In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Computer Science

The University of British Columbia
Vancouver, Canada

Date August 15, 2002
Abstract

In recent years, relating gene expression to cancer development and treatment has received a lot of attention. The precise identification of a cancer’s type and stage is often crucial to the assignment of appropriate treatment. Therefore, a central goal of the analysis of gene expression data is to identify sets of genes that can be used for the classification or diagnosis of cancer. Another important purpose of gene expression studies is to improve the understanding of cellular responses to drug treatment, to identify drug responsive genes, and to discover the potential drug targets. To achieve these goals, researchers are applying various methods of analysis to gene expression data.

Unfortunately, the availability of effective analysis tools lags far behind the availability of data. In this thesis, we present the *Gene Expression Analyzer (GEA)* for performing cluster analysis on gene expression data. In particular, the GEA is developed to address the reality that cluster analysis is typically a multi-step process. The underlying model of the GEA provides a set of algebraic operators for manipulating the data, as well as intermediate results. Moreover, the GEA provides facilities to help the user to identify candidate genes for further clinical analysis. Last but not least, the GEA is optimized to handle the high dimensionality of gene expression data, and provides user-friendly features to help the user organize and manage intermediate analysis results.
# Table of Contents

Abstract ................................................................................................................................. ii
Table of Contents .................................................................................................................... iii
List of Figures ........................................................................................................................ vii
List of Tables .......................................................................................................................... x
Acknowledgements ............................................................................................................... xi
Dedication ............................................................................................................................... xii

## Chapter 1 Introduction
1.1 Data Mining ............................................................................................................. 1
1.2 Bioinformatics ........................................................................................................... 2
1.3 Why Analyze the Gene Expression Data .............................................................. 3
1.4 Motivation for Introducing the Toolkit – GEA ..................................................... 4
1.5 Problem Statement/Contributions ........................................................................ 4
1.6 Outline of Thesis ...................................................................................................... 6

## Chapter 2 Background/Related Work
2.1 Gene Expression ......................................................................................................... 8
2.2 Gene Expression Techniques .................................................................................... 8
    2.2.1 Microarray ........................................................................................................ 9
    2.2.2 EST ............................................................................................................... 10
    2.2.3 SAGE .......................................................................................................... 10
2.3 Gene Expression Analysis ......................................................................................... 11
4.3.3 Case 3: Finding Genes always having a Higher/Lower Expression Level in the Cancerous Libraries.................................52
4.3.4 Case 4: Finding Genes Unique in Each Type of Cancer..................................................54
4.3.5 Case 5: Verification with the ENUM Tables...............................55
4.3.6 General Remarks..................................................56

4.4 Highlight Features in the GEA..................................................57
4.4.1 Range Arithmetic..................................................57
4.4.2 The Lineage Feature..................................................60
4.4.3 Manipulation of Top Gaps..................................................62
    4.4.3.1 Calculate Top Gap Table..................................................62
    4.4.3.2 View Top Gap Table..................................................64
4.4.4 Search Operations..................................................65
    4.4.4.1 Expression Analysis Database..................................................65
    4.4.4.2 General Database Searches..................................................67
4.4.5 Checking Features..................................................71
    4.4.5.1 Error Checking..................................................71
    4.4.5.2 Redundancy Check..................................................72
    4.4.5.3 Confirmation Check..................................................72

4.5 Other Features in the GEA..................................................73

4.6 Complications faced during the Implementation..................................................73
4.6.1 How to Store the SAGE data to the Database?..................................................73
4.6.2 Limited Support from JDBC..................................................75
4.6.3 Database Design..................................................75

Chapter 5 Conclusion and Ongoing Work..................................................76
5.1 Conclusion..................................................76
5.2 Ongoing Work: Integrated Genomic Analysis..................................................77
    5.2.1 UNIGENE..................................................78
    5.2.2 SWISSPROT..................................................78
    5.2.3 PFAM..................................................78
    5.2.4 KEGG..................................................79
    5.2.5 GENBANK..................................................79
List of Figures

1.1 GenBank Statistics ................................................................. 3
2.1 The Gene Expression Process ................................................. 9
2.2 The SAGE Method ............................................................... 11
2.3 Cancer Development in Various Tissues ................................. 12
3.1 The Underlying Model of the GEA ......................................... 25
3.2 The Structure of an ENUM table ............................................ 25
3.3 Structure of the Intensional World ........................................ 26
3.4 Gap value Definition: Non-overlap(left) Overlap(right) .............. 29
3.5 An Example: GAP = diff(SUMY_1, SUMY_2) ............................. 29
3.6 An Example: GAP_3 = minus(GAP_1, GAP_2);
    GAP_4 = intersect(GAP_1, GAP_2) ......................................... 30
4.1 Data Cleaning Process .......................................................... 37
4.2 Fascicle vs. Normal: RIBOSOMAL PROTEIN L12 ..................... 40
4.3 Fascicle vs. Normal: ALPHA TUBULIN .................................. 41
4.4 Data Set Based on System-Defined Tissue Type ....................... 43
4.5 Generate Metadata ............................................................. 44
4.6 Calculate Fascicles...............................................................45
4.7 Examine Fascicles...............................................................46
4.8 Purity Check.................................................................47
4.9 Create GAP.................................................................48
4.10 Visualize the Distribution of a Tag........................................49
4.11 Cancerous Libraries Inside vs. Outside of Fascicle: ADP Protein........50
4.12 Create GAP for Cancerous Inside vs. Outside of Fascicle................51
4.13 Cancerous Tissue always has Lower Express Values than Normal.......53
4.14 Brain, Lower Expression in Cancer than Normal..........................55
4.15 Create Customize Data Set for User-defined Tissue Type.................56
4.16 An Example Window on Range Arithmetic..................................58
4.17 Range Arithmetic – Any Tag................................................59
4.18 An Example Window on the Lineage Feature.................................62
4.19 Calculate Top Gap Table..................................................63
4.20 View Top Gap Table.........................................................64
4.21 Search Expression Analysis Database......................................66
4.22 EADB Search.................................................................66
4.23 Search Library Information................................................68
4.24 Search Tissue Type Information.............................................69
4.25 Retrieve Expression Value for Multiple Tags...............................70
4.26 Retrieve Expression Values for Single Tag...................................70
4.27 Error Checking...............................................................71
4.28 Redundancy Check..........................................................72
4.29 Confirmation Check .................................................................73

4.30 (a) Conceptual Structure of the SAGE Data  
     (b) Physical Structure of the SAGE Data ....................................74

AI.1 Microarray .............................................................................. 87

AI.2 SWISS-PROT Database Statistic ..............................................87

AI.3 The Procedures to Create Gene Expression Profile – SAGE .........88

AIII.1 Login Window .......................................................................92

AIII.2 Login Successful .....................................................................93

AIII.3 Login Failed ...........................................................................93

AIII.4 Database Manage .................................................................94

AIII.5 Simple Text Editor ...............................................................95

AIII.6 Browse Window for Open .....................................................96

AIII.7 Editing Functions .................................................................96

AIII.8 Administration Feature .........................................................97

AIII.9 Create New Account ............................................................98

AIII.10 Delete User Account ..........................................................98

AIII.11 Modify User Information ......................................................99

AIII.12 Configuration Window .........................................................100

AIII.13 Help Features ......................................................................101
# List of Tables

2.1 Software for Gene Expression Analysis ........................................ 19

2.2 A Fragment of the SAGE Data ..................................................... 21

3.1 Number of Indices Require to Guarantee $w$ Indices Hit ................. 34

3.2 Time Saving Regard to $w$ Indices Hit ........................................ 34

4.1 Basic Interval Relations .......................................................... 58
Acknowledgements

The author would like to thank Dr. Raymond Ng for his supervision and guidance without which this work could not have been completed. The author would also like to thank Dr. Anne Condon for her time and comments on this thesis work and her suggestions for improvement. The author would also like to thank Ms. Kirsty Barclay and my husband, Calvin who dedicating their time and effort in reviewing the writing.

Jessica Min Phan

The Department of Computer Science
The University of British Columbia
August 2002
To my dearest family
for their support, love, and inspiration
Chapter 1

Introduction

1.1 Data Mining

Data mining is “a process to uncover ‘interesting’, ‘useful’ and unknown data patterns hidden in large data sets” [HK01]. It is also known as Knowledge Discovery from Database (KDD). As data accumulates, it becomes more and more difficult for us to search or even recognize patterns. One of the best solutions to this is to use data mining techniques, which address problems through automated searching, making the process simpler, more manageable, and more efficient. Data mining is applied in many ways, all around us. For example, supermarket managers want to look for patterns in customer purchasing so they can rearrange items in their stores to maximize sales. Stockbrokers can use sequential data mining programs to identify which purchase sequences of stocks have generated the greatest return. Car insurance companies can apply classification software to classify those customers who are dangerous drivers, or those who are not. Credit card companies can identify unusual spending by applying the outlier detection algorithm to catch irregular behavior patterns in data. Medical researchers can analyze patient records to find diseases with common symptoms, and biologists can apply clustering techniques
to group similar tissue samples into clusters and then analyze these clusters to identify candidate genes that cause cancer.

In recent years, the relation of gene expression to cancer development and treatment has received a lot of attention. Many data mining techniques are developed to rapidly monitor, and efficiently analyze large amounts of gene expression data into useful information, provide insight and understanding of the data, and discover unknown patterns. A new interdisciplinary research area, called bioinformatics, has emerged, comprised of researchers from a variety of research areas to exchange and to combine their knowledge, experiences, and technologies, in order to attempt to solve complicated biological problems on a large scale. In this thesis, we will focus on how the data mining of gene expression data affects medical research.

