation and proliferation because these interactions directly implicate gene expression. However, to gain a systems level understanding of cellular function, it is equally imperative to identify protein-protein interactions that occur within the cell, which may be indirectly affecting transcription (i.e. by a specific protein carrying the transcription factor from the cytoplasm to the nucleus). Also, to complete the cellular picture, the aforementioned classes of interactions need to be placed in the context of molecular pathways to identify the involvement of other biological molecules and signalling cascades that may be playing important roles in the molecular mechanism under observation (For a review of these classes of interactions, refer to Chapter 2 of this thesis).

Although the technological breakthrough of DNA microarrays has revolutionised the field of molecular biology by providing a platform for biologists to quantify the mRNA expression levels of entire genomes concurrently in a single experiment, such high-throughput experiments tend to generate vast amounts of raw data making it problematic for the researcher to extract meaningful biological insight. It is at this stage where the use of appropriate computational tools becomes necessary for the purpose of mining the raw data to yield specific biologically significant data as per the biologist's needs. The need for data mining however, arises not only due to data overload but also more importantly because the raw data itself does not provide a *complete* picture in the context of systems biology due to its lack of integration with external sources of knowledge. Hence the use of relevant computational tools/software become necessary if the biologist is to gain meaningful deductions about their data generated from in-house microarray studies.

Currently, data pertaining to transcription factor binding sites are stored in database applications such as TRANSFAC (Wingender *et al*., 1996), Jaspar (Sandelin, *et al*., 2004) and the Object-Oriented Transcription Factor Database (TFD) (Ghosh, 1998). In addition, there are software available such as Expander (Shamir *et al*., 2005), INCLUSive (Thijs *et al*., 2002), Genesis (Sturn *et al*., 2002), CONFAC (Karanam *et al*., 2004) and GEPAS (Herrero *et al*., 2003), that specialise in carrying out promoter analysis of genes from microarray data to identify common transcription factor binding sites. Although these applications focus on streamlining the analysis flow of microarray data analysis by performing specific functions such as k-means clustering, promoter analysis and functional analysis, they do not offer a complete systems level approach to analysing gene expression data. This is because they lack the faculty to map gene expression data to biological pathways, ultimately providing an incomplete biological representation of the user's data. The lack of such ability impedes the process of obtaining a complete representation because biological pathways can provide vital information about how biological systems are organised (Chung *et al*., 2004). Currently, a number of databases exist, which store data related to transcription factor binding sites, protein-protein interactions, molecular complexes and pathways. Examples of such databases include BIND (Bader *et al*., 2001), PANTHER (Thomas *et al*., 2003) and ABS (Blanco *et al*., 2006). Whereas the focus of ABS is inclined towards manually curating experimentally identified transcription factor binding sites identified in promoters of orthologous vertebrate genes and storing this data in the form of a web-based database, its purpose is not to act as an interactive software tool that can be used in an automated or semi-automated fashion by its users. Hence ABS acts more like a repository of transcription factor binding site information rather than a user interactive software. PANTHER (Thomas *et al*., 2003) and BIND (Bader *et al*., 2001) focus on the storage of high throughput protein sequences, molecular complexes and pathways but share the same problem as ABS (Blanco *et al*., 2006) in terms of acting as static databases lacking user-interactive capabilities. Currently, the pathway database of KEGG (Kanehisa *et al*., 2004) forms a central repository of biological pathways. However, its pathway diagrams are static in nature and KEGG itself cannot be directly used for the purpose of mapping gene expression data to biological pathways. Although users can connect to KEGG via a SOAP interface in order to analyse their microarray data, doing so would be problematic for biologists who lack the necessary computational skills required to write programs. BioCarta (http://www.biocarta.com); another application dedicated to biological pathways, allows users to map gene names to biological pathways but on a gene-by-gene basis only.

The current situation is such that there are no programs available to identify *Cis*-regulatory sequences in genomic promoter sites and protein-protein interactions, from differentially expressed genes coupled with mapping these genes/proteins to an up-to-date repository of molecular pathways, all in a single software package. Making such software available is essential for biologists, especially for those actively involved in conducting microarray experiments to unravel previously unknown molecular mechanisms. This is predominantly because gene expression measurements alone are not sufficient to answer complex biological questions since they represent the amount of gene expression without explaining the molecular causalities that lead to the expression. In light of this, a global approach to analysing microarray data is required, where users are able to integrate their gene expression data in the context of transcription factor binding sites, protein-protein interactions and molecular pathway analysis, all at the same time. Only then will users be able to extract a global visualisation of cellular activities, which implicate the molecular mechanism being investigated by them.

In this chapter, a bioinformatics software package called Genome Interactions Analyser (GIA) is described, which focuses on the interpretation of data generated from gene expression microarray experiments and deciphers it in the context of promoter analysis to identify transcription factor binding sites, protein-protein interactions and molecular pathways for both mouse and human species all at the same time. GIA specifically aims to act as a microarray data analysis system and has been designed to efficiently extract global-level biological meaning from gene expression data in order to aid in the exploitation of molecular mechanisms investigated by biologists. Because GIA is a systems-biology software, it forms an attractive medium for biologists to meaningfully analyse their high throughput data.

4.2 Methodology

GIA was programmed in Visual Basic.NET and Perl and MySQL was used to construct its underlying databases, built for the purpose of generating the hierarchical tree-view structures. Before GIA is executed, users are required to provide a tab-delimited text file containing gene expression data (in the format: Genbank Accession ID, Fold Change/Expression Value, and + or – sign to differentiate up and down regulated genes), which they wish to process. Following this the user is required to connect to the MySQL database containing all the underlying data used by GIA. Once connected, the user will be able to begin their analysis. Refer to Figure 4.1 for a complete illustration of the functions.

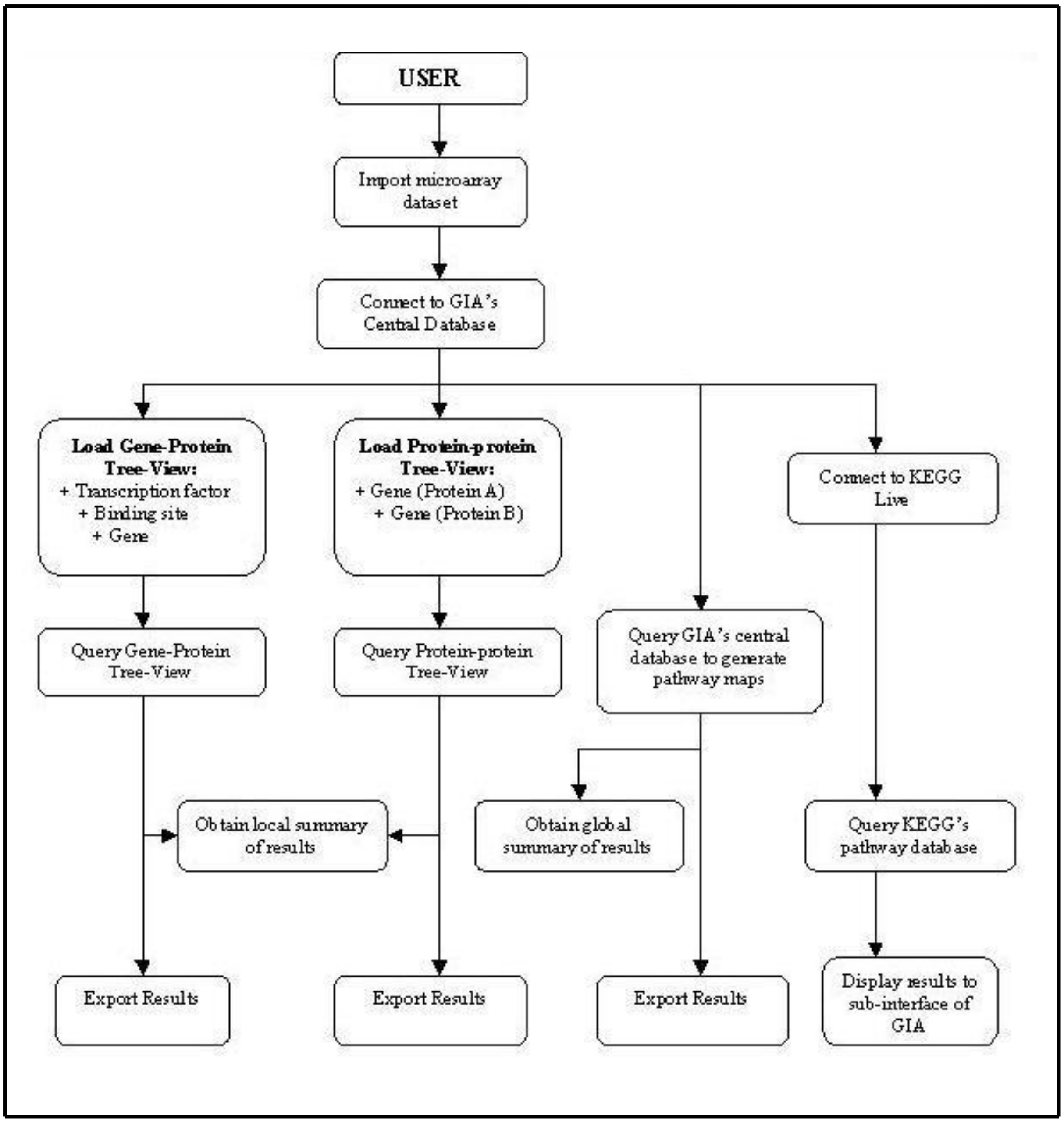

USER
Import microarray dataset
Connect to GIA's Central Database
Load Gene-Protein Tree-View:
+ Transcription factor
+ Binding site
+ Gene
Load Protein-protein Tree-View:
+ Gene (Protein A)
+ Gene (Protein B)
Connect to KEGG Live
Query Gene-Protein Tree-View
Query Protein-protein Tree-View
Query GIA's central database to generate pathway maps
Obtain local summary of results
Obtain global summary of results
Query KEGG's pathway database
Export Results
Export Results
Export Results
Display results to sub-interface of GIA

Figure 4.1 Functional layers of GIA. Once user's data is queried against GIA's database, processed data is displayed in the relevant panel on the interface, after which the user can obtain a local summary of their results (specific to the function of GIA they have executed). Following live connection with KEGG, user's genes are queried against KEGG's pathway database. Results (including the links for each pathway) are then displayed on GIA's sub-interface.

4.2.1 Gene-Protein Interaction Function

The first function of the software (called "Gene-Protein Interaction") is used to generate the hierarchical tree-view structure representing the associations of transcription factors, *Cis* regulatory binding sites and genes containing those binding sites within their promoter regions. The associations are organised in the form of grandparent, parent and child nodes respectively and each node is built as a result of querying GIA's database (Figure 4.1 and 4.2). When this tree-view is queried against the user's data, genes found within the tree-view structure are highlighted in red (for up-regulated genes) and green (for down-regulated genes). The colour codes are based on the thresholds that users set for their input data. When users import their expression data into GIA, they will be prompted with an option to set their thresholds for both up and down-regulated genes. If for example, they choose >+1.5 (for up-regulated) and <-1.5 (for down-regulated) as their thresholds, genes from their dataset (meeting these thresholds) found within the tree-view will be highlighted as red and green respectively. Also because the user's data may also contain transcription factors, these will also be highlighted in the tree-view if found. Following this, users can then use this function to generate a statistical profile for each cluster of transcription factors, comprising of the number of genes, *Cis* regulatory binding sites and mean fold change expression belonging to each cluster. These statistics serve the purpose of identifying clusters of genes from the user's data that are enriched with specific transcription factor binding sites. The purpose of these features is for users to identify genes from their gene expression datasets that contain regulatory binding motifs for specific transcription factors which can then be correlated to the vital information of whether they have been up or down-regulated in their particular microarray experiment. Our motivation for designing and implementing this aspect of our software revolves around the belief that transcription factor binding sites found within promoter sites of genes can play integral roles in governing unknown gene function and having additional gene expression information (i.e. Up or down-regulated signals) can aid in making predictive inferences about the data at the DNA-Protein interaction level.

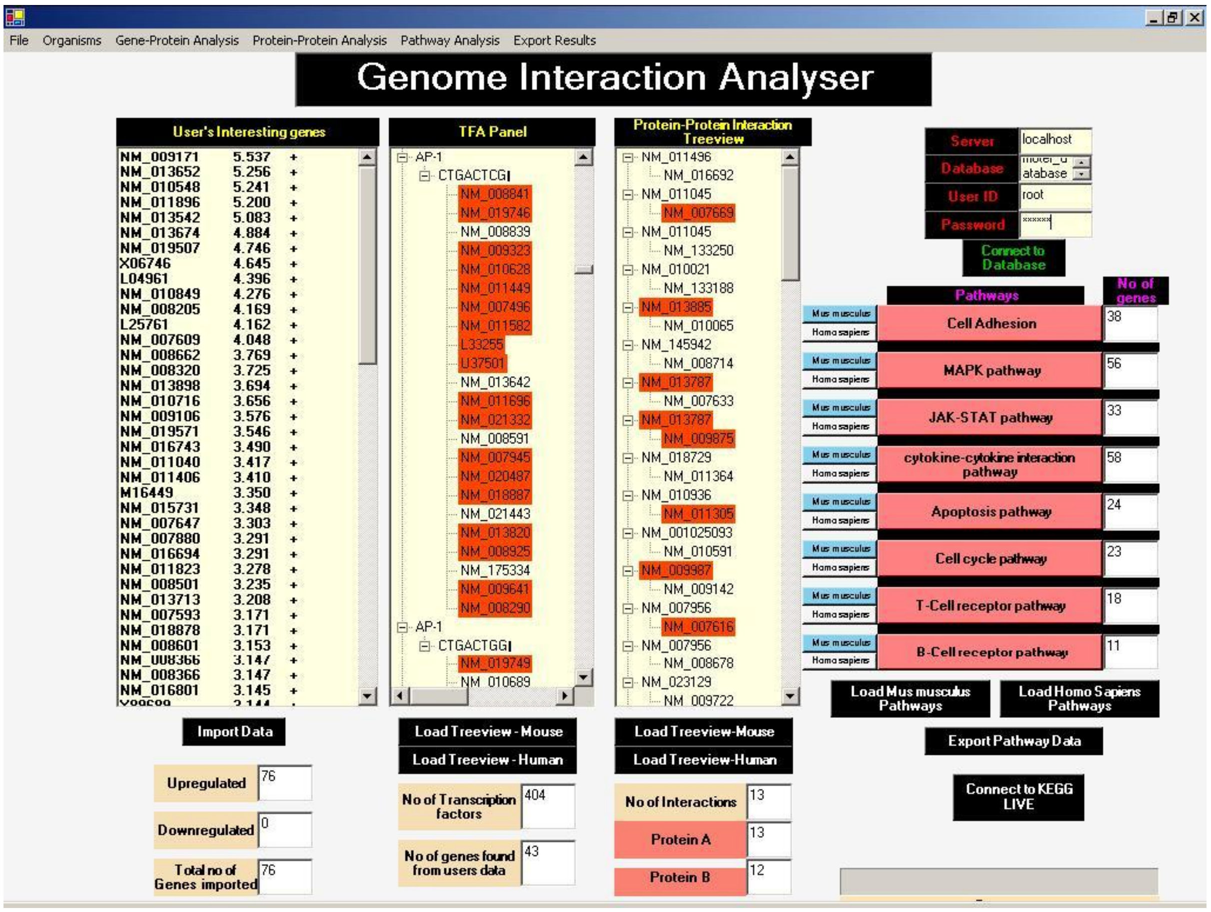
4.2.2 Protein-protein Interaction Function

The second panel (called “Protein-protein interactions”) is used to generate a hierarchical tree-view structure specific to protein-protein interactions and GIA’s database is queried in order to build the tree-view (Figure 4.1 and 4.2). Genbank accession numbers for protein “A” are represented as parent nodes while accession numbers for their interacting proteins (Protein “B”) are represented as child nodes. Once the protein-protein interaction tree-view is queried against the users gene expression data, those genes found within the tree-view are highlighted accordingly (Figure 4.2). Because gene expression datasets typically consist of accession numbers pertaining to both genes and proteins, having such functionality would give biologists the opportunity to identify proteins within their datasets that interact with other proteins not necessarily found within their data. At any point in the analysis stage, the user can export a local summary of their results, delineating the associations of genes (within their dataset) with transcriptions factors to which they have been assigned to (for Gene-Protein Interaction) or alternatively associations between proteins found within their dataset with other interacting proteins (for Protein-Protein interactions). In the case of both types of analyses, the exported data also provides names for molecular pathways in which each gene/protein is found to play a role in.

4.2.3 Molecular Pathway Analysis

To gain a systems level understanding of gene expression data, it is essential for the analysis to encompass an understanding of what roles user’s genes of interest play in currently known molecular pathways. This is important since molecular pathways represent knowledge-base and can be used to validate predictive findings. This function of GIA (Pathway Maps function) is divided into two specific sub-functions. The first allows users to map their expression data to eight core pathways that are known to be pertinent to a broad range of molecular processes (such as immunology and cancer) and the second sub-function allows users to map their data to *all* molecular pathways known to date. The eight core pathways were developed specifically for immunologists and cancer researchers interested in efficiently analysing their immunology-specific or cancer-specific gene expression data.

For the first sub-function, the co-ordinates for both the nodes (biological entity) and associations (relationships) comprising each pathway are stored in appropriate data structures. When the user's data is queried against the nodes for the biological entities of each pathway, a count is provided on GIA's interface for the number of genes from the user's data that are found in each of the pathways (Figure 4.2). Knowing which pathways are worthy of exploration, the user then has the option to select a pathway of their choice. When selected, the pathway map is constructed on the sub interface (independent interface) of GIA. Concurrently, the fold changes for each gene entry present in the user's gene expression data are searched to establish whether the fold changes are + or – and subsequently, the genes from the users data found within the pathway are highlighted accordingly in order to indicate to the user the genes that have been up or down-regulated in their particular microarray experiment. At this point in the analysis stage, users can export a summary of their pathway analysis findings.