1.2 Bioinformatics

Bioinformatics is the study of information content and flow in biological systems and processes on a large scale; it includes a wide range of subject areas, from structural biology, to genomics, to gene expression studies. Traditionally, biological studies examined individual systems in detail, and sometimes compared them with a few related systems. However, in bioinformatics today, researchers can conduct large-scale global analyses of genomic data across many systems. Recently, biological data are being produced at a phenomenal rate; for example, the GenBank [Ben98] repository of nucleic acid sequences contains 13,602,262 entries. A summary of the growing rate of data in GenBank is shown in Figure 1.1 [GB00]. Also the SWISS-PROT [BA00] database of protein sequences contains 103,258 entries (a statistical graph can be found in Appendix I, Figure AI.2 [SP01]) in December 2001. “On average, these databases are doubling in size every 10 months” [LGG01]. The fast growing, tremendous amount of gene expression data, collected and stored in large and numerous databases, has fast exceeded human ability to comprehend it. Therefore, researchers are encouraged to develop new and powerful data mining tools to help them organize, analyze, and understand these vast amounts of information more effectively.
1.3 Why Analyze the Gene Expression Data

A statistical study conducted by the National Cancer Institute of Canada shows that one hundred and thirty-two thousand new cases of cancer are diagnosed, and sixty-five thousand people die of cancer in Canada each year [NCIC00]. About thirty-eight percent of Canadian residents develop cancer in their lifetimes. To better understand the causes of deadly diseases and discover effective treatments, researchers from areas such as molecular biology, computer science, statistics, and applied mathematics have come together to conduct research in the area of bioinformatics. Researchers believe relating gene expression to disease development and treatment can uncover the cause of these diseases. Molecular biologists focus on producing a huge amount of gene expression data, which may be the key to unlocking the mystery of the causes of disease. However, only a very small fraction of this data is understood today. Researchers from other fields are focusing on developing new algorithms and tools to help biologists analyze and understand this massive amount of gene expression data. Chapter 2 discusses what gene expression and gene expression analysis are.
1.4 Motivation for Introducing the Toolkit -- GEA

In any living cell undergoing a biological process, different subsets of its genes are expressed in different stages of the process. The particular genes expressed at a given stage, and their relative abundance, is crucial to the cell's proper function. Measuring gene expression levels in different developmental stages, different body tissues, different clinical conditions, and different organisms is useful in understanding biological processes. Such information can help to characterize gene function, to determine the effects of experimental treatments, and to understand many other biological processes. In the past, the study of gene expression was basically "local", that is one gene at a time. This approach, though limited, has produced a wealth of biological insights. However, in the past few years, there has been a lot of excitement over attempts to make gene expression analysis more "global", this is, simultaneously analyzing genes in the thousands. The expectation is that such analysis could help to shed light on interactions among genes and their co-regulation. This excitement has been fueled by major breakthroughs in biotechnologies—namely the microarray technique [SSDB95, SSHCBD96, HS97], and the SAGE technique [VZVK95]. The general objective is to try to gain a better understanding of the functions of cellular tissues. In particular, one specific goal is to relate gene expression to cancer diagnosis, prognosis, and treatment. However, a key obstacle is that the availability of analysis tools, or lack thereof, impedes the use of the data, making it difficult for cancer researchers to perform analysis efficiently and effectively.

1.5 Problem Statement/Contributions

Clustering is the most widely adopted paradigm for analyzing gene expression data. The most common methods are hierarchical clustering, self-organizing maps (SOM) [TS99], and k-mean clustering [BFR98]. Examples include the studies conducted by Eisen et al. [ESBB98] and Drawid and Gerstein [DG00] on yeast, Alon et al. on normal and cancerous colon tissues [ABNG99], Caron et al. on normal and cancerous tissues [CSMBR01], and Golub et al. on acute myeloid leukemia and acute lymphoblastic leukemia [GSTHG99]. The underlying paradigm is to cluster genes based on some distance functions and such clustering algorithms as the k-means
algorithm, CLARANS [NH94], BIRCH [ZRL96], CLIQUE [AGGR98], CLICK [SS99], and OPTICS [ABKS99, NSS01]. Whatever the specific algorithms chosen, the aforementioned cluster analysis studies assume that clustering is a one-step process. Moreover, cluster analysis by itself does not help identify candidate genes. Among other objectives, the toolkit we present in this thesis, the Gene Expression Analyzer (GEA), is designed and developed to overcome these two problems. It is important to note that the GEA is not a proposed new clustering algorithm. Rather, it is a toolkit that allows existing clustering algorithms to be applied more effectively to gene expression analysis. The GEA is designed and developed to provide better data mining and analysis support for gene expression data. It makes three key contributions:

1. Amongst all data mining paradigms that have been proposed and studied, clustering is the most widely adopted paradigm for analyzing genomic data. Examples include the studies conducted by Eisen et al. [ESBB98], Alon et al. [ABNG99], Den-Dor. et al. [DSY99], Tamayo et al. [TS99], and Drawid and Gerstein [DG00]. However, all these studies regard cluster analysis as a one-step process. That is, there is the assumption that one needs to apply a clustering algorithm only once to the data in order to get the desired outcome. Unfortunately, real data mining and cluster analysis is rarely a one-step process; it involves the repeated manipulation of data, as well as the results of previous manipulations. The proposed GEA attempts to model this reality by providing a set of operations for cancer/biology researchers to conduct cluster analysis more effectively. Underlying the GEA is an algebraic framework that allows the output of one operation to become the input of another.

2. A key objective for performing cluster analysis on gene expression data is to identify candidate genes for further analysis, including clinical studies in a more traditional laboratory setting. However, traditional clustering techniques do not provide much help in identifying such genes. In contrast, the proposed GEA provides various facilities to accomplish this task. One example is integrated support for finding fascicles, as proposed by Jagadish et al. [JMN99].
There are two general types of data essential to effective gene expression analysis. As discussed above, the first type is the actual gene expression data, for example, data produced by the microarray technique, or the SAGE technique. The second type is auxiliary meta data, such as for mapping tags to genes (UNIGENE), for mapping genes to proteins (SWISSPROT), for classifying proteins into functional families (PFAM), or even for identifying scientific publications studying a particular gene or protein (PUBMED). A tag represents the transcription product of a gene; it is a nucleotide sequence of 10 base pairs. From a database standpoint, the second type of data is more traditional, in that the main operations on the data are search, retrieval, and archival. The first type of data is somewhat different in that the main operation on the data is analysis. Thus, the key challenge here is how to design and develop a system that can support these diverse requirements, preferably on top of a relational DBMS. As summarized above, the GEA allows for mining and analysis on one hand, and it supports search and retrieval operations on the auxiliary databases on the other.

1.6 Outline of Thesis

The thesis is organized as follows. In Chapter 2, we give a brief introduction to the background and related work in gene expression analysis. The model and operation issues of the GEA are presented in Chapter 3. In Chapter 4, we present the GEA system and the results that the GEA is capable of delivering, demonstrated by case studies based on the analysis of the SAGE data set. Last, our conclusions and future work are discussed in Chapter 5.
Chapter 2

Background/Related Work

Bioinformatics is a fairly new research area and has attracted a lot of attention; it includes a wide range of research topics and analyzes many data sources. These data sources can be divided into raw DNA sequences, protein sequences, macromolecular structures, genome sequences, and other whole genome data. Some of the research topics in this area include separating the coding and non-coding regions in the raw DNA sequences, developing multiple sequence alignment algorithms for protein sequences, predicting tertiary structure for macromolecular structure, and so on. Among all these research topics, gene expression analysis has arisen as one of the “hottest” in bioinformatics in the past few years.

In this chapter, we provide the reader with a general understanding of the background and some related work in this research area. For example, we discuss what gene expression is, what gene expression techniques have been developed, and what some of the gene expression analyses are that have been conducted in the past. Among all the techniques that are applied to analyze gene expression, Clustering, a data mining technique, is the most widely adopted paradigm. We are also going to introduce the concepts of the Fascicles algorithm and the 3W model here, which are applied in the GEA.
2.1 Gene Expression

*Gene expression* is a two-step process where DNA sequences are initially transcribed into messenger RNA (mRNA) sequences, and then these in turn are translated into the proteins that perform various cellular functions. Figure 2.1 illustrates the gene expression process. A crucial aspect of proper cell function is the regulation of the gene expression process, so that different cell types express different subsets of genes. The mRNA controls the corresponding amount of protein produced in a cell. This means more copies of mRNA produce larger amounts of protein. In each tissue type, the relative levels of mRNA of each gene are called the *gene expression profile* of the tissue. The gene expression profile is understood to be unique for each particular tissue type. This implies that the majority of the genes in the human genome are only expressed in one tissue type, and only some of the “housekeeping genes” are expressed in all cells, which are used to control the transcription and translation in the gene expression process. Measuring mRNA levels can determine the subset of genes expressed in different cell types under different conditions. Measuring gene expression levels under different conditions is important for expanding our understanding of gene function, how various gene products interact, and how experimental treatments can affect cellular function.