NM_009171	5.537	+
NM_013652	5.256	+
NM_010548	5.241	+
NM_011896	5.200	+
NM_013542	5.083	+
NM_013674	4.884	+
NM_019507	4.746	+
X06746	4.645	+
L04961	4.396	+
NM_010849	4.276	+
NM_008205	4.169	+
L25761	4.162	+
NM_007609	4.048	+
NM_008662	3.769	+
NM_008320	3.725	+
NM_013898	3.694	+
NM_010716	3.656	+
NM_009106	3.576	+
NM_019571	3.546	+
NM_016743	3.490	+
NM_011040	3.417	+
NM_011406	3.410	+
M16449	3.350	+
NM_015731	3.348	+
NM_007647	3.303	+
NM_007880	3.291	+
NM_016694	3.291	+
NM_011823	3.278	+
NM_008501	3.235	+
NM_013713	3.208	+
NM_007593	3.171	+
NM_018878	3.171	+
NM_008601	3.153	+
NM_008366	3.147	+
NM_008366	3.147	+
NM_016801	3.145	+

```
AP-1
  CTGACTCG
    NM_008841
    NM_019746
    NM_008839
    NM_009323
    NM_010628
    NM_011449
    NM_007496
    NM_011582
    L33255
    U37501
    NM_013642
    NM_011696
    NM_021332
    NM_008591
    NM_007945
    NM_020487
    NM_018887
    NM_021443
    NM_013820
    NM_008925
    NM_175334
    NM_009641
    NM_008290
AP-1
  CTGACTGG
    NM_019749
    NM_010689
```

```
NM_011496
  NM_016692
NM_011045
  NM_007669
NM_011045
  NM_133250
NM_010021
  NM_133188
NM_013885
  NM_010065
NM_145942
  NM_008714
NM_013787
  NM_007633
NM_013787
  NM_009875
NM_018729
  NM_011364
NM_010936
  NM_011305
NM_001025093
  NM_010591
NM_009987
  NM_009142
NM_007956
  NM_007616
NM_007956
  NM_008678
NM_023129
  NM_009722
```

Server	localhost
Database	atabase
User ID	root
Password	xxxxxx

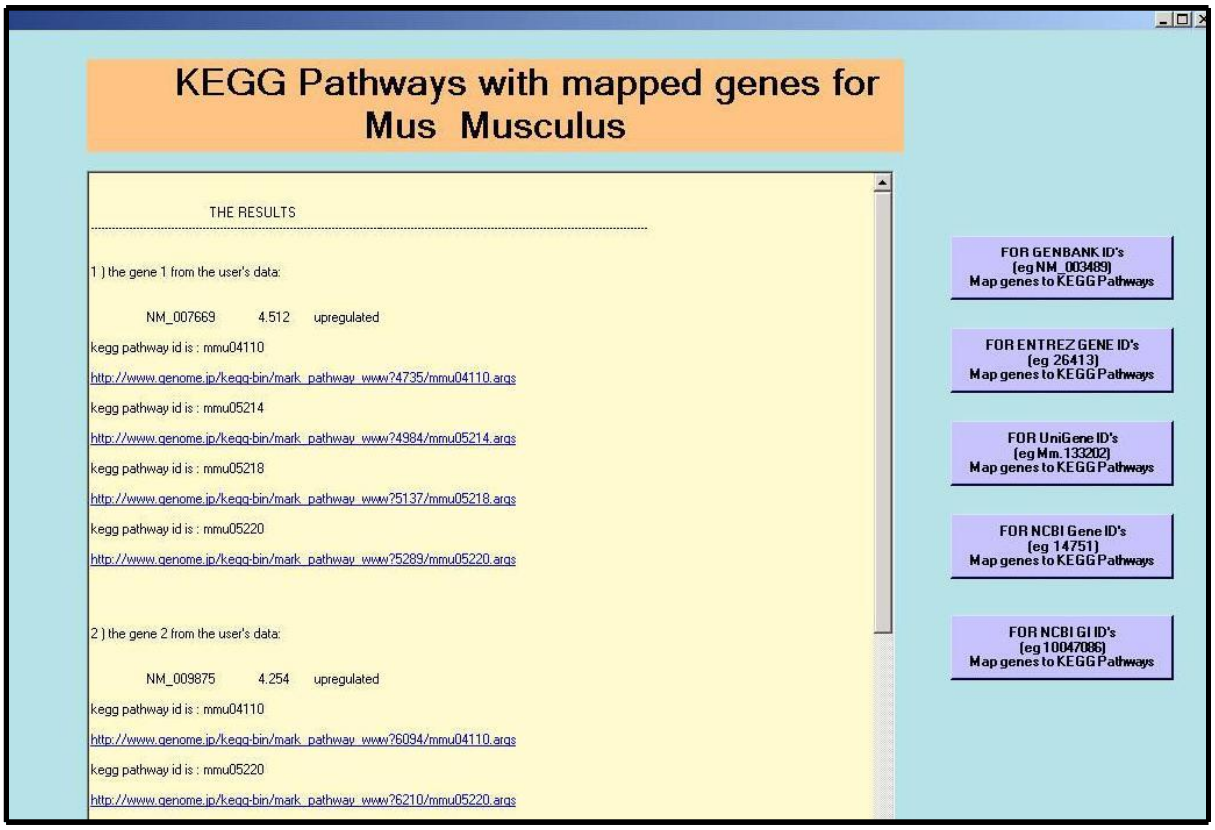
Pathways	No of genes
Cell Adhesion	38
MAPK pathway	56
JAK-STAT pathway	33
cytokine-cytokine interaction pathway	58
Apoptosis pathway	24
Cell cycle pathway	23
T-Cell receptor pathway	18
B-Cell receptor pathway	11

Each pathway: Mus musculus / Homo sapiens

Upregulated	76
Downregulated	0
Total no of Genes imported	76
No of Transcription factors	404
No of genes found from users data	43
No of Interactions	13
Protein A	13
Protein B	12

Figure 4.2 Main Graphical User Interface of GIA. The first panel displays the user's interesting genes, where +/- Signs are used to indicate up or down-regulated genes. The second panel called "TFA" is used to display the hierarchical structure representing the associations between transcription factors, binding sites and genes. Those genes found within the user's data are highlighted accordingly. The third panel called "protein-protein interaction tree-view" is used to generate the hierarchical structure representing the protein-protein interactions. Finally, for each local pathway, a count is provided which reflects the number of genes from the user's data found within the pathways. Clicking on either organism type (*mus musculus* or *homo sapiens*) generates the pathway maps. Also, clicking on the "connect to KEGG live" button takes the users to the sub-interface, where they can map their expression data to live KEGG pathways at the click of a button.

For the second sub-function, GIA has been designed to automatically connect to KEGG (Kanehisa *et al.*, 2004) in a live fashion in order to map user's gene expression data to all molecular pathways known to date. Once GIA has searched for the user's genes in all KEGG pathways, the pathway links are displayed on the second sub-interface of GIA (Figure 4.3). Clicking on each of these links will generate the specific KEGG pathway in html and the user's genes will be highlighted in red or green depending on whether the genes are up or down regulated (Figure 4.4). In order for GIA to establish a live connection with KEGG's pathway database, the KEGG API is accessed in Perl. Also, Perl script was written for GIA to specifically 1) search for user's genes in all of KEGG pathways, 2) return the results of the search (including pathway links) to GIA's sub-interface for users to scrutinise and 3) highlight user's genes as red or green on KEGG pathways once each pathway link is clicked. GIA has been designed to connect to the KEGG API via Soap-Lite, after which the algorithm is executed (Figure 4.5).


KEGG Pathways with mapped genes for Mus Musculus
FOR GENBANK ID's (eg NM_003489) Map genes to KEGG Pathways
FOR ENTREZ GENE ID's (eg 26413) Map genes to KEGG Pathways
FOR UniGene ID's (eg Mm.133202) Map genes to KEGG Pathways
FOR NCBI Gene ID's (eg 14751) Map genes to KEGG Pathways
FOR NCBI GI ID's (eg 10047086) Map genes to KEGG Pathways

```
                THE RESULTS
--------------------------------------------------------------------------------

1 ) the gene 1 from the user's data:

        NM_007669        4.512     upregulated
kegg pathway id is : mmu04110
http://www.genome.jp/kegg-bin/mark_pathway_www?4735/mmu04110.args
kegg pathway id is : mmu05214
http://www.genome.jp/kegg-bin/mark_pathway_www?4984/mmu05214.args
kegg pathway id is : mmu05218
http://www.genome.jp/kegg-bin/mark_pathway_www?5137/mmu05218.args
kegg pathway id is : mmu05220
http://www.genome.jp/kegg-bin/mark_pathway_www?5289/mmu05220.args


2 ) the gene 2 from the user's data:

        NM_009875        4.254     upregulated
kegg pathway id is : mmu04110
http://www.genome.jp/kegg-bin/mark_pathway_www?6094/mmu04110.args
kegg pathway id is : mmu05220
http://www.genome.jp/kegg-bin/mark_pathway_www?6210/mmu05220.args
```

Figure 4.3 Sub-Interface of GIA. User's genes are searched in KEGG's pathway database and results are sent back to GIA's sub-interface for scrutiny. Clicking on each url generates the pathway image with user's genes highlighted in them. Users' can select from a number of gene identifiers, such as Genbank, Entrez, Unigene and NCBI id's to perform the search.

4.2.4 Constructing the databases for GIA

4.2.4.1 Gene-Protein Interaction Database

The Gene-Protein Interaction database was constructed in three steps. At first, data from the Object-Oriented Transcription Factor Database (oo-TFD) (Ghosh, 1998) was manipulated, which contains a list of 1617 unique eukaryotic transcription factors associated with their respective binding sites, all verified according to relevant wet-lab experiments. Then, we obtained the complete set of human and mouse promoters, using EXPANDER (Shamir *et al*, 2005) and Promoser's (Halees *et al.*, 2003) underlying databases respectively. The set of collected human promoter data comprised of 12,981 putative promoter regions for known genes, each with a length of 1200 bp (1200 bp upstream of the TSS). For the mouse data, each promoter sequence contains 2000 bases upstream and 100 bases downstream of the TSS. Having obtained all the necessary data, we then created a parallel pattern finding algorithm to search for the transcription factor binding sites within the entire human and mouse promoter sites. Since there are vast numbers of promoters within the human and mouse genomes, the algorithm needed to be of a distributed type in order to improve processing efficiency. Once binding sites of the 1617 transcription factors were found within the complete sets of human and mouse promoter sites, the genes to which the promoter sites belonged to were assigned to the transcription factors. A total of 320,000 genes for human (including multiple occurances) and 300,000 for mouse (including multiple occurances) were assigned to 1617 transcription factors by the pattern-matching algorithm. Where necessary, Matchminer (Bussey *et al.*,

2003) was used to convert gene identifiers from one type to another. For example, when the human promoter data was collected, each gene identifier was originally presented as "locus id". In this case, Matchminer was used to convert these identifiers to Genbank accession ID.

4.2.4.2 Protein-Protein Interaction Database

The Protein-Protein Interaction database was constructed using human interaction data from the Human Protein Reference Database (Peri *et al*., 2003) and mouse interaction data from the Riken Database for mouse (Suzuki *et al*., 2001). A total of 8000 known human protein-protein interactions were stored together with approximately 145 mouse protein-protein interactions where proteins within both sets of interaction data were represented as Genbank accession ID's.

4.2.4.3 Pathway maps Data

The pathway data for the eight core pathways comprises of co-ordinates belonging to the nodes (biological entities) and associations (arrows and lines). These co-ordinates were obtained from KEGG (Kanehisa *et al*, 2004) and subsequently stored in data structures. By having co-ordinates for all biological entities and associations, each pathway map was then, designed to be generated automatically. For the live KEGG function of GIA, constructing a database was not required since this function was programmed to be executed via the KEGG API using Soap-Lite.

The underlying driving force behind the development of GIA is straightforward. Regardless of the nature of the biological question, every microarray experiment will lead to the generation of a set of differentially expressed genes. Such set of genes then need to be mined in the appropriate biological context in order to better understand the molecular mechanism under investigation. The challenge however, is to carry this out in the perspective of systems biology, by not losing the bigger picture of the organism as a whole. GIA was developed

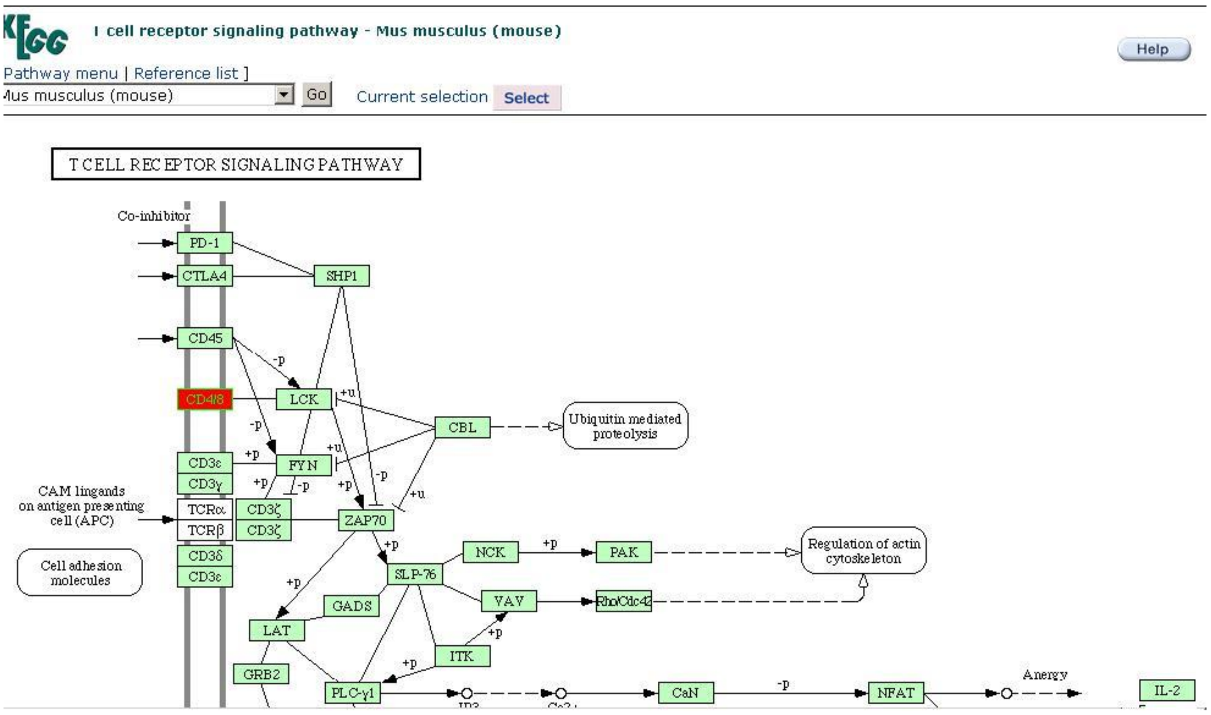

T cell receptor signaling pathway - Mus musculus (mouse)
Pathway menu | Reference list]
Mus musculus (mouse)
Go
Current selection
Select
Help
T CELL RECEPTOR SIGNALING PATHWAY
Co-inhibitor
PD-1
CTLA4
SHP1
CD45
CD4/8
LCK
CBL
Ubiquitin mediated proteolysis
CD3ε
CD3γ
FYN
CAM lingands on antigen presenting cell (APC)
TCRα
TCRβ
CD3ζ
CD3ζ
ZAP70
Cell adhesion molecules
CD3δ
CD3ε
NCK
PAK
Regulation of actin cytoskeleton
SLP-76
VAV
Rho/Cdc42
GADS
LAT
ITK
GRB2
PLC-γ1
CaN
NFAT
Anergy
IL-2

Figure 4.4 KEGG pathway, displaying the T-Cell receptor signalling pathway. User's gene expression dataset is searched through all of KEGG's pathway maps and genes that are found on the pathways are highlighted. CD4/8 was upregulated in our tolerance dataset and was found in the T cell receptor signalling pathway (highlighted in red).

whilst considering this in mind, and as a result, the implemented functions were designed to streamline high throughput analysis in the context of three principal layers; A) The promoter level – Identifying *Cis* regulatory binding sites in promoter sites of genes that have the potential to bind to specific transcription factors, B) The cytoplasmic level – Identifying proteins that can interact with other proteins in the cellular cytoplasm, and C) The pathway level – Mapping molecular interactions and signalling cascades that occur in molecular pathways within the cell. The potential benefits that could be derived from analysing high throughput data within this framework motivated us to develop GIA. The Graphical User Interface of GIA coherently reflects the faculty to analyse expression data at these functional layers and it has been ensured that the software is easy to use, and that results are processed intuitively.

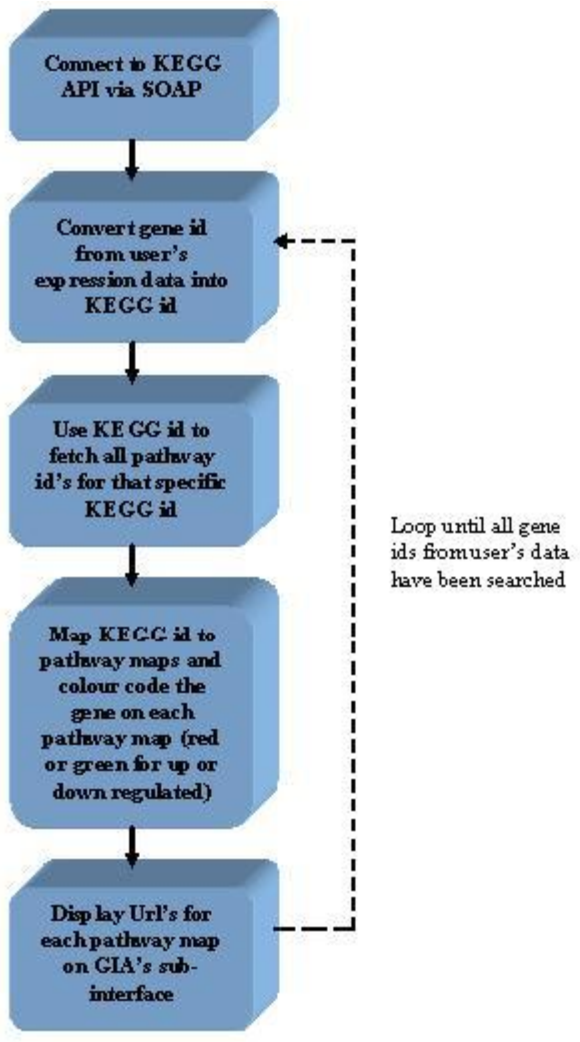

Connect to KEGG API via SOAP
Convert gene id from user's expression data into KEGG id
Use KEGG id to fetch all pathway id's for that specific KEGG id
Map KEGG id to pathway maps and colour code the gene on each pathway map (red or green for up or down regulated)
Display Url's for each pathway map on GIA's sub-interface
Loop until all gene ids from user's data have been searched

Figure 4.5 Algorithm for GIA's Live KEGG function. GIA is firstly required to establish a connection to KEGG's API via Soap-Lite. Once connected, the gene ids within the user's expression data are converted into KEGG ids. Following this, the KEGG ids are used to fetch all pathway ids (which contain each specific KEGG id within the pathway maps) from the KEGG pathway database. The KEGG ids are then mapped to their respective pathway maps and colour coded appropriately and

finally, the urls for all of the mapped pathways are sent back to GIA's sub-interface. Clicking on each url generates the pathway map in html with the highlighted genes.

4.3 Results

For the purpose of demonstrating our software's functionalities, we used data generated from our in-house microarray experiment. The biological purpose of this in-house study centred on the exploitation of the underlying molecular mechanisms concerning immune tolerance and hence, this forms our basis for further investigation. Microarray datasets were utilised to extract interesting biological knowledge from our software via three approaches. The first through the simultaneous identification of DNA-binding transcription factors that have the potential to bind to genes within a given microarray dataset, the second via protein-protein interactions and the third through a dynamic graphical approach allowing the visualisation of genes and proteins within the context of biological pathways (Figure 4.2).

4.3.1 Analysing the molecular mechanism underlying immune tolerance using GIA's Gene-Protein Interaction function

Gene-Protein Interaction allows valid inferences to be made about microarray data because it combines knowledge of transcription factors, *Cis* regulatory binding sites and genomic promoter sites with the crucial information of whether the genes being analysed have been up or down-regulated in the user's microarray experiment. We chose to use ~ 7000 regulatory binding sites from TFD to construct the Gene-Protein Interaction database because these binding sites are evolutionarily conserved in eukaryotic genomes and have been verified by wet lab experimentations. As part of the Gene-Protein Interaction function, genes found to contain evolutionarily conserved binding sites for their respective transcription factors are also searched in the pathway database of KEGG in order to integrate the user's regulatory binding site data with molecular pathway information.

Because we are interested in the molecular mechanisms underlying T-cell anergy and tolerance, we exploited this particular function of our software by utilising differentially expressed genes associated with T-cell tolerance. Querying Gene-Protein Interaction against our immune tolerance dataset revealed several transcription factors that could potentially bind to the genes involved in immune tolerance (Table 4.1). Amongst the differentially expressed genes within our tolerance dataset, AXIN1 was found to contain *Cis* regulatory binding sites within its promoter region for the EGR-2 transcription factor. Furthermore IL-10 was found to contain *Cis* binding sites for AP1 and STAT factors, while Ifnar contained binding sites for AP1 and STAT 3. When integrated with molecular pathway data, these genes were identified to be involved in the Jak-Stat, Cytokine-cytokine and T-cell receptor signalling pathways (Table 4.1).

Table 4.1 Some differentially expressed genes from the tolerance dataset assigned to their respective Transcription factors by GIA

Transcription Factor (TF)	**Accession ID of gene(s) (*)**	**HUGO ID of gene (s) (**)**	**Differential Expression (***)**	**Pathway Involved (****)**	**Citation (*****)**
STAT 3	NM_010508	Ifnar1	+3.521	JAK-STAT	Pfeffer *et al.*, 1997
STAT 3	NM_018731	Atp4a	+4.058	JAK-STAT	~
STAT Factors	NM_010548	Il10	+1.796	JAK-STAT	Benkhart *et al., 2000, 2003*
EGR-2	NM_013866	Zfp385	+1.664	Unknown	~
EGR-2	AF009011	Axin1	+2.571	Cell cycle	~
EGR-2	NM_007669	P21cip1	+4.512	Cell cycle	~
EGR-2	NM_009875	p27kip1	+4.254	Cell cycle	~
E-BOX Factors	NM_013488	Cd4	+4.905	T-Cell receptor signalling	~
E-BOX Factors	NM_013652	Ccl4	+3.51	Cytokine-cytokine Interaction	~
E-BOX Factors	NM_008420	Kcnb1	+4.825	Unknown	~
AP-1	NM_013652	Ccl4	+3.51	Cytokine-cytokine Interaction	~
AP-1	NM_019568	Cxcl14	+4.334	Cytokine-cytokine Interaction	[20]
AP-1	NM_010508	Ifnar1	+4.058	JAK-STAT	~

AP-1	NM_010548	Il 10	+3.521	JAK-STAT	[21]

(*) Accession Id's of some of the genes from the tolerance dataset that contain binding sites in their promoters for the transcription factors shown in the first column. (**) Corresponding Hugo Id's of each gene. (***) Differential expression of genes (+ indicates genes that are up-regulated). (****) Pathways in which the corresponding genes are found in. (*****) Citations for interactions between TF and gene (Interactions found by GIA that are not confirmed by published works are denoted by "~"). The complete table of differentially expressed genes can be found in the supplementary information.

The significance of integrating interaction data between transcription factors and genes with molecular pathways is to allow users of our software to 1) identify key transcription factors that may be playing important roles in the underlying molecular mechanisms being investigated in the microarray experiment and 2) identify other potentially important molecules within known biological pathways that may be directly or indirectly implicating the molecular mechanism under observation. The gene expression/fold change measurements can then be used to assess the likelihood of binding between the gene and transcription factor.