2.2 Gene Expression Techniques

“A most recent source of genomic scale data has been from expression experiments, which quantify the expression levels of individual genes” [HCHRLO00]. Gene expression measures the amount of mRNA that is produced by the cell in a particular state, and the pattern of genes expressed in a cell is characteristic of its current state. Virtually all the differences in cell state or type are correlated with the change in mRNA levels of many genes. To generate gene expression data, five different gene expression techniques are developed. They are microarray, Serial Analysis of Gene Expression (SAGE), Expressed Sequence Tags (EST), Subtractive Cloning, and Differential Display; however, only the first three techniques are widely applied by researchers today.
1) All the cells in a person’s body contain all of that person’s DNA

2) The cells of different tissues in the body use different parts of the DNA

3) The parts of the DNA that a cell is using are copied into RNA

4) The information in the RNA is translated to make proteins the cell needs to function. More copies of RNA means that more proteins will be made

**Figure 2.1: The Gene Expression Process**

### 2.2.1 Microarray

The microarray measures the relative levels of mRNA abundance between different tissue samples. The basic technology is generate DNA complementary to gene of interest and laid out in microscopic quantities on solid surfaces at defined positions. Then DNA from samples is eluted over the surface (complementary DNA binds). The presence of bound DNA is detected by fluorescence following laser excitation. The resulting data in a microarray chip can be easily express as tags with expression values, which is similar to SAGE data. An advantage of this method is that it is relatively inexpensive to produce; however, a major disadvantage is that the experimenter must select the mRNA sequences to be detected in a sample, and the sequence useful for cancer profiling may not be known in the first place. The result of the method is affected by the experimenter’s bias. An image of a microarray chip can be found in Appendix I,
2.2.2 EST

EST stands for Expressed Sequence Tags. This method collects gene expression data in complete segments of mRNA, which is different from the SAGE data. The EST sequence claims to be more distinctive than the SAGE sequence because each of the EST sequences range from tens to hundreds of base pairs compare to SAGE tags only have 10 base pairs. However, only a small amount of data is available due to the expensive cost of the procedure. ESTs have been applied in the discovery of new human genes and the mapping of the human genome.

2.2.3 SAGE

SAGE, which stands for Serial Analysis of Gene Expression, is a technique to obtain a quantitative profile of cellular gene expression. It measures the absolute count of the mRNA in each sample. At present, the SAGE data in the NCBI website consist of 100 libraries, with each library having about 1,000 to 32,000 tags derived from a single expression profiling experiment. A tag represents the transcription product of a gene; it is a nucleotide sequence of 10 base pairs (which is a sequence of the 4 bases A, C, G, and T). That is, a tag corresponds to one gene at the most, but there are tags with no known corresponding genes. The data product of the SAGE technique is a list of tags with their corresponding count values, which are the expression levels of the tags, called a SAGE library. The 100 SAGE libraries are derived from various tissue types including the brain, breast, prostate, ovary, colon, pancreas, vascular, skin, and kidney. For each tissue type, there are libraries from both cancerous tissue and normal tissue. The libraries are made from either cell line or bulk tissue. Cell line means a cell that can grow indefinitely in a test tube or petri dish, making an infinite number of copies of itself. Bulk tissue means a group of cells, which were taken directly out of tissue in a person’s body. The number of unique tags in a library is the number of different tags detected in the sample. The total number of tags is the sum of all the count values of all the tags in a library. The SAGE data set is publicly available at
http://www.ncbi.nlm.nih.gov/SAGE. Figure 2.2 illustrates how the gene expression profiles are generated for SAGE. A diagram explaining the SAGE method in more detail can be found in Appendix I, Figure AI3 [SAGE].

One of the advantages of using the SAGE method is that the results avoid the bias of the experimenter, which gives all the mRNA in a tissue sample an equal chance of being copied and sequenced. Moreover, the complete sequence of the mRNAs profiled does not have to be known in advance. Furthermore, SAGE is claimed to be the only method that collects complete gene expression data in a practical way, and it is highly scalable. However, the SAGE data is very expensive to produce, and this method may suffer sequencing and sampling errors [SUUB00].

1) Obtain samples of both cancerous and normal tissue e.g. brain tumour and normal brain
2) Isolate the RNA from the samples
3) Count how many copies there are of each type of RNA. This is the sample's GENE EXPRESSION PROFILE

Figure 2.2: The SAGE Method

2.3 Gene Expression Analysis

Researchers around the globe invest enormous effort in generating gene expression data. The data is believed to contain a wealth of information that may possibly unlock the secrets of many biological mysteries. By analyzing gene expression data, researchers may be able to identify novel genes, discover potential drug targets, or understand gene function.

Cancers are usually classified and treated based on which tissue they develop in; an example of this is illustrated in Figure 2.3. However, different people with the same type of cancer often
respond differently to treatment, and one treatment can be effective for cancer originating in different tissues. A possible explanation is that cancers which appear to be similar on a macroscopic level, may in fact be very different at the sub-cellular level. One way to detect these differences is through the gene expression profile analysis.

Normal cells can evolve into malicious cancer cells through a series of mutations in genes. Cancer cells are also genetically different from normal cells and indicate altered gene expression. For example, the proteins that prevent uncontrolled cell growth and promote cell death, are no longer produced. This knowledge of the changes in gene expression for certain types and stages of cancer provides insights to cancer development and progression. Ideally, if the gene expression profile of diseased tissue is known, it can be used to determine what has been altered and provide hints on how to fix it. Analyzing gene expression profiles also allows us to identify genes that are expressed at significantly higher or lower levels in cancer than normal tissues. These genes will be analyzed in detail to understand their functions; as a result, we may able to determine the possible causes of cancer and identify candidate genes for cancer diagnosis.

The two main approaches used in gene expression analysis are hypothesis testing and knowledge discovery. Hypothesis testing is mainly used to investigate whether the induction of a biological process leads to predicted results. In this thesis, we focus our discussion on the approach of "knowledge discovery" to identify the differences between healthy and diseased cells. The two
main methodologies applied in knowledge discovery in gene expression data are data mining techniques [AGGR98, ABKS99, BFR98, NH94, NSS01, SS99, TS99, ZRL96], and visualization methods [SGHN01, CCWSC98]. Visualization methods are best suited for where the patterns of interest are clear in advance; however, it does not scale well for larger data sets and it is less appropriate for discovering unexpected patterns. Therefore, many researchers employ data mining techniques to analyze gene expression data. Applying data mining technique to gene expression data could identify “interesting” and unknown patterns, and relationships between multiple gene expression profiles.

Although both microarray and SAGE techniques are widely used to produce gene expression data, currently, most gene expression analysis is based on microarray data. Although SAGE data is publicly available, only a few simple analyses have been conducted. Most of the analyses of SAGE data are based on one library or the comparison of two libraries at a time; therefore, this gives many people the incorrect impression that these are data mining. Real data mining techniques allow data analysis on a large scale, and are not a one step process. In the following sections, we discuss some of the gene expression analyses that have been conducted on microarray and SAGE data, and the common clustering algorithms applied in these analyses.

2.3.1 Common Clustering Algorithms

Many gene expression studies have so far focused on developing methods to cluster tissues using similarities in gene expression profiles. The most common methods for analyzing gene expression profiles are hierarchical clustering, self-organizing maps, and k-means clustering. The hierarchical methods group genes in a “bottom-up” fashion. That is, genes with the most similar expression profiles are clustered first, then those with more diverse profiles are included iteratively. In contrast, self-organizing map and k-means clustering methods employ a “top-down” approach, in which the user pre-defines the number of clusters for the data set. The clusters are initially assigned randomly and the genes are regrouped iteratively until they are optimally clustered.
2.3.2 Gene Expression Analyses on Microarray Data

Eisen et al. is the first to propose the use of a clustering technique to analyze the massive amount of gene expression data. In [ESBB98], the expression data analyzed are the microarrays of a budding yeast genome and a human fibroblast cell line. The hierarchical pairwise average-linkage cluster algorithm is applied, and the standard correlation coefficient is used for the distance measurement. The author applies the clustering analysis on the genes, not on the samples. He concludes that the genes with high degrees of sequence identity are clustered next to each other. Genes with similar function, but different sequences also cluster tightly together. Moreover, co-expressed genes form clusters. Although the author presents many valuable observations, it is unclear to the reader how the clusters are formed; it seems as though the clusters are determined by observing the graphical representation of the results. Since the result is presented in a colored image, if the number of genes (rows) and conditions (columns) increases, it becomes impossible to correctly measure the clusters.

In [ABNG99], Alon et al. employed a two-way clustering method and used the correlation coefficient as a distance function to analyze the microarray data. The testing data consists of 40 cancerous colon bulk tissues, 22 normal colon bulk tissues, and some cancerous colon cell line tissues. Their analysis showed that this clustering technique successfully divides a set of colon tissues into two groups, one containing mostly cancerous tissues, and the other containing mostly normal tissues. The author also identifies some genes expressed at a relatively lower level than normal in cancerous tissues. This clustering method can also separate the cancerous cell lines into a cluster and distinguish them from cancerous bulk tissues, with the cancerous cell lines placed closer to the cancerous bulk tissue than the normal bulk tissues. This paper provides a novel approach to clustering both genes and tissue samples for the first time, and the results provide some very interesting insights into this field. The visualization of the result requires improvement because it will be difficult to observe when the number of samples increases.

Golub et al. develops a generic approach to classify leukemia by reliably using the computer analysis of microarray data. In [GSTHG99], Golub analyzes a test data set consisting of 27 acute lymphoblastic leukemia (ALL) and 11 acute myeloid leukemia (AML). The SOM technique is
applied to cluster these cancerous tissues; the SOM is particularly well suited to identifying a
small number of prominent classes in a small data set. The result shows that the SOM
effectively separates the data set into two classes; class A contains mostly ALL, and class B
contains mostly AML; however, it is not 100 percent accurate. The author mentions that the
SOM is unable to search for finer subclasses of the ALL due to the insufficient number of tissue
samples available. The author also highlights a dangerous factor for class prediction caused by
the sample contamination, which can result in identifying classes reflecting the proportion of
contamination in the samples (especially in the case of a solid cancer), rather than the underlying
cancer biology. Since this approach does not require prior biological knowledge of the diseases,
it may provide a generic strategy for classifying all types of cancer, which can reduce
dependence on expensive laboratory tests.

Perou et al. in [PJRRE99] applies the same algorithm and software described by Eisen et al. in
[ESBB98] to identify the patterns of gene expression in breast tissues. The author points out that
studying breast cancer is very complicated because it is a solid cancer that consists of many
different cell types, and breast cells are genetically diverse. They found some interesting clusters
of genes that are co-regulated. For example, they observed that a set of genes was more highly
expressed in a subset of cancerous tissues than in normal tissues.

A novel clustering algorithm is proposed by Ben-Dor et al. in [DSY99] to cluster and analyze
gene expression data. The algorithm is based on the stochastic model and this clustering
heuristic is called Cluster Affinity Search Technique (CAST). The author applies the CAST
algorithm to verify some experiments done by others in the previous years, such as Wen et al.
[WFMCS98], Kim et al. [KIM], and Alon et al. [ABNG99]. The author claims that CAST
returns superior results, which determine the nature groups of co-regulated genes and group
tissues with similar expression patterns into clusters. However, it is unclear to the reader how to
apply CAST to determine the clusters. Since the CAST algorithm can determine the cluster
boundaries without human intervention, the author claims that CAST is a more advanced
algorithm than others. Also, the number of clusters is determined by the algorithm, instead of
being a constant, given as an input parameter to the program. Some visualization tools are
provided to the user to aid understanding the results as well.
2.3.3 Gene Expression Analyses on SAGE Data

Most cancer research over the past 50 years has been devoted to the analyses of genes that are expressed differently in cancer cells than in their counterparts. No comprehensive study of gene expression in cancer cells, compared to normal cells, has been reported. After the SAGE method was introduced in 1995, Zhang et al. [ZZVHK97] presented a way to understand the complex differences between normal and cancer cells in humans by comparing the gene expression profiles created by the SAGE method. The author conducts a simple experiment comparing the gene expression patterns of normal colon epithelium and primary colon cancers. The results reveal that most genes are expressed at similar levels; however, some genes are expressed significantly differently. This new approach explores the SAGE data to understand the complex differences between normal and cancer cells in humans and provides us with many important observations for future experiments. However, their experiment is based on only a few tissue samples; it may fail to provide a global overview of the data. Moreover, the genes examined are “arbitrarily” chosen by the experimenter, which does not preclude bias.

A SAGE web site was built in 1999 at the National Center for Biotechnology Information (NCBI) [NCBI], which is part of the Cancer Genome Anatomy Project (CGAP) [SBEK00], dedicated to collecting data on the genetics of cancer. The purpose of the SAGE web site is to collect all generated SAGE data; therefore, the researchers can benefit from the SAGE technique without having to suffer the expense of creating all of the data themselves. The web site comprises the SAGE libraries created by various laboratories and makes these libraries available to the public to view and download. A tool for analyzing the SAGE libraries offered on this web site is called the xProfiler. The xProfiler is designed for differential-type analyses, for pooling and comparing SAGE libraries. The user can place similar libraries into one of the two groups based on their characteristics (e.g., normal colon, or cancerous colon). Comparisons are then made between the two groups using a statistical test developed specifically for SAGE data. The disadvantage of the xProfiler is that the user has to guess which SAGE libraries should form a group, and which two groups should be compared, in order to return meaningful results. Other tools included in the SAGE web site are the tag-to-gene mapper and the gene-to-tag mapper, which are elaborated on Chapter 4. The CGAP project also provides analysis tools; however, all
the tools can analyze only one library, or compare only two libraries at a time. None of these tools can analyze data on a large scale.

The chromosomal position of human genes is rapidly being established. Caron et al. in [CSMBR01] presents a new method to determine clusters by clustering highly expressed genes in chromosomal domains. The algorithm is developed to assign these SAGE tags to UniGene clusters and their chromosomal positions. The SAGE libraries being analyzed include five normal tissue and seven cancerous tissue samples. The resulting Human Transcriptome Map [HTM] generates gene expression profiles for any chromosomal region in the twelve tissue types and this map reveals clusters of highly expressed genes in the specific chromosomal regions. As the results show, the average expression levels per gene are up to seven times lower than the genomic average in RIDGEs. RIDGEs (regions of increased gene expression) represent these clusters of highly expressed genes. The author claims that the Human Transcriptome Map provides a tool to search for genes that are over/under expressed in cancer, but no details are given to prove his claims. The RIDGEs are very easy to detect by observing the Human Transcriptome Map; however, the authors did not explain how the RIDGEs are calculated due to their complexity.

A most recent paper by Ng et al. [NSS01] applies a hierarchical clustering algorithm called OPTICS and used the correlation coefficient as a distance function to analyze the SAGE data. Their goals are to detect the differences and similarities between cancerous and normal tissue types. Also, they particularly want to understand whether different cancerous tissues cluster together, and whether samples of the cancer type form sub-clusters. The authors realized that clustering is heavily influenced by noises in the SAGE data, so they propose four pre-processing steps to deal with this problem, which remove the sequence errors and make the data more complete. The clusters found in the “cleaned” data are significantly improved compared to the clusters determined in the raw data. The results show that most of the clusters consist of just one tissue type. The OPTICS algorithm produces a promising hierarchical clustering structure in which the SAGE libraries are grouped according to tissue type and neoplastic state. However, the author mentions that no sub-clusters are discovered due to the insufficient number of libraries available. This paper brings our attention to the importance of pre-processing procedures; for
clustering analysis to achieve its potential, proper filtering of the data is necessary. One thing not clear to the reader is that how they detect differences and similarities between the clusters.

2.4 Software for Gene Expression Analysis

Most of the software for gene expression analysis extracts useful information from the expression of microarray experiments. The analysis can be subdivided into three stages, but none of the software packages carry out all three stages. The three stages include image quantitation, data visualization, and clustering analysis. The image quantitation provides services such as quantifying the individual fluorescence spot intensities in a microarray chip, outputting data to a file in a given format, and other steps to create the microarray data. A summary of the software available and its highlights is illustrated in Table 2.1.

Most of the software is not publicly available and only some of it is in demo form for researchers to use for a trial period. Software with similar features to the GEA includes Expressionist and Gene Spring, which provide clustering analysis and some data visualization tools. Both Expressionist and Gene Spring are designed primarily to analyze microarray data, but the GEA has a more general design that can analyze both SAGE data and microarray data. Expressionist and Gene Spring provide more comprehensive visualization tools than the GEA; however, as mentioned in section 2.4, data visualization methods do not scale well. The availability is very limited for the software; Expressionist requires a very expensive license fee and the price for owning a copy of the Gene Spring is unknown. The Expressionist and Gene Spring apply k-mean and hierarchical clustering techniques to analyze data; on the other hand, the GEA aims to allow users to incorporate any clustering techniques to analyze data. Due to the lack of availability of the software, no specific details of comparisons between them and the GEA are available.

The emphasis of the SportFire Pro is different from the GEA because it is more focused on data visualization than clustering analysis. In general, SpotFire Pro provides a very good visualization tool, and only a simple k-mean algorithm is included for clustering analysis. As
for Array Vision and GeneTAC, they are focused on data quantitation and data visualization, rather than clustering analysis.

<table>
<thead>
<tr>
<th>Software</th>
<th>Company</th>
<th>Category</th>
<th>Highlights</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expressionist</td>
<td>GeneData AG</td>
<td>Data Visualization</td>
<td>Hierarchical clustering, Intuitive data visualization interface, Very expensive</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Clustering Analysis</td>
<td></td>
</tr>
<tr>
<td>Gene Spring v1.2</td>
<td>Silicon Genetics</td>
<td>Data visualization</td>
<td>“Dedicated” to microarray analysis, Hierarchical, k-mean clustering, Designed for large data set</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Cluster Analysis</td>
<td></td>
</tr>
<tr>
<td>SpotFire Pro</td>
<td>SpotFire AB</td>
<td>Visual data mining</td>
<td>General-purpose visualization program, k-mean clustering analysis, Good visualization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Cluster analysis)</td>
<td></td>
</tr>
<tr>
<td>Array Vision v4.0</td>
<td>Imaging Research Inc</td>
<td>Data visualization</td>
<td>Complex interface with steep learning curve, Poor auto-alignment of grids limits high data throughput, Erroneous background determination</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Image quantitation data</td>
<td></td>
</tr>
<tr>
<td>GeneTAC</td>
<td>Genomic Solutions Inc. (GSI)</td>
<td>Data visualization</td>
<td>Analysis tools dedicated to GSI microarray package, therefore, more effort required to analyze non-GSI arrays, No visualization tools</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Image quantitation data</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.1: Software for Gene Expression Analysis
2.5 Clustering Algorithm in the GEA

After a detailed discussion of the gene expression analysis conducted in the past, we can conclude that many gene expression studies emphasized on clustering techniques. The clustering techniques can be used to divide a set of tissue samples into clusters based on their expression profiles and to identify sets of genes that "behave similarly" across conditions. In fact, clustering has been demonstrated to identify functionally related families of genes [DSY99, ESBB98].

When it comes to clustering the gene expression data (microarray or SAGE data), one key complication is the high dimensionality of the data. With each tag corresponding to one dimension, there are more than 60,000 dimensions. Most traditional clustering techniques cannot scale to provide clustering to these extremely high numbers of dimensional data. While some traditional clustering techniques can be used to provide clustering as mentioned above, they cannot be used to identify candidate genes, which may be much smaller in number. To accomplish the latter task, the Fascicle algorithm is chosen to be included in the implementation of the GEA. Note that the model presented in section 2.6 can support clusters produced by other clustering algorithms just as well. But for the sake of concreteness in our ensuing discussion, we assume clusters are produced by the Fascicle algorithm. In this thesis, we apply Fascicles a clustering algorithm to analyze gene expression data for the first time. We briefly introduce the Fascicle algorithm next.