4.3.2 Mining gene expression data specific to Immune tolerance using Protein-Protein Interaction function

Besides carrying out promoter analysis to identify transcription factors that can potentially bind to genes within gene expression datasets, the focus of GIA is also to identify gene-encoding proteins within expression data, which participate in Protein-protein interactions (PPI). Having a functionality to analyse such interactions is important in a software package that specialises in microarray data analysis since the regulation of molecular mechanisms tend to be dependent on gene-specific transcription factor binding as well as interactions at the protein-protein level. The PPI function of GIA has been developed to naturally ensure the identification of protein-encoding genes from user's gene expression data, which interact with other genes within the same dataset. However, certain proteins not present within the gene expression dataset may also participate in interactions with genes within the user's data. GIA is also able to identify these interactions that are external to the user's dataset, leading to the generation of meaningful information for the users. Each identified protein is highlighted either red or green within the Protein-protein hierarchical structure (Figure 4.2). Following this, proteins are then mapped to known biological pathways to identify their interactions with other genes/proteins (Table 4.2).

Table 4.2 Protein-Protein Interactions found for some of the genes within the tolerance dataset, by GIA's Protein-Protein interaction function

Protein A (*)	**Differential Expression (**)**	**HUGO ID's for Protein A (***)**	**Protein B (****)**	**HUGO ID's for Protein B (*****)**	**Pathways Involved (******)**	**Citation (*******)**
NM_013787	+2.067	Skp2	NM_007633	Ccne1	Cell Cycle	[22]
NM_013787	+2.067	Skp2	**NM_009875**	Cdkn1b	Cell Cycle	[23]
NM_009987	+1.625	Cx3cr1	NM_009142	Cx3cl1	Cytokine-Cytokine Interaction	[24]
NM_013822	+1.696	Jag1	NM_010928	Notch2		[25]
NM_008783	+1.829	Pbx1	NM_008714	Notch1		[26]
NM_009875	+2.227	Cdkn1b	NM_009870	Cdk4	Cell cycle, T-Cell	[27]

					receptor signalling	
NM_009875	+2.227	Cdkn1b	NM_009873	Cdk6	Cell Cycle	[27]
NM_009875	+2.227	Cdkn1b	NM_016714	Nup50	Cell Cycle	[28]
NM_007560	+1.566	Bmpr1b	NM_007553	Bmp2	Cytokine-Cytokine Interaction	[29]
NM_007560	+1.566	Bmpr1b	NM_007554	Bmp4	Cytokine-Cytokine Interaction	[30]
NM_011529	+2.228	Tank	NM_019777	Ikbke		[31]
NM_011850	+1.587	Nr0b2	NM_030676	Nr5a2		[32]

(*) Accession id's of some of the protein-encoding genes (Protein A) from the tolerance dataset that were found to participate in PPI's by GIA. (**) Fold Change value of Protein A. (***) HUGO id's of Protein A. (****) Accession id's of interacting proteins (Protein B). (*****) HUGO id's of Protein B. (******) Pathways in which the interactions are found in. (*******) Citation(s) for each PPI between Protein A and B. The proteins highlighted in bold belong to the tolerance dataset, whereas the un-bold ones are external proteins not found within the user data.

Querying the Protein-Protein interaction function of GIA against our tolerance-related dataset revealed 10 specific protein-encoding genes within the dataset (comprising of ~2000 genes in total), which collectively participate in a total of 15 Protein-protein interactions with proteins external to the tolerance dataset and 2 interactions with proteins within the same tolerance dataset (Table 4.2). Amongst the 15 interactions, the Cdkn1b protein-encoding gene from the tolerance dataset was found to interact with Cdk4, Cdk6 and Nup50 and from the 2 internal interactions, Cdkn1b was found to interact with Skp2. Also Jag1 and Pbx1 were found to interact with the Notch2 and Notch1 proteins respectively.

Pathway analysis revealed the involvement of these proteins in processes such as cell cycle, T-cell receptor signalling and Cytokine-cytokine interaction (Table 4.2).

4.3.3 Biological Pathway analysis of genes involved in T-cell tolerance using GIA's Pathway maps function

GIA's pathway maps function is based on 1) the construction of 8 core widely used biological pathways for known cellular and molecular mechanisms and 2) the software's ability to connect to KEGG in a live fashion in order to map users expression data to all biological pathways known to date (Figure 4.3 and 4.4). Executing both of these functions against our tolerance dataset, we identified specific genes mainly involved in the cell adhesion, T-cell receptor signalling, Cytokine-Cytokine interaction, Jak-Stat, and MAP Kinase pathways (Refer to Table 4.3 for complete results). Our tolerance gene expression dataset comprised of 70 upregulated genes, 27 of which were mapped on to a total of 63 KEGG biological pathways at the click of a button (Table 4.3).

Table 4.3 Number and identities of each KEGG pathways mapped for 27/70 genes from our tolerance dataset.

Gene ID (*)	KEGG Pathway Ids (**)	Total number of pathways (***)	Gene ID (*)	KEGG Pathway Ids (**)	Total number of pathways (***)
NM_007381	mmu00071, mmu00280, mmu00410, mmu00640, mmu03320	5	NM_013542	mmu04650 mmu04940	2
NM_007581	mmu04010	1	NM_010548	mmu04060 mmu04630 mmu04660	3
NM_007664	mmu04514	1	NM_013652	mmu04060 mmu04620	2
NM_008008	mmu04010, mmu04810, mmu05218	3	NM_009510	mmu04670 mmu04810	2
NM_013488	mmu04514 mmu04612 mmu04640 mmu04660	4	NM_008205	mmu04514 mmu04612 mmu04940	3
NM_008420	mmu04742	1	NM_013814	mmu00512 mmu01030	2
NM_011696	mmu04020	1	NM_013521	mmu04080	1
NM_011125	mmu03320	1	NM_008601	mmu05218	1
NM_019568	mmu04060 mmu04670	2	NM_019777	mmu04010 mmu04620	2
NM_016772	mmu00350 mmu00362 mmu00628	3	NM_010102	mmu04080	1

NM_010508	mmu04060 mmu04620 mmu04630 mmu04650	4	AF303831	mmu00051 mmu00052 mmu00120 mmu00260 mmu00310 mmu00363 mmu00591 mmu00625 mmu00650	9
NM_021396	mmu04514	1	AF009011	mmu04310 mmu05210 mmu05213 mmu05217	4
NM_013490	mmu00260 mmu00564	2	AF288381	mmu04650	1

(*) Genebank accession Ids for genes from our tolerance dataset, (**) Ids for KEGG pathways in which tolerance-related genes were found in, (***) total number of pathways found for each gene. These molecular interactions are all based on known literature and details of these interactions can be seen in the specified KEGG pathways

4.4 Discussion

The motivation behind the development of Genome Interaction Analyser (GIA) is to focus on streamlining the process of gene expression data analysis with a special emphasis towards systems biology. Since the completion of the human genome project in the year 2000, vast amounts of data ranging from promoter sites of various organisms to identification of transcription factor binding sites, protein-protein interactions and pathway data have been generated. However, there is currently a lack of integration of these data in the form of user-interactive software.

The focus of our software was to give priority to motif prediction for transcription factor binding followed by exploring biological pathways to validate predictions made by GIA. The same holds true for our software's Protein-protein interaction capability in which the protein interactions found are mapped to biological pathways to verify the interactions. Making such software available is a necessity especially for biologists conducting microarray experiments because whilst gene expression measurements represent the quantity of mRNA expression, they cannot solely be used to determine the molecular causality of gene expression. Consequently, GIA aims to extract biological meaning from gene expression data by exploiting it in the context of gene-transcription factor binding, Protein-protein interactions and molecular pathways.

By putting our software to use, we were able to explore mechanisms underlying T-cell specific tolerance. The process of immune tolerance is highly complex and may involve the concerted action of several key transcription factors, which interact with specific genes at the protein-DNA level as well as at the protein-protein level to induce the state of tolerance in B and T cells. The Early Growth Response gene (EGR-2) is one of such transcription factors that has recently been characterised and although it has been extensively studied in the context of the nervous system, its exact role in the immune system has not been clearly described. However, recent studies have shown that EGR-2 is a likely candidate to play a role in the induction of T-cell anergy/immune tolerance (Safford *et al.*, 2005; Warner *et al.*, 1999) and has been found to be up-regulated in tolerised T-cells. Further studies have shown the over-expression of EGR-2 in microarray experiments investigating mechanisms underlying T-cell anergy, hence indicating a negative regulatory effect of EGR-2 towards T-cell activation (Anderson *et al.*, 2006). Interestingly following promoter analysis, our software identified the p21cip1 and p27kip1 gene promoters to contain conserved binding site regions for Egr-2. More specifically, the promoter regions of both of these genes were found to contain the **GAGGGGGCG** and **GGGGAGGCG** binding sites respectively. Both p21cip1 and p27kip1 were highly up regulated in our microarray tolerance dataset (Table 4.1). Furthermore, results from gene shift analysis have specifically confirmed these interactions, suggesting a possible mechanism via which Egr-2 regulates immune tolerance (manuscript in preparation).

Zfp385 (a zinc finger protein) and AXIN1 (also highly upregulated in our tolerance data (Table 4.1)) were both found to contain the **CCGCCCCCGC** binding site for EGR-2 within their promoter regions. The role of AXIN1 is known to be involved in the negative regulation of the Wnt signalling pathway, and has also been attributed to the induction of apoptosis (Satoh *et al.*, 2000). Pathway analysis from GIA revealed the identification of AXIN1 in the cell cycle. This leads to the inference that EGR-2 may interact with AXIN1's promoter region to induce tolerance since both molecules are known to act as negative regulators of T-cell proliferation. However, studies are yet to confirm this interaction.

STAT3, another transcription factor, plays a critical role in the induction of T-cell tolerance in CD4+ T-cells and antigen presenting cells (APC's) devoid of this transcription factor have known to effectively break antigen specific T-cell anergy in vivo, implicating its role as a negative regulator of T-cell activation (Cheng *et al.*, 2003). Studies have closely analysed the promoter site for interleukin-10, a cytokine known to downregulate the immune response, and identified a module consisting of an IFN regulatory factor 1 (Irf-1) binding site and a Stat3 binding site (Ziegler-Heitbrock *et al.*, 2003). Based on these studies, it has been identified that the interleukin-10 gene binds to the Stat3 transcription factor and is consequently up-regulated by Stat3, ultimately playing a role in tolerance (Cheng *et al.*, 2003; Ziegler-Heitbrock *et al.*, 2003). From the imported tolerance data, GIA identified the Ifnar1 interferon (alpha and beta) receptor 1 gene, which was highly expressed in our tolerance data and showed that it contains the TTCCGGAA binding site for the Stat3 transcription factor, which has been confirmed by Pfeffer's research group (Pfeffer *et al.*, 1997), reporting that Stat3 is able to bind to this receptor. Pathway analysis from GIA revealed the identification of Ifnar1 in the Jak-Stat pathway. This suggests a potential target for further investigation since Stat3 signalling has been reported to play a critical role in immune tolerance (Benkhart *et al.*, 2000).

There are several bioinformatics software packages that specialise in the high throughput analysis of gene expression data, such as Expander (Shamir *et al.*, 2005), INCLUSive (Thijs *et al.*, 2002), Genesis (Sturn *et al.*, 2002), CONFAC (Karanam *et al.*, 2004), GeneACT (Cheung *et al.*, 2006) and GEPAS (Herrero *et al*, 2003), which we tested with our in-house immune tolerance gene expression data. However, GIA has some advantages over such tools. Conceptually, there is presently no software available that integrates 1) the discovery of transcription factor specific *Cis* regulatory binding sites within promoter regions of genes, 2) the identification of protein-protein interactions and 3) mapping/overlaying of genes/protein to an up-to-date repository of molecular pathways all at once for high throughput data analysis (Table 4.4). In addition, the binding sites underlying the Gene-Protein interaction function of GIA represent evolutionarily conserved genomic regions, which are used by our software to highlight potential interactions between transcription factors and genes from the user's expression data without the use of clustering algorithms, which tend to generate arbitrary clusters of genes prior to carrying out functional/promoter analysis. Furthermore, GIA's pathway maps function is based on an efficient algorithm, which was programmed to connect to the API interface of KEGG in a live fashion. This algorithm has been designed to map several different gene identifiers to KEGG pathways (GenBank, Entrez, UniGene and NCBI), hence maximising the output for pathway analysis.

Table 4.4 A comparison of GIA with other software tools

Function	GIA	EXPANDER	INCLUSIVE	Pathway Studio	MAPPFinder	BIND	MicroCore	KEGG	BioCarta	GeneACT
Suitable for high throughput data analysis	YES	YES	YES	YES	YES	NO	YES	NO	NO	YES
Carries out Promoter Analysis/TF Assignment	YES	YES	YES	NO	NO	NO	NO	NO	NO	YES
Searches for transcription factors in users data	YES	NO	NO	NO	NO	NO	NO	NO	NO	NO
Protein-Protein Interaction analysis	YES	NO	NO	YES	YES	YES	YES	NO	NO	NO
Construction of pathway maps	YES	NO	NO	YES	YES	NO	YES	YES	YES	NO
Reveals transcription factor binding sites	YES	NO	NO	NO	NO	NO	NO	NO	NO	YES
Mapping gene expression data to pathway maps	YES	NO	NO	YES	YES	NO	NO	NO	NO	NO
Basis for	Motif	Gene	Gene		Functional					Motif

Prediction	Discovery	Expression Clustering	Expression Clustering	N/A	Enrichment using GO	N/A	N/A	N/A	N/A	Discovery
User interactive software (S) or Database (D)	S	S	S	S	S	D	S	S	S	S
Integrative capability **(*)**	YES	NO	NO	NO	NO	NO	NO	NO	NO	NO

(*) This criterion is used to identify software tools that have the capability to integrate transcription factor, protein-protein and pathway analysis all in a single package.

4.5 Conclusion

GIA is a powerful systems biology software package for microarray data analysis, which offers specific functions to its users such as simultaneous promoter analysis of genes within expression data in order to assign genes to specific transcription factors, identification of protein-protein interactions and finally, mapping of several hundreds or thousands of genes to all KEGG biological pathways known to date. We believe that such an overall package is extremely useful for biologists in general, but more specifically for microarray data analysts who are in need of quick computational solutions to complex biological problems.

4.6 Future Work

Chapter 3 addressed the importance of using a hypergeometric model such as the Fisher's exact test to generate P values in order to discern transcription factor binding sites that do not occur in the user's gene list by chance alone. Through the implementation of this transcription factor binding motif enrichment analysis method in BiSAn, we were able to demonstrate the importance of computing a P value for each binding motif. The very same approach can be implemented in GIA, and this is something that will be addressed as a future work. The methodology will entail generating a 2 x 2 contingency table for each of the 7000 unique transcription factor binding sites that belong to 1617 unique transcription factors, followed by implementing the Fishers exact test to generate a P value for each binding site. This will provide a medium for biologists to carry out binding site enrichment analysis for their gene expression datasets, which will narrow down their focus to include those binding sites that generate statistically significant P values.

Chapter 5

5

MicroPath: A software pipeline for the comparison of multiple gene expression studies to identify cellular transcriptional states

5.1 Introduction

The fundamental virtue of microarray technology lies in its ability to provide a global snapshot of the cellular state in the context of any given biological condition. This has spawned opportunities for biologists to simultaneously quantify mRNA transcript levels of entire genomes concurrently in order to observe specific transcriptome states. Although each cell of an organism contains an exact copy of its genome, the expression patterns of genes can vary due to different biological conditions, giving rise to different transcriptome states. However, the immensity of raw biological data generated from microarray experiments in the form of thousands of gene expression data points poses a challenge for biologists to extract such biological meaning from these overwhelming volumes of data. Furthermore, even if such data is statistically analysed to yield a set of differentially expressed genes, there will still be a need to integrate these sets of genes with external knowledge banks in order to make valid biological inferences. These challenges become increasingly difficult when considering the cross comparisons of multiple biologically related gene expression datasets.

Because high throughput technologies such as microarrays have rapidly gained popularity at a global scale due to the prospect of quantifying gene expression in a high throughput fashion and subsequently identifying previously unknown transcriptome states, gene expression data pertaining to various different biological questions are being rapidly generated by scientists worldwide. Such data sets are now readily accessible through public repositories such as ArrayExpress (Sarkans *et al.*, 2005) and the Gene Expression Omnibus (GEO) (Barrett *et al.*, 2006). This has motivated biologists to utilise these sets of data in an attempt to investigate common regulatory signatures that can be potentially found across multiple experiments sharing a similar biological theme. One of the most common methods of comparison is based on the assumption that genes across different biological conditions having similar expression patterns are likely to be involved in the same biological process (Rhodes *et al.*, 2004) and hence, may share the same regulatory signatures. Using this method of comparison, which is one of the most successful methods to date, coupled with the availability of publicly accessible gene expression data repositories, biologists now have the opportunity to answer complex biological questions pertaining to biological phenomena underlying various different disease states. However, there is currently a lack of software tools that have the potential to maximise the benefits that can be derived from the cross comparison of multiple gene expression datasets.

Because signals pertinent to transcriptome states tend to be diluted over entire datasets, it is imperative that specialised software tools are developed that cater for the need of extracting such information buried within masses of gene expression data points. There are currently few applications such as MiCoViTo (Lelandais *et al.*, 2004) that specialise in the analysis of transcriptome states from expression datasets by using a gene-centric approach. However, this is only relevant to the yeast genome and hence, cannot be applied to data generated from the use of other organisms. Furthermore, there are several interfaces and applications such as KEGG (Kanehisa *et al.*, 2004), GenMapp (Salomonis *et al.*, 2007), Reactome (Vastrik *et al.*, 2007) and Ingenuity Pathway Analysis (www.ingenuity.com), which allow biologists to analyse their expression data at the cellular level. In addition, data relevant to transcription factor binding sites are stored in database applications such as TRANSFAC (Wingender *et al.*, 1996), Jaspar (Sandelin, *et al.*, 2004) and the Object-Oriented Transcription Factor Database (TFD) (Ghosh, 1998). With so much wealth of data available, it would be highly fruitful to integrate these external sources of knowledge in the milieu of multiple gene expression data analysis.

In this chapter we introduce a novel bioinformatics software pipeline called MicroPath, which specializes in the cross comparison of multiple gene expression datasets and attempts to identify common regulatory signatures from the standpoint of molecular pathway analysis. When one scrutinizes current literature relevant to automated solutions of gene expression analysis, it becomes apparent that there is an increasing demand for software applications that offer an efficient pipeline to the analysis of multiple gene expression profiles. Although current meta-analyses studies have been conducted with the purpose of employing statistical techniques to compare cDNA and affymetrix gene expression profiles (Ghosh *et al.*, 2003; Rhodes *et al.*, 2002, 2004; Wang *et al.*, 2004), it cannot be denied that there is a mounting need for this process to be automated. Nevertheless, various approaches/algorithms of statistical nature have already been implemented with the purpose of identifying the most relevant pathways in a given experiment (Draghici *et al.*, 2007; Stelling *et al.*, 2004; Joshi-Tope *et al.*, 2005) together with methods such as Gene Set Enrichment Analysis (GSEA), which ranks genes based on the correlations between their expressions and observed phenotypes in the context of biological pathway discoveries (Subramanian *et al.*, 2005). There are also tools available that functionally annotate gene expression data (Khalid *et al.*, 2006a, 2006b). Albeit, it remains infeasible for biologists to cross compare several expression profiles without an automated solution, and hence they are faced with the labour-intensive task of employing manual methods to carry out their comparisons. MicroPath uses the meta-analytic standard and has been specifically developed to: compare several significantly expressed sets of genes in order to find the intersection of common genes using both number crunching methods as well as the classical permutation and combination principle, extract putative regulatory signatures using statistical and graph-based approaches and finally, mapping these sub-sets of co-expressed genes to molecular pathways all in the form of a high throughput pipeline.

5.2 Implementation

The front-end of MicroPath was developed in Visual Basic.Net and Perl, and the database back-end was developed in MySQL. Upon analysing the users input files (gene expression profiles), processed data is displayed intuitively on the graphical user interface, which is equipped with various interactive objects such as charting facilities, buttons, drop-down menus and user input/output dialogues. The interface is also equipped with a function to export processed data into Microsoft excel for further scrutiny and use.

5.2.1 System Architecture

MicroPath carries out meta-profiling of multiple gene expression datasets using two different approaches. Firstly, the intersection of common genes is identified across *n* number of expression profiles, which is then plotted graphically using a simple number crunching exercise. The second approach applies to a situation where an attempt to identify common genes across *n* number of expression profiles using the aforementioned approach fails due to the absence of common genes across all datasets (this situation is especially common when a large number of expression profiles are compared, which reduces the probability of finding a common gene amongst them). Consequently, MicroPath applies the permutations and combinations mathematical principle to solve this problem (refer to *implementation of meta-analysis strategy* below for details). Once the intersection of a set of common genes has been identified and subsequently displayed on the interface (using either of the above methods), the next stage in the analysis is to extract patterns from the intersection in order to identify common genes that are being expressed in accordance with the biological question. MicroPath offers a semi-automated graph-based approach to achieve this as well as classical statistics to identify the overall correlation of gene expression. Finally, co-expressed genes (common genes that are expressed in accordance to the relevant biological question) are mapped to all molecular pathways known to date in order to reveal their molecular dependencies (refer to Figure 5.1 for the complete system architecture).