2.5.1 Fascicles Algorithm

The nature of the Fascicles algorithm is well suited for analyzing gene expression data. It takes two parameters as input: k, the desired number of compact tags, and t, a tolerance vector that determines whether a particular tag is compact or not. A fascicle is a collection of libraries. One of the main concepts is the compact tag. For a tag to be compact within a fascicle, all the libraries in the fascicle must more or less agree on the value of that tag—within the tolerance
specified for that tag. For example, suppose that for tag G, the range of values for all the libraries in a fascicle is between 1 and 4. If the specified tolerance for tag G is at least 3, then tag G is called compact. Otherwise, G is not a compact tag. Given the input parameters, the fascicle algorithm finds fascicles that meet tolerance t and that have at least k compact tags. Note that not only can the algorithm identify libraries contained in a fascicle for some given set of k compact tags, but it can also identify simultaneously (numerous) sets of k compact tags. While details of the algorithm can be found in [JMN99] (and an improved version found in [BGR01]), the relevance here is that if, for example, a fascicle consists of only cancerous tissues, then the k compact tags collectively form a “signature” of the cancerous tissues. That is to say, all of the libraries in the fascicle have very similar levels for the compact tags. By removing the non-compact tags in a fascicle, the cancerous libraries in the fascicle have an even clearer identity, and the compact tags may be good candidates warranting further laboratory research.

To illustrate the concept of the fascicle, a simple example is given based on Table 2.2. Suppose the compactness tolerance imposed on the attributes is: $t_{AAAAA} = 120$, $t_{AAAAAAC} = 3$, $t_{AAAAAAT} = 47$, $t_{AAAAACTCC} = 60$, $t_{AAAAAGAAA} = 20$. Then libraries SAGE_BB542_whitematter, SAGE_Duke_cerebellum, and SAGE_Duke_H1020 are in a 5-D fascicle with AAAAAAAAAA, AAAAAAAAAC, AAAAAAAAAT, AAAAAACTCC, and AAAAAACAAA as the compact attributes. Different fascicles may have different number and set of compact attributes.

<table>
<thead>
<tr>
<th>Library/Tag</th>
<th>AAAAAAAAAA</th>
<th>AAAAAAAAAC</th>
<th>AAAAAAAAAT</th>
<th>AAAAAACTCC</th>
<th>AAAAAAGAAA</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAGE_BB542_whitematter</td>
<td>1843</td>
<td>3</td>
<td>10</td>
<td>15</td>
<td>11</td>
</tr>
<tr>
<td>SAGE_Duke_1273</td>
<td>1418</td>
<td>7</td>
<td>0</td>
<td>30</td>
<td>12</td>
</tr>
<tr>
<td>SAGE_Duke_757</td>
<td>1251</td>
<td>18</td>
<td>0</td>
<td>33</td>
<td>20</td>
</tr>
<tr>
<td>SAGE_Duke_cerebellum</td>
<td>1800</td>
<td>0</td>
<td>58</td>
<td>40</td>
<td>20</td>
</tr>
<tr>
<td>SAGE_Duke_GBM_H1110</td>
<td>1050</td>
<td>25</td>
<td>1</td>
<td>60</td>
<td>15</td>
</tr>
<tr>
<td>SAGE_Duke_H1020</td>
<td>1910</td>
<td>1</td>
<td>17</td>
<td>74</td>
<td>30</td>
</tr>
<tr>
<td>SAGE_95_259</td>
<td>503</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>456</td>
</tr>
<tr>
<td>SAGE_95_260</td>
<td>364</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>222</td>
</tr>
<tr>
<td>SAGE_Br_N</td>
<td>65</td>
<td>5</td>
<td>79</td>
<td>9</td>
<td>300</td>
</tr>
<tr>
<td>SAGE_DCIS</td>
<td>847</td>
<td>4</td>
<td>124</td>
<td>0</td>
<td>500</td>
</tr>
</tbody>
</table>

Table 2.2: A Fragment of the SAGE Date

21
2.6 The General Structure of the GEA (3W)

We designed and implemented a toolkit to help biology/cancer researchers to perform gene expression analysis more efficiently and effectively. The design of the general structure of the GEA toolkit is based on the 3W model [JLNO00] proposed by Johnson et al. that intends to act on the fact that data mining is often a multi-step process. The 3W stands for the three worlds for data mining in general, defined as the intensional world, the extensional world, and the data world. The intensional world is where most of the important analysis processes take place. The region of the intensional world is represented in the form of descriptions of its members. The extensional world is represented by an explicit enumeration of the tuples belonging to each region of the intensional world. The algebra used in the extensional world is relational algebra extended with aggregation, with some modifications. The data world consists of a relational database and applies relational algebra for data manipulation.

For this kind of analysis to be supported here, the framework underlying the proposed GEA is a specialization of the general 3W model in at least three key respects. First, there need only be two worlds, not three, which merge the extensional world and the data world. Second, the structures are tailor-made for the GEA, namely the SUMY and GAP structures, to be introduced in Chapter 3. Finally, there are additional operators, such as \texttt{diff()}, that are included. Chapter 3 elaborates on these aspects.

2.7 Summary

Clustering is the most widely adopted paradigm for analyzing gene expression data. Examples include the studies conducted by Eisen et al. on yeast, Alon et al. on normal and cancerous colon tissues, Golub et al. on leukemia, Perou et al. on breast cancer, and Ng et al. on SAGE. The underlying paradigm is to cluster genes based on some distance functions. Whatever the specific algorithms chosen, the aforementioned cluster analysis studies assume that clustering is a one-step process. Moreover, cluster analysis on its own does not help to identify candidate genes.
Currently, the availability of software tools for gene expression analysis is very limited. Some of them provide very limited and restricted functionality. Some software provides extensive visualization tools, but requires a very steep learning curve. Most of the software mentioned above is not publicly available and requires a very expensive license fee.

Biologists/cancer researchers desperately need better tools to help them analyze the massive volumes of gene expression data. The Gene Expression Analyzer toolkit that we present in Chapter 3 and Chapter 4 overcomes these problems by providing a flexible analysis environment to reflect that data mining is a multi-step process. The clustering algorithm included in the GEA is perfectly suited for gene expression data. It also provides a set of tools to determine candidate genes, and performs a rich variety of search and data manipulations.
Chapter 3

The GEA: Model
And Operational Issues

The Gene Expression Analyzer is designed based on the three world model, as mentioned in Chapter 2, which reflects the fact that data mining is often a multi-step process. The GEA not only provides the user with a flexible analysis environment, but also provides a structure that can be extended to include other database and cluster algorithms. Below, we first introduce the structures in the two worlds. In section 3.2, we show how these structures can be manipulated. Efficiency issues are also addressed in section 3.3. In Chapter 4, we give comprehensive examples of these manipulations and illustrate the significance of the concepts and structures introduced here.

3.1 The GEA Model and Structures
Figure 3.1 shows the model underlying the GEA, within which gene expression data and intermediate clusters/fascicles can take on dual identity. While in the extensional world, a cluster is represented by an explicit enumeration of all the libraries contained in that cluster. Alternatively, while in the intensional world, a cluster is represented by a definition (that is, a set of conditions) that is satisfied by all the libraries contained in the cluster. This is the key to capturing the multi-step nature of a cluster analysis process, which often involves manipulations of previous intermediate results/fascicles.

![Figure 3.1: The Underlying Model of the GEA](image)

3.1.1 Structures in the Extensional World

A cluster in the extensional world is represented as a relation. In particular, we call such a relation an ENUM table (as in “enumeration”), with its structure shown in Figure 3.2. The columns represent the (compact) tags, and the rows correspond to the libraries. For instance, if cluster/fascicle A consists of the 1st, 2nd, 3rd and 4th libraries, then there are four rows in the corresponding ENUM table, with the columns representing the compact tags of the fascicle. To continue, the ENUM table for another cluster, B, may have a different number and a different set of columns (because the sets of compact tags are different), as well as a different number of rows (even though a library may be included in multiple clusters).

<table>
<thead>
<tr>
<th>Library Name</th>
<th>Tag&lt;sub&gt;1&lt;/sub&gt;</th>
<th>…</th>
<th>…</th>
<th>Tag&lt;sub&gt;n&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Library&lt;sub&gt;1&lt;/sub&gt;</td>
<td>expr_level&lt;sub&gt;1,1&lt;/sub&gt;</td>
<td>…</td>
<td>…</td>
<td>expr_level&lt;sub&gt;1,n&lt;/sub&gt;</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
<td></td>
</tr>
<tr>
<td>Library&lt;sub&gt;m&lt;/sub&gt;</td>
<td>expr_level&lt;sub&gt;m,1&lt;/sub&gt;</td>
<td>…</td>
<td>…</td>
<td>expr_level&lt;sub&gt;m,n&lt;/sub&gt;</td>
</tr>
</tbody>
</table>

![Figure 3.2: The Structure of an ENUM table](image)
Because an ENUM table is nothing more than an instance of an ordinary relation, the extensional world also supports any other ordinary relation. More specifically, the original SAGE data set can be stored in an ENUM table, as shown in Figure 3.2, as a "degenerate" cluster. And there may be auxiliary data associated with each library, such as the tissue type, cancerous or normal, the number of tags, and so forth. These pieces of auxiliary data are stored in ordinary relations; see section 5.1 for a more comprehensive discussion.

3.1.2 Structures in the Intensional World

 Whereas, in the extensional world, a cluster is represented by an explicit enumeration of all the libraries it contains, in the intensional world, a cluster is represented by its definition—the set of compact tags and their ranges. This leads to the SUMY table (as in "summary") shown in Figure 3.3(a). The rows correspond to the compact tags of the cluster. The columns give the range, the mean and the standard deviation of the mRNA expression levels for that tag in the cluster. Additional columns, such as those representing other aggregate values, may also be included.