5.2.2 Implementation of Meta-analysis Strategy

In theory, an intersection of a sub-set of common genes across multiple gene expression profiles should be easily attainable using simple number crunching methods of comparison. In practice, this is not always the case since the likelihood of identifying genes sharing common accession identifiers decreases as the number of profiles to compare increases. This inverse relationship makes sense both mathematically and biologically. From a biological perspective, regulatory signatures tend to be diluted over entire datasets and as a result, only a proportion of the total number of profiles to compare may actually share common genes. In such a scenario, using a simple method of comparison would break down at some point and no common genes would be reported to the user, although common genes may be present within $n - 1$ expression profiles.

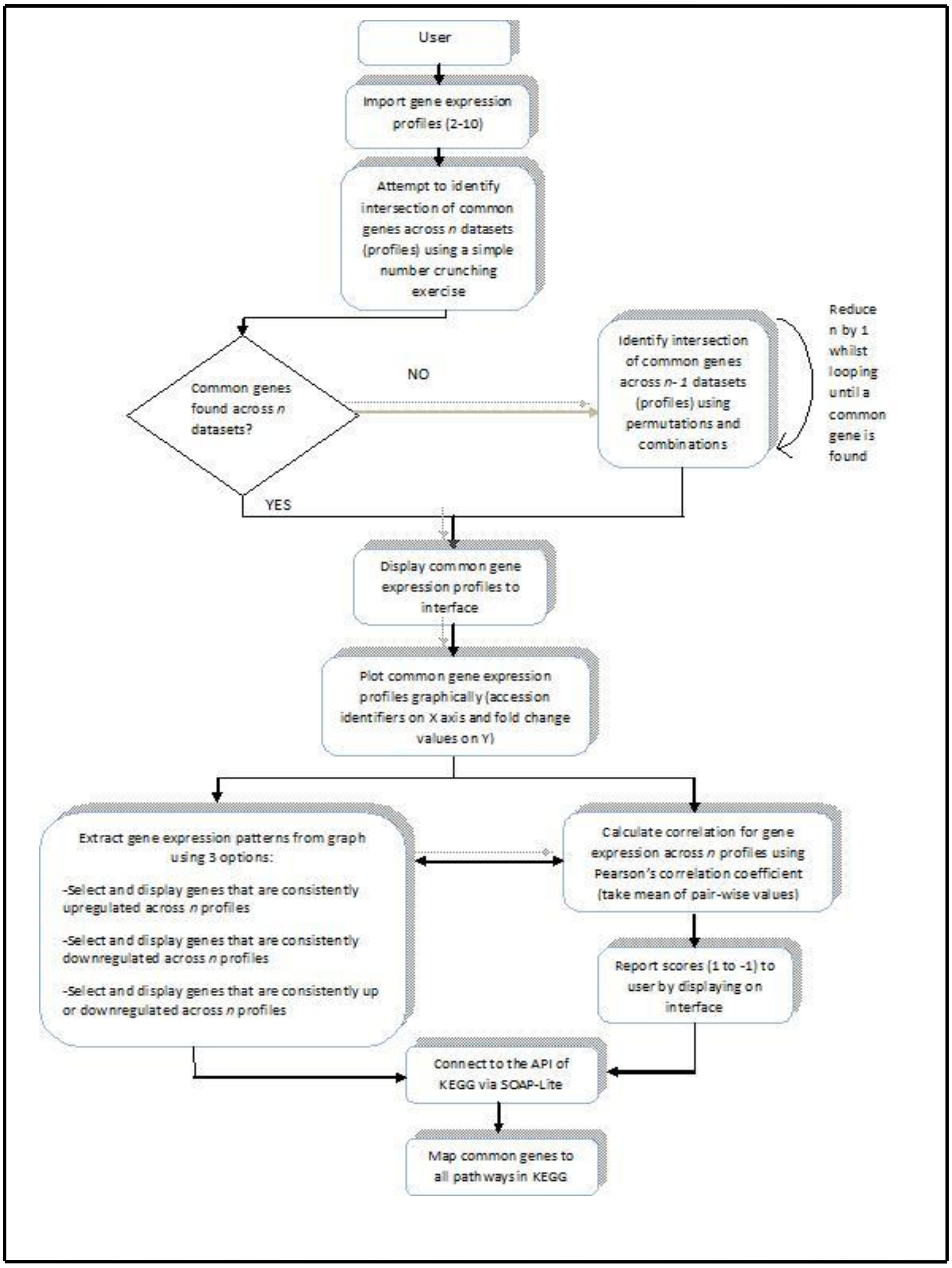

User
Import gene expression profiles (2-10)
Attempt to identify intersection of common genes across n datasets (profiles) using a simple number crunching exercise
Common genes found across n datasets?
NO
YES
Identify intersection of common genes across n- 1 datasets (profiles) using permutations and combinations
Reduce n by 1 whilst looping until a common gene is found
Display common gene expression profiles to interface
Plot common gene expression profiles graphically (accession identifiers on X axis and fold change values on Y)
Extract gene expression patterns from graph using 3 options:
-Select and display genes that are consistently upregulated across n profiles
-Select and display genes that are consistently downregulated across n profiles
-Select and display genes that are consistently up or downregulated across n profiles
Calculate correlation for gene expression across n profiles using Pearson's correlation coefficient (take mean of pair-wise values)
Report scores (1 to -1) to user by displaying on interface
Connect to the API of KEGG via SOAP-Lite
Map common genes to all pathways in KEGG

Figure 5.1 Functions of MicroPath. Users are prompted to import up to 10 gene expression profiles, which are then compared using a direct comparison method. If this method yields zero common genes, MicroPath automatically attempts to identify an intersection of common genes by reducing the

search space to *n – 1* datasets using permutations and combinations. This process is continued until at least 1 common gene is reported. Following this, users are provided with a function to search for expression patterns graphically and gene expression correlations are calculated statistically using the pearson's correlation coefficient algorithm. Finally, co-expressed genes are mapped to all molecular pathways of KEGG in a high throughput fashion by automatically accessing its API via SOAP-Lite.

To prevent potentially interesting biological findings to be hampered at this point in the analysis, we have applied the principle of mathematical combinations to the comparison of multiple gene expression profiles. All possible combinations of comparing *n* number of datasets with each other are firstly computed using the combination equation:

$$^{n}C_{r} = \frac{n!}{r!(n-r)!}$$

Where *n*! Is the factorial of the total number of datasets *n,* and r! Is the factorial of the selected number of datasets to compare when comparing *n* datasets results in zero common genes, r.

This generates the total number of permutations of comparing datasets (Cr) for given values of *n* (total number of datasets imported by user) and *r* (number of intended datasets used to search for common genes when zero common genes are reported across *n* datasets) (Table 5.1).

Table 5.1 Multiple gene expression profile search strategy generated from applying the principle of permutations and combinations.

Total number of expression datasets **(*n*)**	Number of intended expression datasets to compare when comparing *n* datasets yields no results ***(r)***	***n - r***	Total number of combinations of *r* **(*Cr*)**
10	9	1	10
10	8	2	45
10	7	3	120
10	6	4	210
10	5	5	252
10	4	6	210
10	3	7	120
10	2	8	45
9	8	1	9
9	7	2	36
9	6	3	84
9	5	4	126
9	4	5	126
9	3	6	84
9	2	7	36
8	7	1	8
8	6	2	28
8	5	3	56
8	4	4	70
8	3	5	56
8	2	6	28
7	6	1	7
7	5	2	21
7	4	3	35
7	3	4	35
7	2	5	21
6	5	1	6
6	4	2	15
6	3	3	20
6	2	4	15
5	4	1	5
5	3	2	10
5	2	3	10
4	3	1	4
4	2	2	6
3	2	1	3

The first column represents the total number of expression datasets, *n*, that users may import (this is the search space). The second column represents, *r*, the number of expression datasets to compare if zero common genes are reported to be matched across *n* datasets. The final column represents the total number of mathematical combinations possible for each given value of *n* and *r*.

These combinations of datasets (*Cr*) are then used as a criterion to search for common genes across *r* number of gene expression profiles when comparing *n* number of datasets fail to yield any common genes. However in this scenario, *n* number of datasets is still used as the search space from which all possible combinations (*Cr*) of *r* datasets are compared to each other in order to increase the probability of finding a common gene. Once common genes have been identified using this method, MicroPath will report the results to the interface.

5.2.3 Raw data analysis

This function was specifically developed to facilitate the cross comparison of multiple gene expression profiles containing repeated genes. We implemented the Student's t-test in order to firstly identify common genes across two datasets sharing an identical accession number. The subsequent step was designed to take each common gene (existing in both datasets) and use its repeated gene expression data points to compute the actual difference between their means in relation to the standard deviation. This pair-wise method of comparison was implemented to handle all possible comparisons for a maximum of 10 datasets and t-values for each common gene were calculated as follows:

$$t = \frac{\overline{X}_1 - \overline{X}_2}{s_{\overline{X}_1 - \overline{X}_2}} \qquad \text{Where} \qquad s_{\overline{X}_1 - \overline{X}_2} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}$$

Where s^2 is the unbias estimator of the sample variance, n = number of replicates, 1 = sample data one and 2 = sample data two. Based on the recorded t-values, the degrees of freedom were computed using the Welch-Satterthwaite equation, which were in turn used to obtain p-values:

$$\text{D.F.} = \frac{(s_1^2/n_1 + s_2^2/n_2)^2}{(s_1^2/n_1)^2/(n_1-1) + (s_2^2/n_2)^2/(n_2-1)}.$$

5.2.4 Extracting Gene Expression Patterns Graphically and Statistically

Following the identification of common genes across *n* datasets using either of the methods described earlier, the next stage in the analysis is to generate a graphical representation of this expression data from which biologically meaningful patterns can be extracted. Because signals pertaining to transcriptome states tend to be diluted over entire profiles, a specific criterion is required to narrow down the common genes of interest to include only those genes that are consistently regulated according to the biological question. The assumption we have made is that any given common gene across *n* datasets can exhibit one of three specific behaviours. It can either be consistently upregulated across all datasets, downregulated across all datasets and up or downregulated across all datasets. Based on the nature of the specific biological question, users can select the appropriate pattern from the options, which will result in a graphical display of those genes which satisfy the search criteria. Together with this faculty to graphically extract patterns for individual gene expression data points, MicroPath also implements the pearsons correlation coefficient statistical test in order to extract a global gene expression pattern existing between common genes relevant to two individual expression profiles. The correlations are calculated in a pair-wise manner until each expression data has been statistically compared to all other datasets within *n*, according to the pearsons correlation coefficient equation:

$$r = \frac{\sum XY - \frac{(\sum X)(\sum Y)}{n}}{\sqrt{\left(\sum X^2 - \frac{(\sum X)^2}{n}\right)\left(\sum Y^2 - \frac{(\sum Y)^2}{n}\right)}}$$

Each pair-wise score is then finally averaged in order to provide a global measure of correlation existing between *n* expression profiles. Scores are reported from -1 (perfect negative correlation) to +1 (perfect positive correlation).

5.2.5 High Throughput Molecular Pathway Analysis

To decipher molecular mechanisms fundamental to the researcher's biological question, it is necessary to map common gene expression profiles of co-expressed genes to molecular pathways. This is because biological pathways reveal molecular dependencies that exist between genes by illustrating how they collaborate with one another when they participate in specific biological functions. Furthermore, pathways reveal various signalling cascades that play imperative roles in dictating these gene associations. In light of this, we have implemented Micropath to access the Application Programming Interface (API) of the molecular pathway database belonging to KEGG (Kanehisa *et al.*, 2004) using SOAP-Lite in order to dynamically interact with the static pathway maps. Perl scripts were written for MicroPath to specifically 1) search for user's co-expressed genes in all biological pathways, 2) highlight genes on to pathways, and 3) return the results of the search to Micropath's interface (i.e. URL's of colour coded pathway maps) (Figure 5.2). Once MicroPath has searched for all of the user's co-expressed genes in all of the molecular pathways, the URL of each pathway is displayed on the sub-interface. Clicking on these links will generate the specific KEGG pathway in HTML on which users co-expressed genes will be highlighted.

In order to avoid redundancy issues, the URL for each pathway will highlight all co-expressed genes that participate in a given pathway. Also, to help users identify biologically meaningful pathways relevant to their specific biological question, MicroPath will calculate the number of genes identified in a given pathway and 1) express this as a percentage in relation to the total number of common genes from the intersection and 2) express this as a percentage in relation to the total number of genes belonging to that pathway.

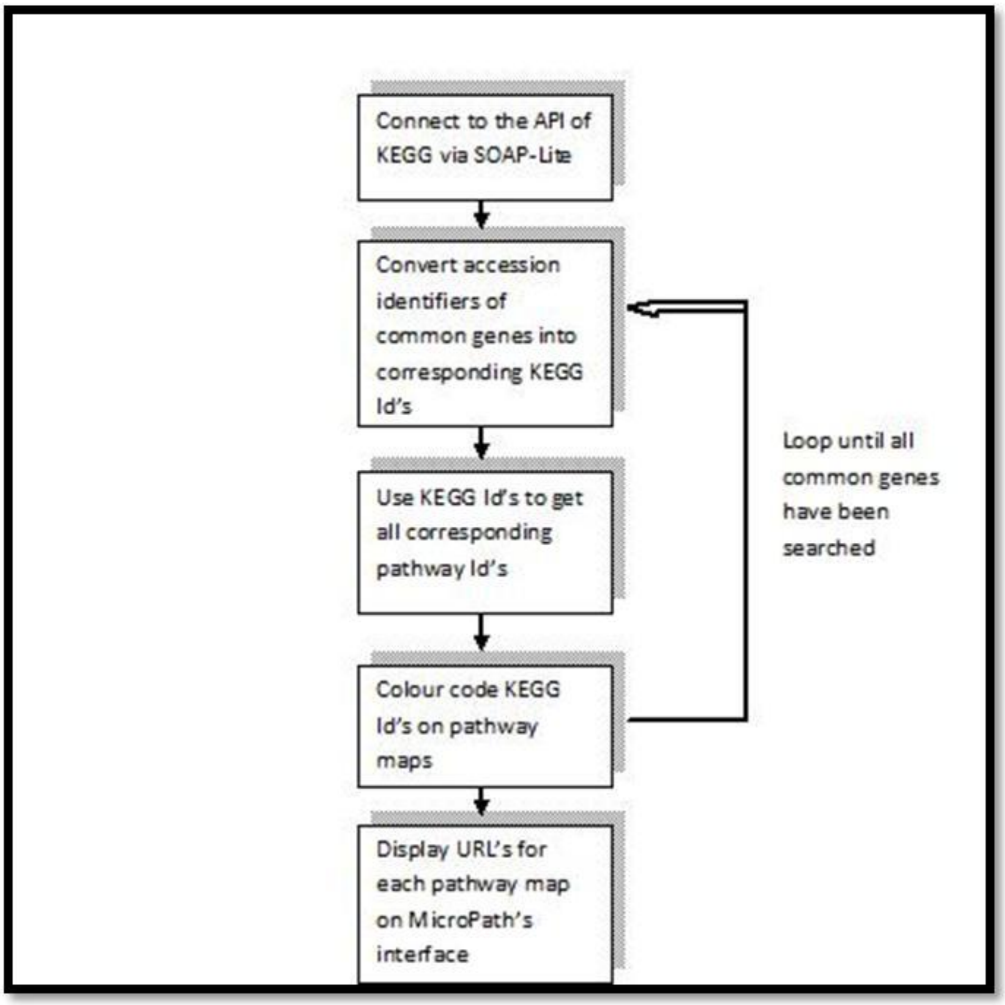

Connect to the API of KEGG via SOAP-Lite
Convert accession identifiers of common genes into corresponding KEGG Id's
Use KEGG Id's to get all corresponding pathway Id's
Colour code KEGG Id's on pathway maps
Display URL's for each pathway map on MicroPath's interface
Loop until all common genes have been searched

Figure 5.2: Flow diagram of how MicroPath carries out high throughput molecular pathway analysis by connecting to the API of KEGG.

5.2.6 Generating and Processing Gene Expression Datasets

Gene expression datasets used for the purpose of this work were generated from our in-house microarray experiments as well as published datasets, where the fold change approach was used to select a set of differentially expressed genes from pre-processed data. Matchminer (Bussey *et al.*, 2003) and the Synergizer (Berriz *et al.*, 2008) tools were used to convert gene Hugo identifiers and long names into Genbank accession Id's in order to ensure that the gene identifiers were of the same type across all datasets prior to comparison. Raw expression data was generated, filtered and normalised using GenePix pro 4.1 (www.axon.com) and Acuity 4.0 (www.moleculardevices.com) software. Although we used cDNA microarray data for the purpose of demonstrating MicroPath's capabilities, other data types generated from different platforms such as affymetrix can also be analysed provided Genbank accession identifiers are used to represent the genes.

5.3 Results and Discussion

Regardless of the biological question, a typical microarray experiment almost always results in the generation of a set of differentially expressed genes, which represents genes of most importance to the biologist. Therefore, by carrying out several biologically related microarray experiments, several sets of differentially expressed genes would be generated, which would need to be compared and mined efficiently in order to help answer the biological questions asked by the investigators from different research laboratories around the world. Employing manual methods of comparison in this situation would be very inefficient and infeasible. In light of this, to demonstrate the benefits that can be derived from analysing multiple gene expression profiles using MicroPath, we employed datasets generated from our in-house microarray experiments as well as published data. The biological question related to these studies focussed on unravelling the underlying molecular mechanisms dictating immune tolerance by analysing the role of Egr-2 in implicating T-cell tolerance. Although the Early Growth Response gene (Egr-2) has been recently characterised as a candidate tolerance-inducing transcription factor, which interacts with specific genes in order to induce the state of T-cell tolerance (Safford *et al*., 2005; Warner *et al*., 1999), the possibility of further putative unknown target genes exists that may be vital to the mechanism of tolerance. Hence, the biological purpose of our experiments was to attempt to identify such potentially important genes via the comparison of biologically related expression datasets using MicroPath.

Data consisting of a set of differentially expressed genes generated from the comparison of tolerance Vs activated mice CD4+ T cells was obtained from the ArrayExpress website (accession number: e-mexp-283). The first in-house experiment aimed to generate differentially expressed genes from the comparison of an un-stimulated T cell line from which the Egr-2 gene had been knocked out and a wild type un-stimulated cell line. The second in-house experiment focussed on the comparison between an Egr-2 knock-out T cell line activated with CD3/CD28 for 6 hours and a wild type cell line also activated with CD3/CD28 for 6 hours. Results generated from these experiments were then compared with the aforementioned published tolerance data using MicroPath in order to understand the molecular mechanisms controlling immune tolerance.

5.3.1 Comparison of Gene Expression Profiles related to Immune Tolerance

The first step in the analysis was to subject the above-mentioned expression profiles to MicroPath in order to identify genes amongst them that had the same accession identifiers. Having done this, MicroPath identified 31 differentially expressed genes that were common to all three expression datasets and generated a graph to delineate their expression values (Table 5.2, Figure 5.3). A simple number crunching exercise was used to perform this task since its use generated a reasonable number of common genes, which did not warrant the use of permutations and combinations to perform the search. The next step was to use these 31 differentially expressed genes as a search space to determine those genes that have the potential to be co-expressed. In order to do this, we employed MicroPath's graphical utility to extract gene expression patterns, which led to the identification of 6/31 genes that were found to be upregulated in tolerance Vs activated CD4+ T-cells and downregulated in both p-KOA0 Vs WTA0 and p-KOA6 Vs WTA6 datasets (Table 5.2). The remaining 25 common differentially expressed genes were found to be highly and lowly expressed in tolerance and knock-out datasets respectively. Statistical analysis revealed an overall pearson's correlation score of 0.109 from the pair-wise comparison of tolerance data with p-KOA0 Vs WTA0 and a score of -0.123 from the comparison of tolerance with p-KOA6 Vs WTA6. Furthermore, Reverse Transcriptase PCR experiments confirmed that 15 genes from our tolerance Vs activated data were found to be highly expressed in immune tolerance and from these 15 genes, 8 were found to be common amongst all three expression profiles (Table 5.2).

Because Egr-2 has been previously characterised and found to be highly upregulated in immune tolerance, these results generated from MicroPath are biologically significant because as expected, those genes that were highly expressed in our tolerance Vs activated datasets were found to be insignificantly expressed in our p-KOA6 Vs WTA6 and p-KOA0 Vs WTA0 datasets (from which the Egr-2 gene was knocked out of the cell lines). Amongst these genes, Ap1s1, Shd, Surf6, Vil2, Lilrb4, Tbx21 and Pdcd1lg2 (Table 5.2) have been confirmed to be upregulated in the process of immune tolerance (Anderson *et al.*, 2006), all of which were found to exhibit low expression values in our knock-out expression datasets.