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Range</th>
<th>Average</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag₁</td>
<td>[min₁, max₁]</td>
<td>avg-level₁</td>
<td>dev-level₁</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Tagₙ</td>
<td>[minₙ, maxₙ]</td>
<td>avg-levelₙ</td>
<td>dev-levelₙ</td>
</tr>
</tbody>
</table>

(a) A SUMY Table

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag₁</td>
<td>avg-level₁</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Tagₙ</td>
<td>avg-levelₙ</td>
</tr>
</tbody>
</table>

(b) A GAP Table

Figure 3.3: Structure of the Intensional World

In the intensional world, apart from the SUMY table, there is another structure called the GAP table. As shown in section 3.2.2, a Gap table is used to summarize the "difference" between two
SUMY tables. This is particularly useful for distinguishing candidate genes that may have different levels of expression in different clusters.

3.2 Algebraic Operations of the GEA

So far, we have introduced the basic structures within the GEA. Recall that one key design objective is to ensure that the retrieval, archival and analysis needs of gene expression analysis can be supported simultaneously. Thus, it is not an accident that the basic structures are all in relational form. The key issue is how these structures can be manipulated, with an emphasis on algebraic operation beyond the standard ones.

3.2.1 Moving between the Worlds

Let us first consider the primary operation of data mining. In the general case, the mine() operator takes a data set from the extensional world and produces an intermediate result represented in the intensional world. Specifically for this thesis, the mining operation is cluster/fascicle production, with the input being the SAGE data set (or any ENUM table) and the output being a cluster represented in a SUMY table. This is shown in Figure 3.1. Note that in the general case, the mining operation can be something other than fascicle production. Examples include other clustering operations, but for simplicity, we focus only on fascicles.

Recall that a SUMY table only stores the definition of the cluster. To obtain an explicit enumeration of all the libraries satisfying the definition, the populate() operator takes a data set (or any ENUM table) and a SUMY table, and finds all rows in the input data set satisfying the conditions laid out in the SUMY table. In other words, the populate operator converts a cluster from its intensional/SUMY form to its extensional/ENUM form, with respect to a given data set.
Finally, to complete the discussion of the operators shown in Figure 3.1, the aggregate() operator can be regarded as an inverse operation to the populate operator. It converts a cluster from the extensional/ENUM form to the intensional/SUMY form.

Recall that there are two key objectives of the GEA: (I) to support the multi-step nature of cluster/fascicle analysis; and (ii) to help identify candidate genes for further clinical analysis. So far, we have seen how the GEA achieves the first objective. That is, it uses algebraic operators to manipulate the data and the cluster. Output from one operator can be input for another operator. With respect to the second objective, so far, fascicle production is the only avenue, because typically the number of compact tags in a fascicle is smaller than the original number of tags. However, the number of compact tags can still be too large (e.g., tens of thousands) for further clinical analysis. This motivates some of the operations presented next for manipulating the SUMY and GAP tables in the intensional world.

3.2.2 From SUMY Tables to GAP Tables

One key reason for identifying candidate genes is to capture the differences between libraries in different tissue types and between libraries in the same tissue type, but in different categories (e.g. cancerous or normal). This comparative analysis provides opportunities to discover the potentially interesting information between the libraries. Within the GEA, the key operation in this regard is the \texttt{diff()} operator that takes two SUMY tables to produce one GAP table, that is, \( \text{GAP} = \text{diff}(\text{SUMY}_1, \text{SUMY}_2) \). For instance, the first SUMY table may correspond to a cluster containing only cancerous breast tissues, whereas the second SUMY table corresponds to normal breast tissues.

Many definitions of a gap value/level are acceptable. In the sequel, we use the following definition that appears to be meaningful in practice. For each tag common in both SUMY tables, the gap level is defined as: gap value = \((\mu_{(hi)} - \sigma_{(hi)}) - (\mu_{(lo)} + \sigma_{(lo)})\), where \(\mu_{(hi)}\) and \(\sigma_{(hi)}\) represent the average and the standard deviation of the tag in the SUMY table having a higher average, and \(\mu_{(lo)}\) and \(\sigma_{(lo)}\) are the counterparts for the SUMY table having the lower average.

Figure 4.4 shows the two cases depending on whether there is overlap. In the first case, for the
given tag, there is no overlap, and the gap value is positive if the first SUMY table has a higher average, negative otherwise. In the second case, there is overlap, and the gap value is defined to be null.

![Diagram of SUMY tables and gap values]

Figure 3.4: Gap value Definition: Non-overlap (left) Overlap (right)

Let us consider the situation of the following two SUMY tables shown in Figure 3.5. First, the resultant table GAP consists of rows corresponding to Tag1, Tag3, and Tag4, because these are the common tags between the two SUMY tables. For Tag1, the gap value is $(7-1) - (5+0) = 1$. But the sign of the gap level is negative because the first SUMY1 table has a lower average. For Tag3, there is an overlap and the gap level is null. Finally, for Tag4, the gap level is $(10 - 4) - (3+1) = 2$, with a positive sign.

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Range</th>
<th>Average</th>
<th>Standard D.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag1</td>
<td>[5, 5]</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>Tag2</td>
<td>[0, 7]</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Tag3</td>
<td>[10, 120]</td>
<td>70</td>
<td>15</td>
</tr>
<tr>
<td>Tag4</td>
<td>[0, 20]</td>
<td>10</td>
<td>4</td>
</tr>
</tbody>
</table>

(a) Table SUMY1

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Range</th>
<th>Average</th>
<th>Standard D.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag1</td>
<td>[0, 14]</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>Tag3</td>
<td>[10, 130]</td>
<td>60</td>
<td>25</td>
</tr>
<tr>
<td>Tag4</td>
<td>[0, 12]</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Tag5</td>
<td>[0, 50]</td>
<td>20</td>
<td>15</td>
</tr>
</tbody>
</table>

(b) Table SUMY2

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag1</td>
<td>-1</td>
</tr>
<tr>
<td>Tag3</td>
<td>Null</td>
</tr>
<tr>
<td>Tag4</td>
<td>+2</td>
</tr>
</tbody>
</table>

(c) Table GAP

Figure 3.5 An Example: GAP = \textit{diff}(SUMY1, SUMY2)
3.2.3 Other Operations in the Intensional World

For identifying candidate tags, a GAP table may still have too many rows. To further reduce the number of tags, the selection operator becomes useful. It takes as input a GAP table and produces a GAP table satisfying the specified selection conditions, as usually done in relational algebra. Just as easily, the selection operator can be applied to a SUMY table to produce another SUMY table. In this case, because a SUMY table may contain range values (e.g., [min, max]), range selection conditions such as overlap are allowed. As a preview to the discussion in Chapter 4, range arithmetic is supported in the GEA.

There are set operations allowed in the intensional world of the GEA as well. These operations apply to either a pair of GAP or a pair of SUMY tables. The intent is to manipulate at the level of tags. For example, the set minus operator extracts the tags appearing in the first table, but missing in the second table. Table GAP3 in Figure 3.6 shows a simple example.

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag1</td>
<td>-11</td>
</tr>
<tr>
<td>Tag2</td>
<td>2</td>
</tr>
<tr>
<td>Tag3</td>
<td>NULL</td>
</tr>
<tr>
<td>Tag4</td>
<td>5</td>
</tr>
</tbody>
</table>

(a) Table GAP1

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag1</td>
<td>-8</td>
</tr>
<tr>
<td>Tag3</td>
<td>9</td>
</tr>
<tr>
<td>Tag4</td>
<td>10</td>
</tr>
<tr>
<td>Tag5</td>
<td>11</td>
</tr>
</tbody>
</table>

(b) Table GAP2

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag2</td>
<td>2</td>
</tr>
</tbody>
</table>

(c) Table GAP3

<table>
<thead>
<tr>
<th>Tag Name</th>
<th>Gap1</th>
<th>Gap2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag1</td>
<td>-11</td>
<td>-8</td>
</tr>
<tr>
<td>Tag3</td>
<td>NULL</td>
<td>9</td>
</tr>
<tr>
<td>Tag4</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

(d) Table GAP4

Figure 3.6: An Example: GAP3 = minus (GAP1, GAP2); GAP4 = intersect (GAP1, GAP2)
The set intersection operator extracts the common tags from the two tables and their corresponding values. Table GAP 4 in Figure 3.6 provides a simple example. Note that in this particular case, the GAP table has two gap columns. In general, as an extension of the basic situation shown in Figure 3.3(b), a GAP table must have one column on tag name and at least one column on gap levels. Similarly, in general, a SUMY table can have more aggregate columns than the ones shown in Figure 3.3(a), so long as it has those columns shown in the figure. Finally, the union operator is defined similarly to intersection. There is also the standard projection operator to remove unwanted columns in a GAP or SUMY table.

3.2.4 Operations in the Extensional World

To complete the presentation of the algebraic framework underlying the GEA, we mention briefly the operations allowed in the extensional world. Because the extensional world is relational, the relational algebra, extended with standard aggregation operations such as sum, average, etc. and sorting, is sufficient. From an implementation point of view, this is significant because little effort is required for the implementation of the extensional world. More details about implementation will be provided in Chapter 4.

3.3 Efficiency Issues

3.3.1 Complexity of Operations

The GEA is intended to support interactive analysis. To achieve this goal, it is important that every operation gives acceptable response time. Thus, in the following, we discuss the complexity of the various operations. Recall that in the intensional and extensional worlds, there are the standard relational operations, and we omit a discussion on those.

1. First, let us consider the three inter-world operations shown in Figure 3.1. The complexity of the mine() operator clearly depends on the exact nature of the mining operation. Especially, for clustering, there are algorithms of varying complexities, from
linear to polynomial on the number of libraries. In the case of fascicles, the complexity is linear with respect to the number of libraries and the number of compact tags. As discussed in [JMN99], there are various parameters that can be tuned to reduce the response time, sometimes at the expense of quality.