This consistent gene expression pattern can be seen graphically in Figure 5.3. However, from the 31 interesting common genes, 16 were not confirmed to be involved in tolerance by RT-PCR yet some of them also exhibited a coherent pattern of gene expression. For example, Ptma, Scd2, Hdac6, Pltp and Chka were all highly expressed in tolerance and conversely downregulated in both knock out datasets. There is a possibility that these genes may also be insignificantly expressed due to the absence of Egr-2. However, conducting RT-PCR for these specific genes would be required in order to confirm that their over-expression results in T-cell tolerance.

Table 5.2: Tabulated overview of gene accession ids, Hugo ids and fold change values belonging to 31 common genes identified from the comparison of tolerant Vs activated CD4+ T cells, p-KOA0 Vs WTA0 and p-KOA6 Vs WTA6 expression datasets.

Gene ID	HUGO ID	Fold Change (p-KOA0 Vs WTA0)	Fold Change (p-KOA6 Vs WTA6)	Fold Change (Tolerance Vs activated)
NM_007381	Acadl	0.371336	0.624525	6.373
NM_007457	Ap1s1 *	0.542474	0.31525	4.965
NM_007664	Cdh2	0.243646	-0.7999	1.658
NM_008205	H2-M9	-0.08048	0.116434	2.857
NM_008972	**Ptma**	**-1.31334**	**-0.46688**	**5.42**
NM_009128	**Scd2**	**-0.18816**	**-0.39366**	**4.552**
NM_009168	**Shd** *	**-0.17495**	**-0.53582**	**2.838**
NM_009298	Surf6 *	0.272072	0.126301	4.365
NM_009465	Axl	0.149539	1.475806	3.836
NM_009510	Vil2 *	-0.49824	0.319645	3.151
NM_010102	Edg6	0.313489	0.132689	1.573
NM_010413	**Hdac6**	**-0.90335**	**-0.8226**	**4.745**
NM_010548	Il10 *	3.083863	1.660739	3.521
NM_010638	Bteb1	0.024803	-0.42533	1.613
NM_011125	**Pltp**	**-0.5354**	**-0.71558**	**4.363**
NM_011620	Tnnt3	-0.61646	0.035844	1.665
NM_011696	Vdac3	-0.98084	0.191964	4.701
NM_011705	Vrk1	0.466922	-0.34601	2.032
NM_013488	Cd4	0.584494	0.420277	4.905
NM_013490	**Chka**	**-2.13728**	**-0.69458**	**5.677**
NM_013532	Lilrb4 *	0.792335	1.110898	2.111
NM_013615	Odf2	2.776384	3.004449	4.809
NM_013814	Galnt1	-0.47752	0.500297	2.246
NM_013866	Zfp385	0.118995	0.428591	1.664
NM_016772	Ech1	-0.0666	0.053081	4.284
NM_019507	Tbx21 *	0.124767	-0.32731	1.595
NM_019561	Ensa	0.778767	-0.44703	1.718
NM_019777	Ikbke	0.291602	-0.00772	1.609
NM_020027	Bat2	0.291219	-0.23966	5.091
NM_021396	Pdcd1lg2 *	1.140087	0.079182	3.921
NM_021538	Cope	0.154049	0.264541	2.035

Entries highlighted in bold represent genes that were found to be up-regulated in tolerance Vs activated CD4+ T cells and down-regulated in both p-KOA0 Vs WTA0 and p-KOA6 Vs WTA6 datasets. Entries with * represent genes that have been confirmed to be highly expressed in tolerance by RT-PCR.

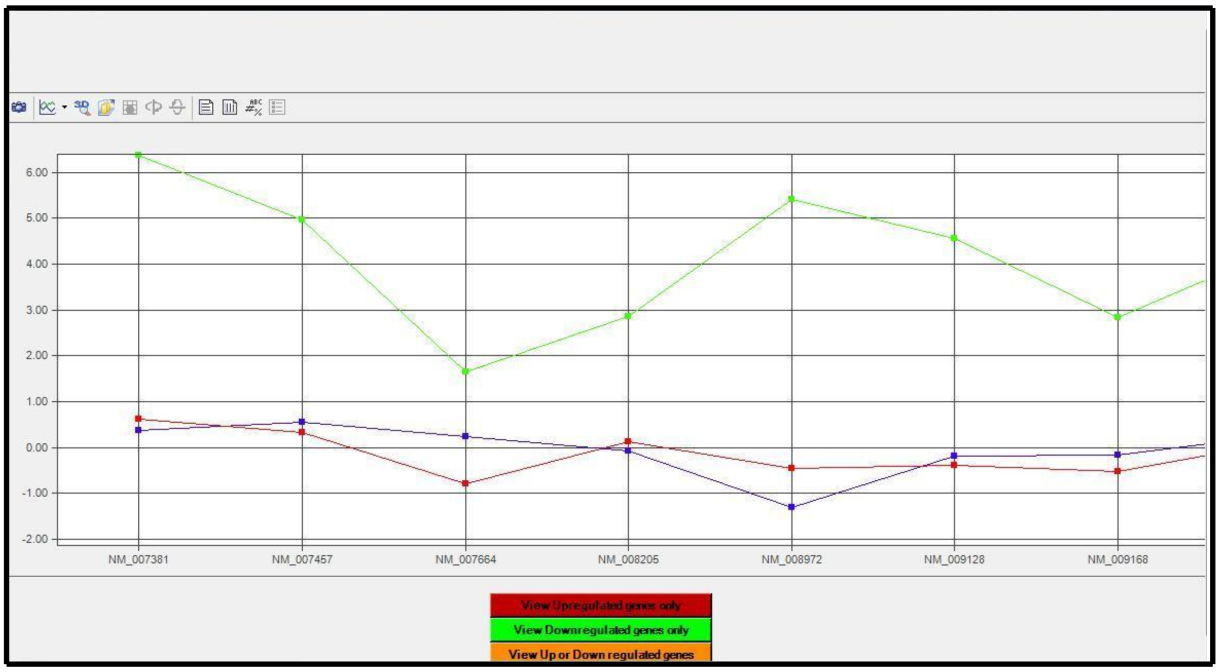


Figure 5.3: A preliminary graphical overview of common interesting genes generated from the comparison of tolerant Vs activated CD4+ T cells (green), p-KOA0 Vs WTA0 (red) and p-KOA6 Vs WTA6 (blue) expression datasets. It can be seen that genes that are highly expressed in tolerance appear to be expressed poorly in the knock-out datasets. This pattern is consistent throughout the 31 gene expression data points.

5.3.2 Deciphering gene regulatory networks of co-expressed genes via high throughput molecular pathway analysis

The final stage of the analysis entails using MicroPath's function to connect to the Application Programming Interface (API) of KEGG via SOAP-Lite in order to carry out high throughput molecular pathway analysis. Therefore, for this stage in the analysis, we used MicroPath to map 31 of our co-expressed interesting genes to KEGG pathways and from these 31 genes, 14/31 were identified in a total of 31 molecular pathways (Table 5.3). Interestingly, several of these pathways were related to the study of immunology and illustrated biological networks such as MapKinase, Jak-Stat, T-cell receptor signalling and Cytokine-cytokine interactions. More specifically, the Pdcd1lg2 gene (accession id: NM_021396) was identified in the Cell Adhesion Molecules (CAM) pathway (Table 5.3) and studies have confirmed that the over-expression of Pdcd1lg2 has resulted in consistently low levels of Interleukin-2 (IL-2) in naive CD4(+) T-cells (Kuipers *et al.*, 2006). Further studies have correlated the over-expression of this gene to the negative regulation of T-cell activation. In one particular study, PDL2 (Pdcd1lg2) deficient mice were created in order to characterise the function of this gene in T-cell activation and tolerance, and results generated from this study suggested that Antigen-presenting cells from PDL2-deficient mice were found to be more potent in activating T-cells in vitro when compared to the wild-type counterparts (Zhang *et al.*, 2006). These findings are conclusive and correlate well with the results generated from our in-house microarray experiments because using MicroPath to compare all three of our datasets followed by extracting gene expression patterns from them resulted in an important finding that Pdcd1lg2 was not only found to be over-expressed in tolerance (fold change of 3.921), but it was also under-expressed in our KOA0 Vs WTA0 and KOA6 Vs WTA6 knock-out datasets (with a fold change of 1.140 and 0.079 respectively) (Table 5.2). This particular finding is in agreement with the aforementioned studies, concluding that Pdcd1lg2 has a negative inhibitory role towards the process of T-cell activation. In addition, molecular pathway analysis of the Interleukin-10 (IL-10) gene using MicroPath identified its role in the Cytokine-cytokine interaction, Jak-STAT and T-cell receptor signalling pathways; all three of which are important immunological pathways. IL-10 is a well known cytokine, which has previously been shown to successfully induce immune tolerance in Dendritic Cells (Li *et al.*, 2007). Results generated from MicroPath revealed that IL-10 was highly expressed in our tolerance data with a fold change of 3.521, which was found to be expressed lower in our KOA0 Vs WTA0 profile (fold change: 3.084). Interestingly, following activated with CD3/CD28 for 6 hours, its expression dropped significantly to 1.66, perhaps attributable to the absence of Egr-2. Likewise, other genes from the 31 co-expressed interesting genes show similar patterns of expression and perhaps may be candidate genes for Egr-2 mediated T-cell tolerance. However, this is yet to be confirmed by publications. Finally, the pathway analysis function of MicroPath was used to calculate the percentage of genes identified in each pathway in relation to 1) the intersection of common genes and 2) the total number of genes comprising each pathway. From the results, the Cell Adhesion Molecules (CAM) pathway was particularly significant since 12.91% of the overall pathway was affected by 6.84% of genes common to all 3 expression profiles (Table 5.4).

Table 5.3: Tabulated data generated from high throughput molecular pathway analysis of co-regulated genes. 14/31 common interesting genes were identified in a total of 31 molecular pathway maps of KEGG.

GenBank Accession ID	HUGO ID	Pathway ID	Total No of pathways	GenBank Accession ID	HUGO ID	Pathway ID	Total No of pathways
NM_007381	Acadl	mmu00071 mmu00280 mmu00410 mmu00640 mmu03320	5	NM_009510	Vil2	mmu04670 mmu04810	2
NM_007664	Cdh2	mmu04514	1	NM_008205	H2-M9	mmu04514 mmu04612 mmu04940	3
NM_013488	Cd4	mmu04514 mmu04612 mmu04640 mmu04660	4	NM_013814	Galnt1	mmu00512 mmu01030	2
NM_011696	Vdac3	mmu04020	1	NM_019777	Ikbke	mmu04010 mmu04620	2
NM_011125	Pltp	mmu03320	1	NM_010102	Edg6	mmu04080	1
NM_016772	Ech1	mmu00350 mmu00362 mmu00628	3	NM_021396	Pdcd1lg2	mmu04514	1
NM_010548	Il10	mmu04060 mmu04630 mmu04660	3	NM_013652	Ccl4	mmu04060 mmu04620	2

Table 5.4: Results generated from pathway analysis showing the extent to which each pathway is affected by common genes from the intersection. The percentages reflect the proportion of common genes that contribute towards controlling the proportion of each pathway.

Pathway ID	**Pathway Name**	**GenBank Accession ID**	**Result from Analysis**
mmu00071	Fatty Acid Metabolism	NM_007381	3.26% of genes contribute 8.45% role in pathway
mmu00280	Valine, leucine and isoleucine degradation	NM_007381	3.26% of genes contribute 2.73% role in pathway
mmu00410	Beta Alanine Metabolism	NM_007381	3.26% of genes contribute 7.14% role in pathway
mmu00640	Propanoate Metabolism	NM_007381	3.26% of genes contribute 5.88% role in pathway
mmu03320	PPAR Signalling Pathway	NM_007381	3.26% of genes contribute 1.92% role in pathway
mmu04514	Cell Adhesion Molecules	NM_007664 NM_008205 NM_013488 NM_021396	12.91% of genes contribute 6.84 % role in pathway
mmu04612	Antigen Processing & Presentation	NM_013488	3.26% of genes contribute 2.44% role in pathway
mmu04640	Hematopoietic Cell Lineage	NM_013488	3.26% of genes contribute 0.76 % role in pathway
mmu04660	T Cell Receptor Signalling Pathway	NM_013488 NM_010548	6.45 % of genes contribute 3.33 % role in pathway
mmu04020	Calcium Signalling Pathway	NM_011696	3.26% of genes contribute 2.33 % role in pathway
mmu00350	Tyrosine Metabolism	NM_016772	3.26% of genes contribute 2.17 % role in pathway
mmu04060	Cytokine-cytokine receptor interaction	NM_010548 NM_013652	6.45 % of genes contribute 0.73 % role in pathway
mmu04630	JAK-STAT Signalling Pathway	NM_010548	3.26% of genes contribute 3.85 % role in pathway
mmu04670	Leukocyte Transendothelial Migration	NM_009510	3.26% of genes contribute 1.25 % role in pathway
mmu04810	Regulation of Actin Cytoskeleton	NM_009510	3.26% of genes contribute 1.47 % role in pathway
mmu04940	Type I Diabetes Mellitus	NM_008205	3.26% of genes contribute 4.35 % role in pathway
mmu00512	O-Glycan Biosynthesis	NM_013814	3.26% of genes contribute 10 % role in pathway
mmu04010	MAPK Signalling Pathway	NM_019777	3.26% of genes contribute 0.83 % role in pathway
mmu04620	Toll-Like Receptor Signalling Pathway	NM_019777 NM_013652	6.45% of genes contribute 1.32 % role in pathway
mmu04080	Neuroactive Ligand-Receptor Interaction	NM_010102	3.26% of genes contribute 1.15 % role in pathway

The fundamental strength of MicroPath stems from the implementation of a novel search strategy for the comparison of multiple gene expression profiles. Although there are a few software that cater for multiple gene expression comparison, there is currently no software that searches for common genes beyond simple number crunching methods of comparison (Table 5.5). Just because a direct comparison of a given number of datasets may not yield any common genes, it does not mean that the analysis should end here since there is a potential to identify common genes across $n - 1$ profiles. MicroPath ensures that such genes are identified, which current software would overlook. When coupled with other important functions such as pattern extraction and pathway analysis, it becomes apparent that MicroPath would offer valuable assistance to biologists wanting to decipher their high throughput data.

Table 5.5: Functional comparison of MicroPath to similar software packages and applications.

Function	**MicroPath**	**EXPANDER**	**INCLUSIVE**	**Pathway Studio**	**KEGG**	**BioCarta**	**MaXlab**
Suitable for high throughput data analysis	YES	YES	YES	YES	NO	NO	YES
Suitable for comparing multiple gene expression profiles	YES	YES	NO	YES	NO	NO	YES
Implementation of efficient algorithm to search for common genes from *n – 1* datasets	YES	NO	NO	NO	NO	NO	NO
Graphical representation of gene expression values from multiple datasets	YES	NO	NO	NO	NO	NO	YES
Pattern extraction from Graph data	YES	NO	NO	NO	NO	NO	NO
Construction of pathway maps	YES	NO	NO	YES	YES	YES	NO
Mapping gene expression data to pathway maps	YES	NO	NO	YES	NO	NO	NO
User interactive software (S) or Database (D)	S	S	S	S	D	D	S

5.4 Conclusion

In this chapter, we have illustrated the potential benefits that can be derived from using MicroPath for the analysis of multiple gene expression profiles. Each function of the software has been developed to streamline the overall analysis pipeline, providing users with a walkthrough of how their data is biologically deciphered. Here, we have applied to our software, microarray datasets generated from different laboratories pertaining to the molecular mechanisms underlying immune tolerance. However, MicroPath is capable of analysing data for any given biological question, whether the datasets are taken from public repositories such as ArrayExpress or generated from in-house microarray experiments. We believe that its faculty to use both number crunching and permutations and combinations as the search strategy to identify the intersection of common genes, coupled with its function to extract gene expression patterns graphically and statistically makes this a attractive software for biologists to use. Finally, its ability to carry out live streaming of mapping genes to biological pathways makes it a useful tool for the automation of multiple gene expression analysis.

5.5 Future Work

Chapters 3 and 4 discussed the application of generating contingency tables and subsequently using them for the Fisher's exact test in order to generate P values in the context of transcription factor binding motif enrichment analysis. However, this approach can also be employed for the purpose of identifying molecular pathways that are enriched with genes from the user's expression data. The principle is the same as TFBM enrichment analysis, except that for MicroPath, a contingency table will be generated for each KEGG pathway that contains common genes from the user's datasets. Following this, the Fishers exact test will be applied to generate a P value for each pathway to discern KEGG pathways that are significantly enriched with common genes from the user's datasets in relation to the entire genome.

For this future work, a contingency table for each KEGG pathway will first need to be derived as follows:

	Chosen	Not Chosen	Total
TFBM	***a***	***b***	***a + b***
Absent	***c***	***d***	***c + d***
Totals	***a + c***	***b + d***	***n***

Where: a = No of common genes in user's gene lists found in KEGG pathway (chosen from gene list)

b = No of genes in the genome found in KEGG pathway minus a (Not chosen from gene list)

c = No of common genes in user's gene lists not found in KEGG pathway (chosen from gene list)

d = Total no of genes in genome – (a + b + c)

n = Sum of each total (Grand total)

These contingency tables will reflect the degree of enrichment of a given KEGG pathway in the user's sets of differentially expressed genes (common genes) relative to the entire genome. Once contingency tables have been generated for all consensus binding sites, Micropath will then apply the Fisher's exact test to compute P values for each KEGG pathway from their contingency tables using the following hypergeometric distribution:

$$p = \binom{a+b}{a}\binom{c+d}{c} \Big/ \binom{n}{a+c} = \frac{(a+b)!(c+d)!(a+c)!(b+d)!}{n!a!b!c!d!}$$

Where $\binom{n}{k}$ = binomial coefficient and ! = factorial operator

Chapter 6

6

Conclusions and Future Work

6.1 Introduction

The reductionist approach to biological research has proven to be imperative in developing our basic but necessary understanding of living systems. Studying and subsequently identifying the individual components (such as genes, proteins and metabolites) that regulate specific physiological phenomenon (for instance, metabolic activity, response to external stimuli etc) has no doubt proven to be an effective strategy in elucidating key molecular components of living systems, leading to a variety of important applications in agriculture and medicine (Aggarwal & Lee, 2003). However, it has become clear now that in order to fully understand the behaviour of biological systems, we have to look beyond isolated parts of an organism. In other words, scrutinizing the genome or the proteome autonomously does not warrant a complete understanding of the biological system until and unless they are examined in the context of the organism as a whole unit. This is the point where systems biology becomes an extremely important consideration.

It is precisely the aforementioned reasons that have motivated systems biology, which has resulted in a paradigm shift from a reductionist approach to an integrative one. This is indeed an effective strategy since it encourages a scrutiny of the structure and dynamics of cellular function in the context of the organism as a whole rather than examining the characteristics of isolated parts of the cell or organism (Kitano, 2002). The importance of systems biology becomes apparent when we examine the current situation with drug discovery. Despite the fact that an astronomical amount of investment has taken place over the past 20 years towards screening technologies and genomics, the truth remains that the costs associated with new drug discovery continue to rise while approval rates fall. This is largely attributable to the fact that merely knowing a target is not sufficient to warrant an understanding of what the target does, let alone knowing the effects of a chemical inhibitor in diverse disease settings (Butcher, 2004). It is hence not surprising that approval rates are continuously falling in the milieu of drug development programs. After all, before we can truly understand the physiological implications of a particular drug, we must be in a position to understand how the target biological system functions as a synchronized unit. This is why bioinformatics is becoming a necessity in the study of biological sciences due to the fact that the latter has become highly reliant on informatics, mathematics, computer algorithms, software development and statistics. This especially holds true when we consider the marriage between high throughput technologies such as microarrays with systems biology.

On one hand we have a high throughput technology offering a platform to measure mRNA transcript levels for entire genomes simultaneously in a cost effective and efficient fashion, and on the other hand, we have a interdisciplinary science specializing in the application of mathematics, computer science and biology with the fundamental purpose of unravelling the underlying functional dynamics of the cell in the perspective of the entire organism. The potential strength of this collaboration is based on a common goal; *to understand cellular and molecular function in the context of the organism as a whole.* In theory, the possibilities are endless. In practice however, systems biology is still at its infancy and consequently, there is much to be desired.

It is this exact need to better apply systems biology to solve biological problems that motivated the research presented in this thesis. We began systematically by firstly developing BiSAn (Chapter 3), which when applied to a set of microarray data proved to show an important capability to compute binding affinities of several Transcription Factors in relation to promoter regions belonging to the genes of interest. The second step was to move towards an integrative approach, which motivated the development of Genome Interactions Analyzer (GIA) (Chapter 4). By integrating the analysis of transcription factors, Protein-protein interactions and molecular pathways in an attempt to decipher gene expression data in a systems biology context, we were able to demonstrate that such an approach can be effective in helping biologists to elucidate molecular mechanisms underlying a given biological question. The most notable contribution however was the development of MicroPath (Chapter 5), which emphasized on the need to analyze multiple biologically related gene expression datasets in a systems biology context. The result of this is that a novel rationale has emerged, which could have putative benefits in treating disease. Before this rationale is explained, some of the weaknesses of this research need to be addressed.