2. For the creation of a SUMY table, the time taken to apply the aggregate() operator depends on the exact nature of the aggregation. If the aggregation on each tag simply amounts to finding the range, the average and the standard deviation, then one pass over the libraries is sufficient. But if the aggregation is more complex (e.g., finding the median), the complexity can be higher (e.g., $O(n \log n)$).

3. For the populate() operator, its performance turns out to be challenging. We defer the discussion to subsection 3.3.2.

4. For the creation of a GAP table, the time taken is linear on the number of tags.

3.3.2 Optimizing the Populate() Operator

Recall that for a given SUMY table and an ordinary relation, the populate() operator finds all tuples in the relation that satisfy all the tag ranges contained in the SUMY table. On first sight, the operation is nothing more than a conjunction of a number; say $p$, of range conditions. However, the problem here is that $p$ is very large by the normal standard of relational queries. In the case studies shown in the previous section, a SUMY table easily contains $p = 25,000$ or $p = 30,000$ tags. Thus, the populate operation becomes extremely highly dimensional, amounting to a conjunction of 25,000 or 30,000 range conditions.

The naïve way to handle such a high-dimensional range query is to do a sequential search, which clearly is time-consuming. A better way is to rely on indexing to help reduce the proportion of data that needs to be checked. But there are two immediate questions when it comes to indexing.
For the SAGE data test case, there are 60,000 tags in all. Clearly, we cannot index each and
every one of them. Thus, the first question is which ones to pick. One natural strategy is to pick
the most selective tags. Because a SUMY table is produced by a mining operation, there is little
that is known \textit{a priori} as to which tags and what ranges can be included in the SUMY table. Our
heuristic is to pick the tags with the highest entropy [HK01], that is, highest variation. More
specifically, we seek to build \(m\) indices for the tags with the top-\(m\) highest entropy.

The second, and more difficult, question is what an appropriate value of \(m\) is. To solve for \(m\), we
conduct the following analysis. Let the total number of tags be \(n\) (e.g., 60,000), the number of
tags in a SUMY table be \(p\) (e.g., 25,000), and the number of indices built be \(m\). Assume that
there is a uniform distribution as to which tags are included in the SUMY table. Then the
probability that a tag in the SUMY table is not indexed is this:

\[
Prob(\text{a non-indexed tag selected}) = (1 - \frac{m}{n})
\]

Among the \(p\) tags included in the SUMY table, the probability that only non-indexed tags are
picked is this:

\[
Prob(\text{none of the } m \text{ indices hit}) = (1 - \frac{m}{n})^p
\]

To generalize, let \(w\) be the number of indices hit. That is, among the \(p\) tags included in the
SUMY table, there are \(w\) tags with associated indices. Thus, the above equation corresponds to
\(Prob(w = 0)\), and can be generalized as follows:

\[
Prob(\text{exactly } w \text{ indices hit}) = \binom{p}{w} \times \left(\frac{m}{n}\right)^w \times \left(1 - \frac{m}{n}\right)^{p-w}
\]

Finally, we can fix a probability threshold, say 0.999, such that we solve for the smallest \(m\) that
guarantees at least a 99.9\% chance of hitting at least \(w\) indices:

\[
Prob(\text{at least } w \text{ indices hit}) = [1 - \sum_{i=0}^{w-1} \text{Prob(\text{exactly } i \text{ indices hit})}] \geq 0.999
\]

Based on \(n = 60,000\) total tags, and \(p = 25,000\) tags in a SUMY table, the solutions of \(m\) based on
the above equation are these:
<table>
<thead>
<tr>
<th>At Least ( w ) Indices Hit</th>
<th>Number of Indices Required ((m))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>17</td>
</tr>
<tr>
<td>2</td>
<td>23</td>
</tr>
<tr>
<td>3</td>
<td>27</td>
</tr>
<tr>
<td>4</td>
<td>32</td>
</tr>
<tr>
<td>5</td>
<td>36</td>
</tr>
<tr>
<td>6</td>
<td>40</td>
</tr>
<tr>
<td>7</td>
<td>44</td>
</tr>
<tr>
<td>8</td>
<td>48</td>
</tr>
<tr>
<td>9</td>
<td>51</td>
</tr>
<tr>
<td>10</td>
<td>55</td>
</tr>
</tbody>
</table>

Table 3.1: Number of Indices Require to Guarantee \( w \) Indices Hit

Finally, based on the SAGE data set used in the case studies, the following table shows the percentage of time saved for the populate() operator if \( w \) indices are hit.

<table>
<thead>
<tr>
<th>( w ) Indices Hit</th>
<th>Time Save (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>45</td>
</tr>
<tr>
<td>2</td>
<td>76</td>
</tr>
<tr>
<td>3</td>
<td>78</td>
</tr>
<tr>
<td>4</td>
<td>85</td>
</tr>
<tr>
<td>5</td>
<td>85</td>
</tr>
<tr>
<td>6</td>
<td>85</td>
</tr>
<tr>
<td>7</td>
<td>85</td>
</tr>
<tr>
<td>8</td>
<td>90</td>
</tr>
<tr>
<td>9</td>
<td>90</td>
</tr>
<tr>
<td>10</td>
<td>90</td>
</tr>
</tbody>
</table>

Table 3.2: Time Saving Regard to \( w \) Indices Hit

The Table 3.1 and 3.2 combined tell an interesting story. To save approximately 50% of time needed to execute the populate() operator, there is a 99.9% chance that if the top-17 highest entropy tags are selected for index creation, at least 1 index will be hit (out of the 25,000 tags included in the SUMY table). Similarly, if a 85% time saving is desired, then 32 indices are needed to guarantee that there is a 99.9% chance of having at least 4 indices hit. Finally, for the maximum saving of 90%, 48 indices should be created to guarantee the 99.9% chance to have at least 8 indices hit.
Chapter 4

The GEA System: Case Studies

Because the GEA is designed mainly for domain experts, ease of use and user-friendliness are key design objectives. These objectives are achieved not only by the usual means of developing an easy-to-understand GUI, but also by the development of many macro operations. These are sequences of operations commonly used for convenience. For example, immediately after the mining operation, both the SUMY table and the corresponding ENUM table are created with an automatic invocation of the populate operation. In this way, the user may not need to be too conscious about whether the current analysis is in the extensional or intensional world. Moreover these operations are designed so that the output of an operation becomes the input of another. Therefore, the user can apply these operations repeatedly until the desired results are obtained. In this chapter, we provide a general implementation overview of the GEA system and the cases studies to illustrate the capabilities and usefulness of the GEA.

4.1 Implementation Overview

The GEA is implemented in Java 1.2.2. The system details are encapsulated inside the Java classes, each of this class is responsible for a component of the system. One of the reasons for
using Java is its simple integration and portability. The commercial database tool IBM DB2 Universal Database 7.0 is used for providing the basic relational database support. The GEA has a tight coupling with the underlying DBMS to ensure efficient data access, storage and retrieval. Java Swing is mainly used to create the graphical user interface (GUI). JDBC 2.0 is the application programmer’s interface (API) for connecting the DB2 and Java. JDBC has many drivers; each adapts to different DBMS, such as DB2, Oracle and mySQL. However, from our experience, the operation of switching from one DBMS to another is not as easy a task as the JDBC author claims. Appendix II provides some problems that may be encountered when porting the GEA from DB2 to mySQL. The GEA is implemented under the Microsoft Windows 2000 environment. The experiments are conducted using Intel Pentium III 933 MHz CPU processor and 512 MB RAM. Cygwin 1.3.1 is used to simulate the Unix environment under Windows.

4.2 Pre-processing and Data Cleaning

For a typical data mining exercise in practice, data preparation and cleaning is a component that requires considerable effort, which is necessary to guarantee the quality of the eventual analysis. Analyzing the SAGE data set is the same, partly because each library may have a different number of tags. It has been estimated that because of how the data were collected, 10% of the total number of tags contains sequencing error in each library. These errors result in noise and increase the dimensionality of the data set; consequently, filtering out these errors will clean the data and reduce the dimensionality dramatically. We estimate more than 80% of the unique tags have a frequency of 1, which is 20% of the total number of tags. If we remove all the tags with a frequency of 1 from each library, this will result in the removal of too many unique tags. Sometimes, it is legitimate for a tag to have a frequency of 1 to represent an actual low mRNA; therefore, we can’t conclude a tag is an error based on observations in one library. To reduce errors embedded in the data, we take the union of all the tags in the libraries, and then remove those tags whose expression levels are 0 or 1 in all libraries. The tags have a frequency of 1 in some libraries, and higher frequencies in other libraries are not removed. This step alone
reduces the original number of tags from 350,000 to 60,000. Each of the libraries had 5% to 15% of its total number of tags removed.

Because the SAGE libraries are produced in different laboratories around the world, each of the libraries contains a different number of total tags, which depends on the resources and effort they invest in producing the library. When comparing these libraries, it is more meaningful to put them on the same scale. Therefore, we normalized the data sets by scaling them up to the same total number of tags. It is estimated that there are 300,000 mRNAs per cell, so all libraries are scaled up to this amount. We scale up the data sets by proportionally increase the count of genes that are exist in the library, and the genes that does not exist will remain as zero. With respect to the GEA, all these operations can be carried out in the extensional world. The GUI for carrying out this data cleaning process is show in Figure 4.1.

![Figure 4.1: Data Cleaning Process](image-url)

To carry out the data cleaning process, three pieces of information are required from the user. The first piece of information required is the name of the directory where the SAGE libraries are stored in her computer. By default, all the SAGE libraries are stored in a separate directory for maintenance purposes; the SageLibrary is the default directory in the GEA. Second, according to the libraries existing in this directory, the input file sageName.txt is created to obtain the statistical information of each library. Finally, the user also needs to define the minimum tolerance value for data cleaning calculations and the output files, to store the cleaned data. The clean data will later be loaded to the TAGS table in the database. According to the minimum
tolerance value, if the frequency of a tag in all libraries is lower than or equal to this value, this tag will be removed from the database. The user can click the OK button to start the data cleaning process, or click the Cancel button to exit the current window and return to the main menu.