6.2 Weaknesses

The fundamental weakness of the research presented in this thesis is based on its potential to be scalable. The first challenge is directly related to the limitation of MicroPath. Although the Permutation and Combination algorithm underlying MicroPath is important in the sense that it increases the probability of finding a common gene across multiple gene expression datasets, in its current state it is only able to cater for the comparison of 10 gene expression datasets. In light of this, the algorithm needs to be optimized so that it can handle the cross comparison of a much larger number of datasets.

The second limitation of this research is that the several different functions developed, namely transcription factor binding site analysis and molecular pathway analysis, need to be more tightly integrated. Currently, the software tools developed during the course of this research exist as autonomous pipelines. Because each of our software have different functionalities, it is imperative for these key functions to be more tightly integrated together in a scalable fashion to generate an overall pipeline that would apply a true systems biology method to the analysis of gene expression profiles.

What this effectively means is that MicroPath needs to be scaled so that it can handle a much larger number of expression dataset comparisons coupled with a need to integrate it with TFBM analysis from BiSAn. Furthermore, although the current state of Micropath allows it to map 10 expression profiles to all molecular pathways known to date, this function needs to be optimized to cater for a much larger number of datasets. Recognizing these limitations and weaknesses, a rationale is proposed in the following section of this chapter

6.3 A rationale for treating disease - Future Work

It cannot be denied that high throughput technologies such as microarrays have rapidly gained popularity at a global scale due to the prospect of quantifying gene expression in a high throughput fashion and subsequently identifying previously unknown transcriptome states. For this reason, gene expression data pertaining to various different biological questions are being rapidly generated by scientists worldwide and such datasets are now readily accessible through public repositories such as ArrayExpress (Sarkans *et al.,* 2005) and the Gene Expression Omnibus (GEO) (Barrett *et al.*, 2006). This has motivated biologists to utilise these sets of data in an attempt to investigate common regulatory signatures that can be potentially found across multiple experiments sharing a similar biological theme. One of the most common methods of comparison is based on the assumption that genes across different biological conditions having similar expression patterns are likely to be involved in the same biological process (Rhodes *et al.*, 2004) and hence, may share the same regulatory signatures. Using this method of comparison, which is one of the most successful methods to date, coupled with the availability of publicly accessible gene expression data repositories, biologists now have the opportunity to answer complex biological questions pertaining to biological phenomena underlying various different disease states. Chapter 5 demonstrated the works of Khan *et al.*, (*In Press*) illustrating a novel algorithm that applied the principle of Permutations and Combinations to increase the probability of identifying common genes across multiple expression profiles. The need for optimizing this algorithm has been explained above and now, the importance of this optimization will be discussed.

Gene Expression Omnibus (GEO) and ArrayExpress contain hundreds and thousands of expression profiles relevant to a variety of different biological questions underlying many diseases such as Cancer. A common complaint is that because such data repositories hold vast volumes of microarray data in different file formats, it becomes problematic to facilitate their comparisons due to heterogeneity. The second problem is that if hundreds or thousands of datasets are compared, chances of finding a common gene identifier across all datasets would be very low. The first problem can be tackled by segregating expression profiles according to the format they subscribe to. This would create sets of homogeneous profiles, which can be compared with ease. The solution to the second problem lies in the optimization of MicroPath due to its ability to search for common gene identifiers across $n - 1$ profiles, when a simple number crunching method of comparison fails to yield any results. Coupled with the algorithm underlying BiSAn, and optimization of the High Throughput pathway analysis function, it becomes apparent that this rationale could lead to some important biological findings at the disease level. For instance if common genes identified across hundreds of different expression profiles pertinent to a certain type of Cancer exhibit similar expression patterns, it may constitute important *in silico* findings because there may be candidate genes present within the intersection. Such genes could then be investigated at the *in vivo* level, where certain candidate oncogenes for example could be silenced in tumerogenic mice to see the observed effects.

We are faced with a situation where we have vast amounts of biological data available to decipher, but not enough focus is being shifted to it. The answer to several biological questions lies in the data itself but sophisticated algorithms and automated software pipelines are required to effectively mine them in a true systems biology context.

Appendix

A. Code

The magnitude of code written for developing the software presented in this thesis is too vast to document. Therefore, only specific key sections of the code will be shown here, mainly relevant to some of the algorithms developed for the purpose of this research.

A1. High throughput TFBM Detection

The following code was written in Perl to facilitate high throughput Transcription Factor Binding Motif (TFBM) detection for microarray data. This code pertains to the underlying algorithm explained in Chapter 3 of this thesis.

```
#!/usr/bin/perl
use strict;
use Win32::OLE qw(in with);
use Win32::OLE::Const 'Microsoft Excel';

open(INFILE, "<C:/Perl/eg/userdata.txt") or die ("couldn't open the file
userdata.txt: $!\n");
my @udata = <INFILE>;
close(INFILE);

my $promoter;
my @temp=();
my @tempp=();

my $count;
my $coun;
my $cou;

my $jasparid;
my $tfname;
my $class;
my $species;
my $sysgroup;
my $length;
my $max;

my @digit=();
my $digit;
my $marks;
my $coll;
my @scorearray=();
my $scorearray;
my $percent;
my $totpercent;
my @percentarray=();
my $percentarray;
my $kount;
my @result=();
my $result;

my %final=();
my $final;
foreach my $udata(@udata)
{
chomp($udata);
$count=0;
$cou=0;
my @array1=split(/\s+/, $udata);
my $array1;
my $geneid=$array1[0];
my $genename=$array1[1];

print("--------------------------------------------------------------------
------------\n");
print("                       $geneid\t$genename\n\n");
print("--------------------------------------------------------------------
------------\n\n");


open(INFILE, "<C:/promoterid.txt") or die("couldn't open the file
promoterid.txt: $!\n");
my @promoterids=<INFILE>;
close(INFILE);


my $promoterids;
my $promoterseqs;

for my $i(0..$#promoterids)
{
chomp($promoterids[$i]);

if($geneid =~ m/$promoterids[$i]/i)
{


$cou++;

$Win32::OLE::Warn = 3;

my $Excel = Win32::OLE->new('Excel.Application', 'Quit');

my $Book = $Excel->Workbooks->Open("C:/promoterseqs.xls");
my $Sheet = $Book->Worksheets(1);
$promoter=$Sheet->Cells($i+1, 1)->{'Value'};
}
}
if($cou==0)
{
print("The gene was not found in the promoters database.\n\n");
print("---------------------------------------------------------------------------
-----------\n");
print("---------------------------------------------------------------------------
-----------\n\n");

goto end;
}


$Win32::OLE::Warn = 3;

my $Excel = Win32::OLE->new('Excel.Application', 'Quit');

my $Book = $Excel->Workbooks->Open("C:/mouse.xls");
my $Sheet = $Book->Worksheets(1);

for(my $row=1;$row<114;$row+=5)
      {

$kount=0;
$coun=0;
 my $pattern = $Sheet->Cells($row, 2)->{'Value'};


if($promoter =~ m/$pattern/gi)
            {
$coun++;
$count++;
my $endpos = pos($promoter);


$jasparid=$Sheet->Cells($row, 1)->{'Value'};
$tfname=$Sheet->Cells($row, 3)->{'Value'};
$class=$Sheet->Cells($row, 4)->{'Value'};
$species=$Sheet->Cells($row, 5)->{'Value'};
$sysgroup=$Sheet->Cells($row, 6)->{'Value'};
$length=$Sheet->Cells($row, 7)->{'Value'};
$max=$Sheet->Cells($row, 8)->{'Value'};


my $startpos=($endpos+1)-$length;
my $lastpos=$endpos;

my $consensus = substr($promoter, ($startpos-1), $length);

push(@temp, $consensus, $startpos, $lastpos);

}
if($coun==0)
{
goto nex;
}
push(@tempp, $tfname, $class, $species, $sysgroup, $jasparid);
```

```
my $temp;
my $tempp;

print("-------------------------------------\n");
print("Transcription Factor Name:\t$tempp[0]\n\n");
print("Transcription Factor Class:\t$tempp[1]\n\n");
print("Species:\t\t$tempp[2]\n\n");
print("Sysgroup:\t\t$tempp[3]\n\n");
my $size=@temp;
for (my $k=0;$k<=$size-1;$k+=3)
{
$coll=10;
$kount++;
print("Transcription Factor binding site Sequence :$temp[$k]\t");
@digit = split(//, $temp[$k]);
foreach $digit (@digit)
{
if($digit =~ m/A/)
{
$marks = $Sheet->Cells($row, $coll)->{'Value'};
push(@scorearray, $marks);
}
if($digit =~ m/C/)
{
$marks = $Sheet->Cells($row+1, $coll)->{'Value'};
push(@scorearray, $marks);
}
if($digit =~ m/G/)
{
$marks = $Sheet->Cells($row+2, $coll)->{'Value'};
push(@scorearray, $marks);
}
if($digit =~ m/T/)
{
$marks = $Sheet->Cells($row+3, $coll)->{'Value'};
push(@scorearray, $marks);
}
$coll++;
}
@digit=();
my $tem=0;
foreach $scorearray(@scorearray)
{
$tem = $tem+$scorearray;
}
@scorearray = ();
$percent = ($tem/($length*$max))*100;
$totpercent = ($tem/(10.09*29.35))*100;
push(@percentarray, $totpercent);
print("Start position in promoter: \"$temp[$k+1]\"\t");
print("End position : \"$temp[$k+2]\"\n\n\n");
my $fpercent = sprintf("%.2f", $percent);
print("Possibility of binding of the Transcription Factor $tempp[0] at the
above positions of promoter site of the gene $genename ($geneid) is :
$fpercent\%\n\n\n");

}

my $url= "http://jaspar.genereg.net/cgi-bin/jaspar_db.pl?ID=" . $tempp[4] .
"\&rm=present\&db=0";
```

```
print("The URL to the JASPAR database for this Transcription Factor:\n\n
$url\n");
print("\n\n\n");
my $whole=0;
foreach $percentarray(@percentarray)
{
$whole = $whole+$percentarray;
}
@percentarray = ();
my $overall = ($whole/($kount*100))*100;
my $strin = ($tempp[0]."\t".$kount."\t".$url);
push(@result, $strin, $overall);

@temp=();
@tempp=();
  next:
}
if($count>0)
{
for (my $l=0;$l<=$#result;$l+=2)
      {
my $fresult = sprintf("%.2f", $result[$l+1]);
$final{$result[$l]} = $fresult;
}
print("------------------------------------------------------------------
-----------\n\n");
print("Order of Transcription factors based on their possible binding
affinity to this gene.\n\n");
print("------------------------------------------------------------------
-----------\n\n");
print("TFname\tNo.of binding sites\tJASPAR link\tBinding affinity
percentage\n\n");
print("------------------------------------------------------------------
-----------\n\n");
foreach my $value (sort {$final{$b} cmp $final{$a} }
 keys %final)
{
    print "$value\t $final{$value}\%\n\n";
}
%final=();
print("------------------------------------------------------------------
-----------\n");
print("------------------------------------------------------------------
-----------\n");
print("\n\n");
goto end;
}
print("No Transcription Factor was found for this gene.\n\n\n");
print("------------------------------------------------------------------
------------\n");
print("------------------------------------------------------------------
------------\n\n\n");
end:
}
```

A1.1 TFBM Enrichment Analysis

The following code was written to carry out TFBM enrichment analysis as a part of BiSAn:

```perl
#!/usr/bin/perl

use strict;
use Win32::OLE qw(in with);
use Win32::OLE::Const 'Microsoft Excel';
use Text::NSP::Measures::2D::Fisher2::right;


my @promoterseqs = ();
my @consensusseqs = ();


my @geneids = ();
my @genenames = ();

my @userproms = ();

my $count;
my $kount;
my $cou=1;

my $promoterseqs;
my $userproms;

my $consensusid;
my $consensusname;

my $rightpvalue;
my $errorCode;

my $absenusers;

my $check;
my $checktwo;

my $kou=0;

my @array;
my $array;

my $size;
my $notfound;


open(INFILE, "<C:/userdata.txt") or die ("couldn't open the file
userdata.txt: $!\n");
my @udata = <INFILE>;
close(INFILE);

foreach my $udata(@udata)
{
	chomp($udata);
```

```
         @array = split(/\s+/, $udata);


        chomp($array[0]);

        chomp($array[1]);

        push(@geneids, $array[0]);

        push(@genenames, $array[1]);


}


open(INFILE, "<C:/promoterid.txt") or die("couldn't open the file
promoterid.txt: $!\n");

my @promoterids = <INFILE>;

close(INFILE);


        foreach my $geneids(@geneids)
        {
                chomp($geneids);

                for my $k(0..$#promoterids)
                {
                        chomp($promoterids[$k]);

                        if($promoterids[$k] =~ m/$geneids/i)
                        {
                                $kou++;


                                $Win32::OLE::Warn = 3;

                                my $Excel = Win32::OLE->new('Excel.Application',
'Quit');

                                my $Book = $Excel->workbooks-
>open("C:/promoterseqs.xls");

                                my $Sheet = $Book->worksheets(1);

                                my $prom = $Sheet->Cells($k+1, 1)->{'Value'};

                                push(@userproms, $prom);

                                goto next;

                        }

                }


        next:
      }

       print("\n\nPromoter sequences found from user's data = $kou\n\n");

       $size = @geneids;

      $notfound = ($size-$kou);

       print("Promoter sequences not found from user's data =
$notfound\n\n");


$Win32::OLE::Warn = 3;


my $Excel = Win32::OLE->new('Excel.Application', 'Quit');


my $Book = $Excel->Workbooks->open("C:/promoterseqs.xls");

my $Sheet = $Book->Worksheets(1);


for my $i(1..18073)
{

      my $promoter = $Sheet->Cells($i, 1)->{'Value'};

      push(@promoterseqs, $promoter);


}


$win32::OLE::Warn = 3;


my $Excel = Win32::OLE->new('Excel.Application', 'Quit');


my $Book = $Excel->Workbooks->open("C:/mouse.xls");

my $Sheet = $Book->Worksheets(1);


for(my $j=1;$j<114;$j+=5)
{

      my $consensus = $Sheet->Cells($j, 2)->{'Value'};


      push(@consensusseqs, $consensus);


}
```

```
foreach my $consensusseqs(@consensusseqs)
{
      chomp($consensusseqs);


      $count = 0;




      foreach $promoterseqs(@promoterseqs)
      {


       chomp($promoterseqs);

      if($promoterseqs =~ m/$consensusseqs/i)
         {


            $count++;


         }




      }




       print("The consensus binding site $consensusseqs occurs $count times
in the genome.\n\n");


      $kount = 0;


      foreach $userproms(@userproms)
      {


            chomp($userproms);

            if($userproms =~ m/$consensusseqs/i)
            {

                  $kount++;


            }

      }



      $win32::OLE::Warn = 3;


my $Excel = Win32::OLE->new('Excel.Application', 'Quit');
my $Book = $Excel->Workbooks->open("C:/mouse.xls");

my $Sheet = $Book->Worksheets(1);


      $consensusid = $Sheet->Cells($cou, 1)->{'Value'};

      $consensusname = $Sheet->Cells($cou, 3)->{'Value'};


      $cou+=5;


      print("This consensus binding site is found $kount times in the
user's data.\n\n");


      $absenusers = ($kou-$kount);

      my $second = ($count-$kount);

      my $fourth = (18073-($kount+$absenusers+$second));



      my $npp = ($kount+$absenusers+$second+$fourth);
      my $n1p = ($kount+$second);
       my $np1 = ($kount+$absenusers);
        my $n11 = $kount;

  $rightpvalue = calculateStatistic( n11=>$n11,
                                      n1p=>$n1p,
                                      np1=>$np1,
                                      npp=>$npp);



 print ("$consensusid-($consensusname)-$consensusseqs\n\n

            p-value = $rightpvalue\n\n
      --------------------------------------------\n\n");




}
```

A2. High throughput Molecular pathway analysis

A2.1 Analyzing Genbank Accession Identifiers in KEGG pathways

The following code was written in Perl to facilitate high throughput molecular pathway analysis of microarray data by connecting to the Application Programming Interface (API) of KEGG in a live fashion. This code is specific to Genbank Accession ID's so the code picks up these ID's from the users input file and searches for them in all of KEGG's pathways. Where found, genes are highlighted on the pathways automatically. A detailed explanation of this function is explained in Chapter 4 of this thesis in the section pertaining to high throughput molecular pathway analysis.

```
#!/usr/bin/perl -w
open(FILE, "C:/mousekegg.txt");
my @array = <FILE>;
print "\n\n";
print "                                         THE RESULTS
      \n";
print "---------------------------------------------------------------------
---------------------------------------------------------------------------
-----------------";
my $count = 1;
foreach my $array (@array)
{print "\n\n\n$count ) the gene $count from the user's data:\n\n\n";
      $count++;

chomp ($array);
  my @temp = split (/\s+/, $array);
  for(my $j=1;$j<=1;$j++)
{
        my $key = $temp[0];
        print "\t$key\n";
        my $val = $temp[1];

  my $exprr = $temp[2];
print("HUGO name of the gene: $temp[1]\n\n");
      print("mean expression of the gene: $exprr\n\n");
      my @aaaa=split(//, $exprr);
      my $express=$aaaa[0];

        if ($express eq "+")
          {
            print "upregulated\n\n";

          }
         elsif ($express eq "-")
            { print "downregulated\n\n";
            }
use Data::Dumper;
use SOAP::Lite +trace => [qw(debug)];


my $serv = SOAP::Lite ->service("http://soap.genome.jp/KEGG.wsdl");
my $result = $serv ->bconv("genbank:$key");
my $length = length($result);
unless($length==0)
{

my @tem = split(/\s+/, $result);
 for(my $k=1;$k<=1;$k++)
{
      my $id = $tem[1];

my $keggid = [$id];

  my $arrayRef = $serv ->get_pathways_by_genes([$id]);


my $final = Dumper $arrayRef;


my @hifi = split(/'/, $final);
my $size = @hifi;
 for(my $i=1;$i<$size;$i=$i+2)

 {
      my $keggpath = $hifi[$i];



      if ($express eq "+")
      {

      print "kegg pathway id is : $keggpath\n\n";




            my $fg_list= ['green'];
                  my $bg_list=['red'];
my $extreme = $serv->get_html_of_colored_pathway_by_objects($keggpath,
$keggid, $fg_list, $bg_list);

print $extreme, "\n\n";
}
elsif ($express eq "-")
{ print "kegg pathway id is : $keggpath \n\n";

 $fg_list= ['red'];
 $bg_list=['green'];

 $extreme = $serv->get_html_of_colored_pathway_by_objects($keggpath,
$keggid, $fg_list, $bg_list);

print $extreme, "\n\n";

}
```

```
}goto outside;
}
}

use Win32::OLE qw(in with);
use Win32::OLE::Const 'Microsoft Excel';

$Win32::OLE::Warn = 3;


my $Excel = Win32::OLE->new('Excel.Application', 'Quit');


my $Book = $Excel->Workbooks->Open("C:/MMU.xls");


my $Sheet = $Book->Worksheets(1);

foreach my $row (1..4829)
{
for (my $col=3;$col<=3;$col++)
 {

  next unless defined $Sheet->Cells($row,$col)->{'Value'};
  my $id = $Sheet->Cells($row, $col)->{'Value'};

            while ($id =~ m/$key/)
                {

                        my $mainid=$id;
                   my $entrez = $Sheet->Cells($row,2)->{'Value'};
my $gene = $Sheet->Cells($row,1)->{'Value'};
 my $pathid = $Sheet->Cells($row,4)->{'Value'};
 my @pathway = split(/ /, $pathid);


my $size = @pathway;

for (my $i=0; $i<$size; $i++)
{use SOAP::Lite +trace => [qw(debug)];


my $serv = SOAP::Lite ->service("http://soap.genome.jp/KEGG.wsdl");
if ($express eq "+")
{
print "kegg pathway id is : $pathway[$i] \n\n";
my $keggid=["mmu:".$entrez];
my $keggpath = ("path:".$pathway[$i]);
my $fg_list= ['green'];
my $bg_list=['red'];
my $result = $serv->get_html_of_colored_pathway_by_objects($keggpath,
$keggid, $fg_list, $bg_list);
print $result, "\n\n";
}
elsif ($express eq "-")
      { print "kegg pathway id is : $pathway[$i] \n\n";
      my $keggid=["mmu:".$entrez];
      my $keggpath = ("path:".$pathway[$i]);
      my $fg_list= ['red'];
      my $bg_list=['green'];
my $result = $serv->get_html_of_colored_pathway_by_objects($keggpath,
$keggid, $fg_list, $bg_list);

      print $result, "\n\n";
      }
}
  goto outside;
 }
       }
      } print "the gene is not found \n\n\n";
print "try by converting this gene id into entrez id and use the entrez
button to get the perfect results\n\n\n";
goto outside;
}
outside:
}

sub SOAP::Serializer::as_ArrayOfstring{
  my ($self, $value, $name, $type, $attr) = @_;
  return [$name, {'xsi:type' => 'array', %$attr}, $value];
}
sub SOAP::Serializer::as_ArrayOfint{
  my ($self, $value, $name, $type, $attr) = @_;
  return [$name, {'xsi:type' => 'array', %$attr}, $value];
}
```

A2.2 Analyzing Entrez Identifiers in KEGG pathways

Sometimes Genbank Accession Identifiers are not found in KEGG pathways and consequently, these genes will not be highlighted on pathways although they may be present under a different identifier. To deal with this problem, code was written in Perl to search for entrez ID's in all of KEGG's pathways in the event of the Genbank analysis yielding no results.

```
#!/usr/bin/perl -w
open(FILE, "C:/mousekegg.txt");
my @array = <FILE>;
print "                                   THE RESULTS
      \n";
print "----------------------------------------------------------------------
------------\n\n\n";
my $count = 1;
foreach my $array (@array)
{print "\n$count ) the gene $count from the users data:\n\n\n";
      $count++;
 print ("$array\n");
chomp ($array);
  my @temp = split (/\s+/, $array);
  for(my $j=1;$j<=1;$j++)
      {
        my $key = $temp[0];
print "\nusers input gene id is: $key\n\n";
 my $exprr = $temp[2];

print("HUGO name of the gene: $temp[1]\n\n");
print("mean expression of the gene: $exprr\n\n");
my @aaaa=split(//, $exprr);
my $express=$aaaa[0];

 print "expression of the gene in users experiment : $express\n\n";
 if ($express eq "+")
{
print "the gene is upregulated.\n\n";
}
  elsif ($express eq "-")
{ print "the gene is downregulated.\n\n";
}
use Data::Dumper;
use SOAP::Lite +trace => [qw(debug)];


my $serv = SOAP::Lite ->service("http://soap.genome.jp/KEGG.wsdl");
my $keggid=["mmu:".$key];
my $id = "mmu:".$key;
my $arrayRef = $serv ->get_pathways_by_genes([$id]);
my $final = Dumper $arrayRef;
my @hifi = split(/'/, $final);
my $size = @hifi;
unless ($size == 0)
{
for(my $i=1;$i<$size;$i=$i+2)
 {
my $keggpath = $hifi[$i];
print $keggpath, "\n\n";
if ($express eq "+")
{
print "kegg pathway id is : $keggpath\n\n";

my $fg_list= ['green'];
my $bg_list=['red'];
my $extreme = $serv->get_html_of_colored_pathway_by_objects($keggpath,
$keggid, $fg_list, $bg_list);
print $extreme, "\n\n";
}
elsif ($express eq "-")
{ print "kegg pathway id is : $keggpath \n\n";

 $fg_list= ['red'];
 $bg_list=['green'];
 $extreme = $serv->get_html_of_colored_pathway_by_objects($keggpath,
$keggid, $fg_list, $bg_list);
print $extreme, "\n\n";
}
}goto outside;
}
 print "\n\nthe gene is not found in KEGG pathways\n\n\n\n";
}
outside:
}

sub SOAP::Serializer::as_ArrayOfstring{
 my ($self, $value, $name, $type, $attr) = @_;
 return [$name, {'xsi:type' => 'array', %$attr}, $value];
}

sub SOAP::Serializer::as_ArrayOfint{
  my ($self, $value, $name, $type, $attr) = @_;
  return [$name, {'xsi:type' => 'array', %$attr}, $value];
}
```

A2.3 Calculating percentage of genes from input that participate in *X*% of genes in a pathway

MicroPath calculates the number of genes identified in a given pathway and 1) expresses this as a percentage in relation to the total number of common genes from the intersection and 2) expresses this as a percentage in relation to the total number of genes belonging to that pathway. The following code was written specifically for this function.

```
#!/usr/bin/perl
open(FILE, "C:/Perl/eg/userdata.txt");
my @inp = <FILE>;
print "\n\n";
print "                                   THE RESULTS
      \n";
print "--------------------------------------------------------------------
--------------------------------------------------------------------------
-----------------";
my $count = 1;
$no = 1;
@ array = ();
@notfound = ();
foreach my $inp (@inp)
{
chomp ($inp);
  @temp = split (/\s+/, $inp);
  for(my $j=1;$j<=1;$j++)
{
 $gene = $temp[0];
$val = $temp[1];

  $express = $temp[2];
use Data::Dumper;
use SOAP::Lite +trace => [qw(debug)];

my $serv = SOAP::Lite ->service("http://soap.genome.jp/KEGG.wsdl");
my $rresult = $serv ->bconv("genbank:$gene");
my $length = length($rresult);
unless($length==0)
{
my @tem = split(/\s+/, $rresult);
 for(my $k=1;$k<=1;$k++)
{
my $iid = $tem[1];

  my $arrayRef = $serv ->get_pathways_by_genes([$iid]);

my $final = Dumper $arrayRef;
 my @hifi = split(/'/, $final);
my $size = @hifi;
 for(my $i=1;$i<$size;$i=$i+2)
 {
```

```
$keggpath = $hifi[$i];

  if($no>0)
{
$strin = ($keggpath . "£" . $gene . "£" . $express . "£" . $iid);
push(@array, $strin);
}
else {       $si = @array;
for($a=0;$a<$si;$a++)
{
@pat = split(/£/, $array[$a]);
while($pat[0]=~m/$keggpath/)
{
$string = ($array[$a] . "£" . $gene . "£" . $express . "£" . $iid);
splice(@array, $a, 1);
push(@array, $string);
goto side;
}
} $yd = ($keggpath . "£" . $gene . "£" . $express . "£" . $iid);
push(@array, $yd);
side:
}

 }goto outside;
}
}

use Win32::OLE qw(in with);
use Win32::OLE::Const 'Microsoft Excel';
$Win32::OLE::Warn = 3;

my $Excel = Win32::OLE->new('Excel.Application', 'Quit');

my $Book = $Excel->Workbooks->Open("C:/Perl/eg/MMU.xls");


my $Sheet = $Book->Worksheets(1);
foreach my $row (1..4829)
{
for (my $col=3;$col<=3;$col++)
 {

  next unless defined $Sheet->Cells($row,$col)->{'Value'};
  my $id = $Sheet->Cells($row, $col)->{'Value'};

while ($id =~ m/$gene/)
 {
my $pathid = $Sheet->Cells($row,4)->{'Value'};
my $entrez = $Sheet->Cells($row,2)->{'Value'};
 my @pathway = split(/ /, $pathid);
my $cize = @pathway;
for (my $x=0; $x<$cize; $x++)
{
$keggiid = ("mmu:".$entrez);
$keggp = ("path:".$pathway[$x]);
if($no>0)
{
$strinn = ($keggp . "£" . $gene . "£" . $express . "£" . $keggiid);

push(@array, $strinn);
}
else {       $si=@array;
for($b=0;$b<$si;$b++)
{
@patt = split(/£/, $array[$b]);
while($patt[0]=~m/$keggp/)
{
$stringg = ($array[$b] . "£" . $gene . "£" . $express . "£" . $keggiid);
splice(@array, $b, 1);
push(@array, $stringg);
goto side;
}
} $yyd = ($keggp . "£" . $gene . "£" . $express . "£" . $keggiid);
push(@array, $yyd);
side:
}
}
goto outside;
  }
 }
} push(@notfound, $gene);
outside:
}
$no=0;
}
use SOAP::Lite +trace => [qw(debug)];
my $serv = SOAP::Lite ->service("http://soap.genome.jp/KEGG.wsdl");
@jean=();
@obb=();
@blist = ();
@flist = ();
@expre=();
$num = 1;
foreach $array (@array)
{
@all = split(/£/, $array);
$ize = @all;
my $kepath = ($all[0]);
for($n=3;$n<$ize;$n+=3)
{      $ob = $all[$n];
push(@obb, $ob);
$e = ($n-1);
$je = ($n-2);
$ex = $all[$e];
$ex = substr($ex, 0, 1);
$jea = $all[$je];
if($ex eq "+")
{
$fj ='green';
push(@flist, $fj);
$bj = 'red';
push(@blist, $bj);
push(@expre, 'upregulated');
push(@jean, $jea);
}
elsif($ex eq "-")
{
$fj ='red';
push(@flist, $fz);
$bj= 'yellow';
```

```perl
push(@blist, $bz);
push(@expre, 'downregulated');
push(@jean, $jea);
}

}
$ci = @jean;
$sy = @obb;
if($sy==1)
{
      $obj_list=[$obb[0]];
      $fg_list=[$flist[0]];
$bg_list=[$blist[0]];
       $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==2)
{
      $obj_list=[$obb[0],$obb[1]];
      $fg_list=[$flist[0],$flist[1]];
      $bg_list=[$blist[0],$blist[1]];
       $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==3)
{
      $obj_list=[$obb[0],$obb[1],$obb[2]];
      $fg_list=[$flist[0],$flist[1],$flist[2]];
$bg_list=[$blist[0],$blist[1],$blist[2]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==4)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==5)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==6)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==7)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
```

```
t[6]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6]];
$result = $serv->get_html_of_colored_pathway_by_objects($kepath, $obj_list,
$fg_list, $bg_list);
}
elsif($sy==8)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7]]
;
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==9)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==11)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==12)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
```

```
elsif($sy==13)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==14)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$
$result = $serv->get_html_of_colored_pathway_by_objects($kepath, $obj_list,
$fg_list, $bg_list);
}
elsif($sy==15)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==16)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==17)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16]];
```

```
$result = $serv->get_html_of_colored_pathway_by_objects($kepath, $obj_list,
$fg_list, $bg_list);
}
elsif($sy==18)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17]];

$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17]];
$result = $serv->get_html_of_colored_pathway_by_objects($kepath, $obj_list,
$fg_list, $bg_list);
}
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==20)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18],$obb[19]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18],$flist[19]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18],$blist[19]];
$result = $serv->get_html_of_colored_pathway_by_objects($kepath, $obj_list,
$fg_list, $bg_list);
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18],$obb[19],$obb[20]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18],$flist[19],$flis
t[20]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18],$blist[19],$blis
t[20]];
$result = $serv->get_html_of_colored_pathway_by_objects($kepath, $obj_list,
$fg_list, $bg_list);
}
elsif($sy==22)
```

```
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18],$obb[19],$obb[20],$obb[21]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18],$flist[19],$flis
t[20],$flist[21]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18],$blist[19],$blis
t[20],$blist[21]];
$result = $serv->get_html_of_colored_pathway_by_objects($kepath, $obj_list,
$fg_list, $bg_list);
elsif($sy==23)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18],$obb[19],$obb[20],$obb[21],$obb[22]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18],$flist[19],$flis
t[20],$flist[21],$flist[22]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18],$blist[19],$blis
t[20],$blist[21],$blist[22]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==24)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18],$obb[19],$obb[20],$obb[21],$obb[22],$obb[23]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18],$flist[19],$flis
t[20],$flist[21],$flist[22],$flist[23]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18],$blist[19],$blis
t[20],$blist[21],$blist[22],$blist[23]];
$result = $serv->get_html_of_colored_pathway_by_objects($kepath, $obj_list,
$fg_list, $bg_list);
}
elsif($sy==25)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18],$obb[19],$obb[20],$obb[21],$obb[22],$obb[23],$obb[24]
];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18],$flist[19],$flis
t[20],$flist[21],$flist[22],$flist[23],$flist[24]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18],$blist[19],$blis
t[20],$blist[21],$blist[22],$blist[23],$blist[24]];
```

```
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==26)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18],$obb[19],$obb[20],$obb[21],$obb[22],$obb[23],$obb[24]
,$obb[25]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18],$flist[19],$flis
t[20],$flist[21],$flist[22],$flist[23],$flist[24],$flist[25]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18],$blist[19],$blis
t[20],$blist[21],$blist[22],$blist[23],$blist[24],$blist[25]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==27)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18],$obb[19],$obb[20],$obb[21],$obb[22],$obb[23],$obb[24]
,$obb[25],$obb[26]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18],$flist[19],$flis
t[20],$flist[21],$flist[22],$flist[23],$flist[24],$flist[25],$flist[26]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18],$blist[19],$blis
t[20],$blist[21],$blist[22],$blist[23],$blist[24],$blist[25],$blist[26]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==28)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18],$obb[19],$obb[20],$obb[21],$obb[22],$obb[23],$obb[24]
,$obb[25],$obb[26],$obb[27]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18],$flist[19],$flis
t[20],$flist[21],$flist[22],$flist[23],$flist[24],$flist[25],$flist[26],$fl
ist[27]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18],$blist[19],$blis
t[20],$blist[21],$blist[22],$blist[23],$blist[24],$blist[25],$blist[26],$bl
ist[27]];
 $result = $serv->get_html_of_colored_pathway_by_objects($kepath,
$obj_list, $fg_list, $bg_list);
}
elsif($sy==29)
{
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18],$obb[19],$obb[20],$obb[21],$obb[22],$obb[23],$obb[24]
,$obb[25],$obb[26],$obb[27],$obb[28]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18],$flist[19],$flis
t[20],$flist[21],$flist[22],$flist[23],$flist[24],$flist[25],$flist[26],$fl
ist[27],$flist[28]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18],$blist[19],$blis
t[20],$blist[21],$blist[22],$blist[23],$blist[24],$blist[25],$blist[26],$bl
ist[27],$blist[28]];
$result = $serv->get_html_of_colored_pathway_by_objects($kepath, $obj_list,
$fg_list, $bg_list);
$obj_list=[$obb[0],$obb[1],$obb[2],$obb[3],$obb[4],$obb[5],$obb[6],$obb[7],
$obb[8],$obb[9],$obb[10],$obb[11],$obb[12],$obb[13],$obb[14],$obb[15],$obb[
16],$obb[17],$obb[18],$obb[19],$obb[20],$obb[21],$obb[22],$obb[23],$obb[24]
,$obb[25],$obb[26],$obb[27],$obb[28],$obb[29]];
$fg_list=[$flist[0],$flist[1],$flist[2],$flist[3],$flist[4],$flist[5],$flis
t[6],$flist[7],$flist[8],$flist[9],$flist[10],$flist[11],$flist[12],$flist[
13],$flist[14],$flist[15],$flist[16],$flist[17],$flist[18],$flist[19],$flis
t[20],$flist[21],$flist[22],$flist[23],$flist[24],$flist[25],$flist[26],$fl
ist[27],$flist[28],$flist[29]];
$bg_list=[$blist[0],$blist[1],$blist[2],$blist[3],$blist[4],$blist[5],$blis
t[6],$blist[7],$blist[8],$blist[9],$blist[10],$blist[11],$blist[12],$blist[
13],$blist[14],$blist[15],$blist[16],$blist[17],$blist[18],$blist[19],$blis
t[20],$blist[21],$blist[22],$blist[23],$blist[24],$blist[25],$blist[26],$bl
ist[27],$blist[28],$blist[29]];
$result = $serv->get_html_of_colored_pathway_by_objects($kepath, $obj_list,
$fg_list, $bg_list);
}


my @super;
push(@super, "The genes involved in this pathway from user's input data :
");
for($z=0;$z<$ci;$z++)
{

push(@super, "user's input id : $jean[$z]\n\nkegg gene id : $obb[$z] ");
   ";
push(@super, "The pathway id is : $kepath ");

push(@super, "The URL for the pathway is : ");
push(@super, "$result ");
my $totalgenes = $serv->get_genes_by_pathway($kepath);
my $totnogenes = Dumper $totalgenes;
my @totarray = split(/'/, $totnogenes);
my $totarray;
my @empty=();
for (my $new=1;$new<=$#totarray;$new+=2)
{
push(@empty, $totarray[$new]);
}
my $totno = @empty;
my $totinp = @inp;
my $inpperc = ($ci/$totinp)*100;
my $finpperc = sprintf("%.2f", $inpperc);

push(@super, "The percentage of genes from the user's data involving in
this pathway is : $finpperc\% ");
my $pathperc = ($ci/$totno)*100;
my $fpathperc = sprintf("%.2f", $pathperc);

push(@super, "$finpperc\% of genes from the user's data are contributing
$fpathperc\% of role in this pathway. ");
my $superr = join('£', @super);
@super = ();
$superfinal{$superr} = $fpathperc;
$num = $num+1;
@jean = ();
@obb = ();
@blist = ();
@flist = ();
@expre = ();
}
foreach my $value (sort {$superfinal{$b} cmp $final{$a} }
keys %superfinal)
{
my @superarray = split(/£/, $value);
foreach my $superarray (@superarray)
{
print("$superarray\n\n");
}
print("\n---------------------------------------------------------------------
--------------\n");
print("-----------------------------------------------------------------------
------------\n\n\n");
}
$saize = @notfound;
if($saize>0)
{
print "\nPathways were not found for the below genes\n\n";
foreach $notfound (@notfound)
{
print $notfound, "\n\n";
}
print "-----------------------------------------------------------------------
------------\n";
print "Try by converting the above gene ids into entrez ids and use the
entrez button from the interface to get the results\n\n";
print "-----------------------------------------------------------------------
------------\n"
}

sub SOAP::Serializer::as_ArrayOfstring{
 my ($self, $value, $name, $type, $attr) = @_;
 return [$name, {'xsi:type' => 'array', %$attr}, $value];
}
sub SOAP::Serializer::as_ArrayOfint{
 my ($self, $value, $name, $type, $attr) = @_;
return [$name, {'xsi:type' => 'array', %$attr}, $value];
}
```

Publications (Published/In Press)

I

II

III

IV

V

References

Adomas, A; Heller, G; Olson, A; Osborne, J; Karlsson, M; Nahalkova, J; Van Zyl, L; Sederoff, R; Stenlid, J; Finlay, R. and Asiegbu, FO. (2008). Comparative analysis of transcript abundance in Pinus sylvestris after challenge with a saprotrophic, pathogenic or mutualistic fungus. *TREE PHYSIOLOGY*. **28**(6):885-897

Allison, D. B., Cui, X., Page, G. P. and Sabripour, M. (2006). Microarray data analysis: from disarray to consolidation and consensus. *Nat Rev Genet*. **7**(5):406.

Al-Shahrour, F., Diaz-Uriarte, R. and Dopazo, J. (2004). FatiGO: a web tool for finding significant associations of Gene Ontology terms with groups of genes. *Bioinformatics*. **20**(4):578-580.

Anderson, P.O., Manzo, B.A., Sundstedt, A., Minaee, S., Symonds, A., Khalid, S., Rodriguez-Cabezas, M.E., Nicolson, K., Li, S., Wraith, D.C. and Wang, P. (2006) Persistent antigenic stimulation alters the transcription program in T cells, resulting in antigen-specific tolerance. *European Journal of Immunology***.** **36**, 1374-85.

Auman, J. T., Chou, J., Gerrish, K., Huang, Q., Jayadev, S., Blanchard, K. and Paules, R. S. (2007) Identification of Genes Implicated in Methapyrilene-Induced Hepatotoxicity by Comparing Differential Gene Expression in Target and Nontarget Tissue. *Environmental Health Perspectives*. **115**(4).

Babu, M. M., Luscombe, N. M., Aravind, L. Gerstein, M. and Teichmann, S. A. (2004). Structure and evolution of transcriptional regulatory networks. *Curr. Opin. Struct. Biol*. **14**(3): 283-291.

Bader G.D., Donaldson, I., Wolting, C., Ouellette, B.F.F., Pawson, T. and Hogue, C.W.V. (2001) BIND – The Biomolecular Interaction Network Database. *Nucleic Acids Res*, **29**, 242-245.

Barrett, T. and Ron Edgar. (2006) Mining Microarray Data at NCBI's Gene Expression Omnibus (GEO). *Methods Mol Biol* **338**: 175–190

Battistella, E., G.C.de Souza, J., Barcellos, C. K., Lemke, N. and Mombach, J. C. M. (2005) An integrated model for cellular analysis. *Genet. Mol. Res.* **4** (3): 506-513.

Beckstette M, Homann R, Giegerich R, and Kurtz S. (2006) Fast index based algorithms and software for matching position specific scoring matrices. *BMC Bioinformatics,* **7**:389.

Benkhart EM, Siedlar M, Wedel A, Werner T, Ziegler-Heitbrock HW. (2000) Role of Stat3 in lipopolysaccharide-induced IL-10 gene expression. *J Immunol* **165**,1612-7

Berriz, G.F. and Roth, F.P. (2008) The Synergizer service for translating gene, protein, and other biological identifiers. *Bioinformatics*. [Epub ahead of print].

Blanco, E., Farre D., Alba, M.M., Messeguer, X. and Guigo, R. (2006) ABS: a database of Annotated regulatory Binding Sites from orthologous promoters. *Nucleic Acids Res*, **34**.

Budhraja, V., Spitznagel, E., Schaiff, W. T. and Sadovsky, Y. (2003) Incorporation of gene-specific variability improves expression analysis using high-density DNA microarrays. *BMC Biol.* **1**, 1.

Bussey, K.J., Kane, D., Sunshine, M., Narasimhan, S., Nishizuka, S., Reinhold, W.C., Zeeberg, B., Ajay, W. and Weinstein, J.N. (2003) MatchMiner: a tool for batch navigation among gene and gene product identifiers. *Genome Biology.* **4**, R27.

Butcher, E. C., Berg, E. L. and Kunkel, E. J. (2004) Systems biology in drug discovery. *Nat Biotech.* **22,** 1253-1259.

Chekmenev DS, Haid C, and Kel AE. (2005) P-Match: transcription factor binding site search by combining patterns and weight matrices. *Nucleic Acids Res*, 33(Web Server issue):W432–W437.

Chen, Y., Dougherty, E. R. and Bittner, M. L. (1997) Ratio-based decisions and the quantitative analysis of cDNA microarray images. *J. Biomed. Opt.* **2**, 364-374.

Cheng, F., Wang, H.W., Cuenca, A., Huang, M., Ghansah, T., Brayer, J., Kerr, W.G., Takeda, K., Akira, S., Schoenberger, S.P., Yu, H., Jove, R. and Sotomayor, E.M. (2003) A Critical Role for Stat3 Signaling in Immune Tolerance. *Immunity* **19**, 425-36.

Cheung, TH; Kwan, YK; Hamady, M. and Liu, X. (2006) Unravelling transcriptional control and *cis*-regulatory codes using the software suite GeneACT. *Genome Biology,* **7:R97.**

Chung, H., Kim, M., Park, C.H., Kim, J. and Kim, J.H. (2004) ArrayXPath: mapping and visualizing microarray gene-expression data with integrated biological pathway resources using Scalable Vector Graphics. *Nucleic Acids Res*, **32**.

Crick, F. (1970) Central Dogma of Molecular Biology. *Nature*. **227**: 561-563.

Doniger SW, Salomonis N, Dahlquist KD, Vranizan K, Lawlor SC, Conklin BR. (2003) MAPPFinder: using Gene Ontology and GenMAPP to create a global gene-expression profile from microarray data. *Genome Biology*, **4(1):R7**.

Collins, K., Jacks, T. and Pavletich, N. P. (1997). The cell cycle and cancer. *PNAS*. **94**(7): 2776-2778.

Cummings, C. A. and Relman, D. A. (2000). Using DNA microarrays to study host-microbe interactions. *Emerg. Infec. Dis*. **6**(5): 513-525.

Dhillon, A. S., Hagan, S., Rath, O. and Kolch, W. (2007). MAP kinase signalling pathways in cancer. *Oncogene*. **26**, 3279-3290.

Draghici, S., Khatri, P., Tarca, A. L., Amin, K., Done, A., Voichita, C., Georgescu, C. and Romero, R. (2007). A systems biology approach for pathway level analysis. *Genome Res*. **17**, 1537-1545.

Dudoit, S., Yang, Y. H., Callow, M. J. and Speed, T. P. (2002) Statistical methods for identifying differentially expressed genes in replicated cDNA microarray experiments. *Statist Sinica*. **12**, 111-139.

Eisen, M. B., Spellman, P. T., Brown, P. O. and Botstein, D. (1998) Cluster analysis and display of genome-wide expression patterns. *Proc Natl Acad Sci USA*. 95. 25. 14863-14868.

Ekstrom, C. T., Bak, S., Kristensen, C. and Rudemo, M. (2004) Spot shape modelling and data transformations for microarrays. *Bioinformatics*. **20**, 2270-2278.

Everitt, B. (1980) Cluster analysis. 1st ed. Heinemann, London.

Falcon, S. and Gentleman, R. (2006) Using GOstats to test gene lists for GO term association. *Bioinformatics*. **23(2)**, 257-258.

Ghosh, D., Barette, T. R., Rhodes, D. and Chinnaiyan, A. M. (2003). Statistical issues and methods for meta-analysis of microarray data, A case study in prostate cancer. *Funct. Integr. Genomics* **3**, 180-188.

Halees, A.S., Leyfer, D. and Weng, Z. (2003) PromoSer: a large-scale mammalian promoter and transcription start site identification service. *Nucleic Acids Research***.** **31**, 3554-3559

Hand, D. J. and Heard, N. A. (2005) Finding groups in gene expression data. *J Biomed Biotechnol*. **2**, 15-25.

Herrero, J., Al-Shahrour, F., Diaz-Uriarte, R., Mateos, A., Vaquerizas, J.M., Santoyo, J. and Dopazo, J. (2003) GEPAS: a web-based resource for microarray gene expression data analysis. *Nucleic Acids Res*, **31**, 3461-3467.

Heyer, L. J., Kruglyak, S. and Yooseph, S. (1999) Exploring expression data: identification and analysis of coexpressed genes. *Genome Research.* **9**(11): 1106-1115.

Hsiao, A., Worrall, D. S., Olefsky, J. M. and Subramaniam, S. (2004) Variance-modeled posterior inference of microarray data: detecting gene-expression changes in 3T3-L1 adipocytes. *Bioinformatics.* **20**, 3108-3127.

Joshi-Tope, G., Gillespie, M., Vasrik, I., D'Eustachio, P., Schmidt, E., de Bone, B., Jassal, B., Gopinath, G.R., Wu, G.R., Matthews, L., et al. (2005). A knowledgebase of biological pathways. Nucleic Acids Res. **33**, D428-D432.

Kanehisa, M., Goto, S., Kawashima, S., Okuno, Y. and Hattori, M. (2004) The KEGG resource for deciphering the genome. *Nucleic Acids Res*, **32**

Karanam, S., and Moreno, C.S. (2004) CONFAC: automated application of comparative genomic promoter analysis to DNA microarray datasets. *Nucleic Acids Res*, **32**.

Kel AE, G¨ossling E, Reuter I, Cheremushkin E, Kel-Margoulis OV, and Wingender E. (2003) MATCH: A tool for searching transcription factor binding sites in DNA sequences. *Nucleic Acids Res*, 31(13):3576–3579.

Kel, A., Voss, N., Jauregui, R., Kel-Margoulis, O. and Wingender, E. (2006). Beyond microarrays: Finding key transcription factors controlling signal transduction pathways. *BMC Bioinformatics*. **7**(Suppl 2): S13

Khalid. S., Fraser, F., Khan, M., Wang, P., Liu, X. and Li, S. (2006a). Analysing Microarray Data using the Multi-functional Immune Ontologiser. J. Integrative Bioinformatics 3, 25.

Khalid, S., Khan, M., Wang, P., Liu, X. and Li, S.-L. (2006b). Application of bioinformatics in the design of gene expression microarrays. Second International Symposium on Leveraging Applications of Formal Methods, Verification and Validation (isola 2006), pp. 146-160.

Khalid, S., Khan, M., Gorle, CB; Fraser, K., Wang, P., Liu, X. and Li, S. (2008) MaXlab: A novel application for the cross comparison and integration of biological signatures from microarray studies. *In Silico Biology* 8, 0029.

Kitano, H. (2002a) Systems biology: a brief overview. *Science*. **295**: 1662-1664.

Kitano, H. (2002b) Computational systems biology. *Nature*. **420**: 206-210.

Kitano, H. (2002c) Looking beyond the details: a rise in system-oriented approaches in genetics and molecular biology. *Current Genetics*. **41**: 1-10.

Kohonen, T. *Self-organizing Maps*. Berlin, Germany: Springer; 1995.

Kuipers, H., Muskens, F., Willart, M., Hijdra, D., van, Assema FB., Coyle, AJ., Hoogsteden, HC. and Lambrecht, BN. (2006). Contribution of the PD-1 ligands/PD-1 signaling pathway to dendritic cell-mediated CD4(+) T cell activation. *Eur. J. Immunol*.**36 (9)**, 2472-82.

Kulesh, D. A., Clive, D. R., Zarlenga, D. S. and Greene, J. J. (1987). Identification of interferon-modulated proliferation-related cDNA sequences. *Proc. Natl. Acad. Sci. USA*. **84**(23): 8453-8457.

Kulterer *et al.* (2007) *BMC Genomics* **8**:70

Kulyk, O. and Wassink, I. Getting to know Bioinformaticians: Results of an exploratory user study, in British HCI Workshop Combining Visualizations and Interaction to Facilitate Scientific Exploration and Discovery, September 12 2006, Adriaansen T. And Zudilova-Seinstra E. V. (eds), London, UK, 30-36. 2006

Lee, M. L. T., Kuo, F. C., Whitmore, G. A. and Sklar. J. (2000) Importance of replication in microarray gene expression studies: statistical methods and evidence from repetitive cDNA hybridizations. *PNAS*. **97**, 9834-9839.

Li, X., Dou, K., Liu, H., Zhang, F. and Cai, L. (2007). Immune tolerance induced by IL-10 and methylprednisolone modified dendritic cells in vitro. *Chinese Journal of cellular and Molecular Immunol.* **23 (5)**, 436-8.

Lu, J. P., Beatty, L. K. and Pinthus, J. H. (2008). Dual expression recombinase based (DERB) single vector system for high throughput screening and verification of protein interactions in living cells. *Nature Precedings*.

Matys, V et al (2003) TRANSFAC: transcriptional regulation, from patterns to profiles. *Nucleic Acids Res,* **31**:1, 374-8.

Matys V, Kel-Margoulis OV, Fricke E, Liebich I, Land S, Barre-Dirrie A, Reuter I, Chekmenev D, Krull M, Hornischer K, Voss N, Stegmaier P, Lewicki-Potapov B, Saxel H, Kel AE, and Wingender E. (2006) TRANSFAC and its module TRANSCompel: transcriptional gene regulation in eukaryotes. *Nucleic Acids Res,* **34**(Database issue):D108–D110

Miller, R. A., Galecki, A. and Scmookler-Reis, R. J. (2001) Interpretation, design and analysis of gene array expression experiments. *J. Gerontol. A.* **56**, B52-B57.

Mintseris, J. and Weng, Z. (2005) Structure, function, and evolution of transient and obligate protein–protein interactions. *PNAS*. **102**(31): 10930-10935.

Mitchell, P. J. and Tjian, R. (1989). Transcriptional regulation in mammalian cells by sequence-specific DNA binding proteins. *Science*. **245**(4916): 371-378.

Muench R, Hiller K, Barg H, Heldt D, Linz S, Wingender E, and Jahn D. (2003) PRODORIC: prokaryotic database of gene regulation. *Nucleic Acids Res,* **31**(1):266–269.

Nikitin, A., Egorov, S., Daraselia, N. and Mazo, llya. (2003) Pathway studio – the analysis and navigation of molecular networks. *Bioinformatics,* **19**, 2155-2157.

Patel, M. and Nagl, S. (2004) Microcore: mapping genome expression to cell pathways and networks. *Comparative and Functional Genomics*, **5**, 75-78.

Peri, S. *et al.* (2003) Development of human protein reference database as an initial platform for approaching systems biology in humans. *Genome Research*. **13**:2363-2371.

Pfeffer LM, Mullersman JE, Pfeffer SR, Murti A, Shi W, Yang CH. (1997) STAT3 as an adapter to couple phosphatidylinositol 3-kinase to the IFNAR1 chain of the type I interferon receptor. *Science*, **276**, 1418-20

Ptashne, M. and Gann, A. (1997). Transcriptional activation by recruitment. *Nature*. **386**, 569-577.

Qin, J., Lewis, D. P. and Noble, W. S. (2003) Kernel hierarchical gene clustering from microarray expression data. *Bioinformatics.* **19**, 2097-2104.

Ressom, H., Wang, D. And Natarajan, P. (2003) Clustering gene expression data using adaptive double self-organizing map. *Physiol. Genomics*. **14**, 35-46.

Rhodes, D. R., Barrette, T. R., Rubin, M. A., Ghosh, D. and Chinnaiyan, A. M. (2002). Meta-analysis of microarrays: Interstudy validation of gene expression profiles reveals pathway dysregulation in prostate cancer. *Cancer Res*. **62**, 4427-4433.

Rhodes, D. R., Yu, J., Shanker, K., Deshpande, N., Varambally, R., Ghosh, D., Barrette, T., Pandey, A. and Chinnaiyan, A. M. (2004). Large-scale meta-analysis of cancer microarray data identifies common transcriptional profiles of neoplastic transformation and progression. *Proc. Natl. Acad. Sci. USA* **101**, 9309-9314.

Safford, M., Collins, S., Lutz, M.A., Allen, A., Huang, C., Kowalski J., Blackford, A., Horton, M.R., Drake, C., Schwartz, R.H. and Powell, J.D. (2005) Egr-2 and Egr-3 are negative regulators of T cell activation. *Nature Immunology*, **6**, 472-480.

Sandelin, A; Alkema, W; Engstrom, P; Wasserman, W; Lenhard, B. (2004) JASPAR: an open-access database for eukaryotic transcription factor binding profiles. *Nucleic Acids Res,* **32**:D91–4.

Sarkans, U., Parkinson, H., Lara, G.G., Oezcimen, A., Sharma, A., Abeygunawardena, N., Contrino, S., Holloway, E., Rocca-Serra, P., Mukherjee, G., Shojatalab, M., Kapushesky, M., Sansone, S.A., Farne, A., Rayner, T. and Brazma, A. (2005) The ArrayExpress gene expression database: a software engineering and implementation perspective**.** *Bioinformatics* **21(8)**:1495-1501.

Satoh, S., Daigo, Y., Furukawa Y., Kato, T., Miwa, N., Nishiwaki, T., Kawasoe, T., Ishigur0, H., Fujita, M., Tokino, T., Sasaki, Y., Imaoka, S., Murata, M., Shimano, T., Yamaoka, Y. and Nakamura, Y. (2000) *AXIN1* mutations in hepatocellular carcinomas, and growth suppression in cancer cells by virus-mediated transfer of *AXIN1*. *Nature Genetics*, **24**, 245-250.

Schadt, E. E., Li, C., Ellis, B. and Wong, W. H. (2001) Feature extraction and normalization algorithms for high-density oligonucleotide gene expression array data. *J. Cell. Biochem.* **Suppl. 37**, 120-125.

Schena, M., Shalon, D., Heller, R., Chai, A., Brown P. O. and Davies, R. W. (1996) Parallel Human Genome Analysis: Microarray-Based Expression Monitoring of 1,000 Genes. *PNAS*. **93**. 10614-10619.

Schena, M., Shalon, D., Davies, R. W. and Brown, P. O. (1995) Quantitative monitoring of gene expression patterns with a complementary DNA microarray. *Science*. **270**. 467-470.

Shamir, R., M.K, Adi., Tanay, A., Linhart, C., Steinfeld, I., Sharan, R., Shiloh, Y., and Elkon, R. (2005) EXPANDER – an integrative program suite for microarray data analysis. *BMC Bioinformatics,* **6**:232.

Shay, E. (2003). “Microarray cluster analysis and applications”, Available at: http://www.science.co.il/enuka/Essays/Microarray-Review.pdf.

Slonim. D. K. (2002) From patterns to pathways: gene expression data analysis comes of age. *Nature Genetics* **32**, 502: 508.

Spang, R. (2003) Diagnostic signatures from microarrays: a bioinformatics concept for personalized medicine. *BIOSILICO* 1, 2, 64-68.

Spellman, P. T., Sherlock, G., Zhang, M. Q. and Iyer, V. R. (1998). Comprehensive Identification of Cell Cycle-regulated Genes of the Yeast *Saccharomyces cerevisiae* by Microarray Hybridization. *Mol. Biol. Cell*. **9**, 3273-3297.

Statnikov, A., Aliferis, C. F., Tsamardinos, I., Hardin, D. and Levy, S. (2005) A comprehensive evaluation of multicategory classification methods for microarray gene expression cancer diagnosis. *Bioinformatics*. **21**(5): 631-643.

Steinfath, M *et al*. (2001) Automated image analysis for array hybridization experiments. *Bioinformatics*. **17**, 634-641.

Stelling, J. (2004). Mathematical models in microbial systems biology. *Curr. Opin. Microbiol.* **7**, 513-518.

Stormo GD, Schneider TD, Gold L, and Ehrenfeucht A. (1982) Use of the 'Perceptron' algorithm to distinguish translational initiation sites in E. coli. *Nucleic Acids Res*, **10**(9):2997–3011.

Stryer, L., Berg, J. and Tymoczko. J. L. (2006) Biochemistry 6th edition. WH Freeman (ISBN 0:7167:8724:5).

Sturn, A., Quackenbush, J. and Trajanoski, Z. (2002) Genesis: cluster analysis of microarray data. *Bioinformatics*, **18**, 207-208.

Subramanian, A., Tamayo, P., Mootha, V.K., Mukherjee, S., Ebert, B.L., Gillette, M.A., Paulovich, A., Pomeroy, S.L., Golub, T.R., Lander, E.S., et al. (2005). Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles. *Proc. Natl. Acad. Sci*. **102:** 15545–15550

Sundrud MS, Vancompernolle SE, Eger KA, Bruno TC, Subramaniam A, Mummidi S, Ahuja SK, Unutmaz D. (2005) Transcription factor GATA-1 potently represses the expression of the HIV-1 coreceptor CCR5 in human T cells and dendritic cells. *Blood,* **106**(10):3440-8.

Suzuki. H. et al (2001) Protein-Protein Interaction Panel using Mouse Full-Length cDNAs. *Genome Res*. **11**:1758-1765.

Teichmann SA and Babu MM. (2004) Gene regulatory network growth by duplication. *Nat Genet*, **36**(5):492–496.

Tavazoie, S., Hughes, D., Campbell, M. J., Cho, R. J. and Church, G. M. (1999) Systematic determination of genetic network architecture. *Nature Genet*, 281-285.

Thijs, G., Moreau, Y., Smet, F.D., Mathys, J., Lescot, M., Rombauts, S., Rouze, P., Moor, B.D and Marchal, K. (2002) INCLUSive: Integrated Clustering, Upstream sequence retrieval and motif Sampling. *Bioinformatics*, **18**, 331-332.

Thomas, P.D., Kejariwal, A., Campbell, M.J., Mi, H., Diemer, K., Guo, N., Ladunga, I., Ulitsky-Lazareva, B., Muruganujan, A., Rabkin, S., Vandergriff, J.A. and Doremieux, O. (2003) PANTHER: a browsable database of gene products organized by biological function, using curated protein family and subfamily classification. *Nucleic Acids Res*, **31**, 334-341.

Wang, J., Coombes, K. R., Highsmith, W. E., Keating, M. J. and Abruzzo, L. V. (2004). Differences in gene expression between B-cell chronic lymphocytic leukemia and normal B cells: A meta-analysis of three microarray studies. *Bioinformatics.* **20**, 3166-3178.

Warner, L.E., Svaren, J., Milbrandt, J. and Lupski, J.R. (1999) Functional consequences of mutations in the early growth response 2 gene (*EGR2*) correlate with severity of human myelinopathies. *Hum. Mol. Genet.* **8**, 1245-1251.

Weinstein, I. B., Begemann, M., Zhou, P., Han, E. K., Sgambato, A., Doki, Y., Arber, N., Ciaparrone, M. and Yamamoto, H. (1997). Disorders in cell circuitry associated with multistage carcinogenesis: exploitable targets for cancer prevention and therapy. *Clin. Cancer Res*. **3**, 2696.

Westerhoff, H. V. and Palsson, B. O. (2004) The evolution of molecular biology into systems biology. *Nat Biotech.* **22** (10): 1249-1252.

Wu, Xikun. and Watson, M. (2009) CORNA: Testing gene lists for regulation by microRNAs. *Bioinformatics.* **25** (6): 832-833.

Yang, Y. H., Buckley, M. J. and Speed, T. P. (2001) Analysis of cDNA microarray images. *Brief Bioinform*. **2**, 341-349.

Yusuf, I; Kharas, MG; Chen, J; Peralta, RQ; Maruniak, A; Sareen, P; Yang, VW; Kaestner ,KH. and Fruman , DA. (2008) KLF4 is a FOXO target gene that suppresses B cell proliferation. *Int Immunology*, **20**(5):671-681.

Zahurak, M., Parmigiani, G., Yu, W., Scharpf, R. B., Berman, D., Schaeffer, E., Shabbeer, S. and Cope, L. (2007) Pre-processing Agilent microarray data. *BMC Bioinformatics.* **8**, 142.

Zhang, Y., Chung, Y., Bishop, C., Daugherty, B., Chute, H., Holst, P., Kurahara, C., Lott, F., Sun, N., Welcher, A. A. and Dong, C. (2006). Regulation of T cell activation and tolerance by PDL2. *Proc Natl Acad Sci U S A*, **103(31)**, 11695-11700.

Ziegler-Heitbrock L, Lotzerich M, Schaefer A, Werner T, Frankenberger M, Benkhart E. (2003) IFN-alpha induces the human IL-10 gene by recruiting both IFN regulatory factor 1 and Stat3. *J Immunol*, **171**, 285-90.

http://www.biocarta.com - BioCarta, Charting pathways of life.

http://www.mysql.com – MySQL

http://biowulf.bu.edu/zlab/PromoSer/ - Promoser

http://discover.nci.nih.gov/matchminer - MatchMiner

http://www.hprd.org/ - HPRD

http://genome.gsc.riken.jp/ppi/ - Protein-Protein Interaction Panel using Mouse Full-Length cDNAs:

http://www.axon.com - GenePix pro 4.1

http://www.moleculardevices.com/pages/software/gn_acuity.html - Acuity 4.0

http://idconverter.bioinfo.cnio.es/ - Gene Id Converter

http://www.1066technologies.co.uk/bisan - BiSAn

http://www.ebi.ac.uk/arrayexpress/ - ArrayExpress