4.3 Case Studies

So far in this thesis, we have presented the model and the operations of the GEA. To demonstrate the kinds of results the GEA can deliver, we present several of the case studies we conducted using the SAGE data set. Each of the case studies is accompanied by a set of windows, which explain how the GEA operates in detail.

4.3.1 Case 1: Cancerous Brain Versus Normal Brain Tissues

After the data is cleaned, let SAGE be the relation containing the cleaned data. In this case study, we compare normal brain tissues to cancerous brain tissues. Specifically, the steps and the corresponding windows are described in sections 4.3.1.1 and 4.3.1.2 respectively.

4.3.1.1 Steps for Tissues Comparison Operation

1. We use a simple relational selection to collect all libraries involving brain tissues in the extensional world, that is, $E_{\text{brain}} = \sigma_{\text{tissueType} = \text{brain}}(\text{SAGE})$.

2. We apply the fascicle algorithm to produce a cluster containing only cancerous tissues, that is, $\text{SUMY}_1 = \text{mine}(E_{\text{brain}}, \text{fascicle})$.

3. Correspondingly, we apply the populate operation to create the appropriate ENUM table, that is, $\text{ENUM}_1 = \text{populate}(\text{SUMY}_1, E_{\text{brain}})$.
4. To set up the control groups for comparative analysis, we created the following two ENUM tables. ENUM₂ contains all cancerous brain libraries that were not in the cluster, that is, \( \text{ENUM}_2 = (\sigma_{\text{tissueStatus} = 'cancerous'}(E_{\text{brain}})) - \text{ENUM}_1 \). Similarly, ENUM₃ contains all the normal brain tissues, that is, \( \text{ENUM}_3 = (\sigma_{\text{tissueStatus} = 'normal'}(E_{\text{brain}})) \).

5. To create the corresponding SUMY tables, we apply the aggregation operation, that is, \( \text{SUMY}_2 = \text{Aggregate} (\text{ENUM}_2) \) and \( \text{SUMY}_3 = \text{Aggregate} (\text{ENUM}_3) \).

6. To contrast the various groups, we create the GAP table, that is, \( \text{GAP}_1 = \text{diff} (\text{SUMY}_1, \text{SUMY}_3) \).

7. Finally, we remove all the tags in GAP₁ with overlapping ranges, sorted the remaining ones, and plotted the results. Two examples are shown in Figure 4.2 and Figure 4.3.

Figure 4.2 shows an example of a gene identified by contrasting libraries in the fascicle and the normal brain. The expression levels of Ribosomal Protein L12 of various libraries are shown, with asterisks corresponding to the cancerous libraries in the cluster, and squares corresponding to normal libraries. The bar chart shows that the average expression level in the cancerous libraries of the cluster is around 275, compared with the average expression level of around 100 in normal libraries. The values for the cancerous libraries not in the cluster are also shown for reference. Although not all of the cancerous libraries cluster into a fascicle, the average expression levels is higher than expression levels in normal libraries.
By contrasting the libraries in the cluster with normal libraries, Figure 4.2 corresponds to a positive gap scenario, while Figure 4.3 corresponds to a negative gap scenario. The expression levels of Alpha Tubulin are much lower in the cancerous libraries (e.g., close to 0 for those in the cluster) than in normal libraries (i.e., the average being close to 90).
The previous section describes the steps taken to compare the normal and cancerous brain tissues; in this section, we discuss how each window corresponds with these step(s).
**Step 1: Select Tissue Type**

By definition, fascicle is a collection of libraries with a number of similar attributes. For the GEA, fascicles are calculated based on the SAGE libraries grouped by either the system-defined or the user-defined tissue type. For example, system-defined tissue types include the brain, breast, prostate, ovary, colon, pancreas, vascular, skin, and kidney. Alternatively, in some cases, after the data set is examined, the user may find that analyzing a data set formed by a user-defined tissue type is more meaningful. A user-defined tissue type can be the combination of libraries from both the brain and breast tissue type; it allows the user to combine the libraries from different tissue types or remove libraries from a tissue type. In a situation where some of the libraries could never cluster into a fascicle because they contain only a very small amount of total tags, the user may consider removing these libraries from the data set. In other situations where some of the tissue types only contain a small number of libraries, the user could combine it with other tissue types to form a larger data set. By applying the fascicle algorithm to a larger data set, the probability of finding interesting information increases. When a data set contains libraries from more than one tissue type, libraries from the same tissue type are expected to cluster in a fascicle; however, if libraries from different tissue types are clustered into a fascicle, this may also suggest some interesting insights between the two tissue types. Therefore, the GEA is implemented to give the user an opportunity to be creative, which allows him or her to freely combine or separate tissue types to form user-defined data sets, and have better control over data sets they examine; creativity is very important in the analytical process. Figure 4.4 is a GUI for creating a data set based on the system-defined tissue type. The GUI for creating a data set based on the user-defined tissue type is shown in Figure 4.15.

When the user creates a data set based on a system-defined tissue type, the GEA allows him/her to view information, such as the total number of libraries, and the libraries contained in the tissue type. When a tissue type is chosen to create the data set, a new table is formed in the database to store the data; a plain text file and a binary file are also created to store the data in ASCII and binary format. The example in Figure 4.4 shows that the brain tissue type is selected, and consists of 24 libraries, with the names of the libraries displayed in the Libraries in Tissue List on the right-hand-side of the window. When the user double clicks a library in the list, detailed information from the library is retrieved and displayed in a new window (a more detailed
discussion is found in section 4.4.4.2). When a tissue type is selected for analysis, we click the **OK** button to form the data set; if the data set already exists, we click the **NEXT** button to proceed directly to the next step, which is “calculate fascicles”.

![Figure 4.4: Data Set Based on System-Defined Tissue Type](image)

**Figure 4.4: Data Set Based on System-Defined Tissue Type**

### Steps 2-3: Calculate Fascicles

The Fascicles algorithm requires two parameters, one is the number of compact attributes, and the other is a tolerance vector. In order for the user to calculate the fascicles, the tolerance vector must first be created. The tolerance vector in the GEA is called the metadata. Each attribute in the data set requires a compact tolerance value assigned to it; this value is a percentage of the width of the attribute. For example, if the width of the value of tag AAAAAAAAAA is 200, five percent of the width is selected as the compact tolerance, which is equal to 10. Theoretically, if the percentage is smaller, fewer libraries are contained in the fascicles, while the attributes remain the same.

Figure 4.5 is the window for generating the metadata. First, we select the tissue type to be examined. (See Figure 4.4) The input file is a plain text file, created in Step 1 (for performance purposes, reading a large amount of data from a plain text file proves faster than from a database). The metadata generated is saved into a file with the extension of ‘.meta’, which consists of the attribute name and the compact tolerance value for each attribute in a pre-defined
format. The example in Figure 4.5 generates the metadata for the brain tissue type. Reading data from the input file brainfile reveals that 10 percent of the width of the attribute is the compact tolerance value; the output is saved to file brainfile.meta. According to the input parameters, the user can click the OK button to generate the metadata file. If the metadata already exists, an error message is displayed and the user can then click the NEXT button to proceed to the next step directly.

![Create metadata for this tissue type](image)

**Figure 4.5: Generate Metadata**

After the tolerance vector is generated, the user can apply the Fascicle algorithm to the data set and calculate the possible fascicles. Figure 4.6 is the window for calculating the fascicles, and it requires six essential input parameters to complete its calculation. In this example, the brain tissue type is selected for analysis; each fascicle needs to contain at least 35,000 compact attributes. By definition, the number of compact attributes cannot exceed the total number of attributes in the tissue type. The data set in binary format is brainfile.b, and the metadata file is brainfile.meta. By definition, the batch size represents the number of libraries the program can read in at once; in this example, six libraries are read in at once. Also, a fascicle must contain at least three libraries in order for it to be frequent. The resulting fascicles write to an output file, brain35k, and the details of each fascicle are saved in separate files. After the input parameters are provided, the user can click the OK button to calculate the fascicles.

44
Steps 4-5: Select Fascicles to Examine

The ENUM₁ and SUMY₁ tables are created while the fascicles are calculated. Figure 4.6 and 4.7 illustrate how the results are displayed to the user. According to the selected output file in the fascicles pull-down combo-box, the fascicles in this output file are displayed in the Fascicles List on the left-hand-side of the window. The output of brain35k includes four fascicles such as brain35k_1, brain35k_2, brain35k_3, and brain35k_4. When the user applies a single click on a fascicle in the Fascicles List, two sets of information are displayed. One is the related information of the fascicle that is displayed in the Fascicles Info section on the right-hand-side of the window. This information includes the name of the fascicle and all of the input parameters used to calculate it. If the fascicle has been through a purity check (more details are found in Figure 4.8), the purity checks field displays “pure” or “non-pure” or else null is assigned. The definition of a pure fascicle is that the libraries in the fascicle consist of only one property, which means the libraries can only be cancer, normal, bulk tissue, or cell line, and not a mix of different properties. In addition, the libraries contained in the fascicle are displayed in the Libraries List.
located in the center of the window. By double clicking a library in the Library List, the detailed information regarding the library is displayed to the user in a separate window (more details are found in section 4.4.4.2).

In Figure 4.7, the fascicle brain35k_4 is selected, which is a brain tissue type with 35,000 compact attributes. It uses brainfile.b and brainfile.meta as inputs. The batch size is 6, with at least 3 libraries contained in the fascicle, which contain only pure cancerous libraries, including 31, 32, and 33.

The ability to examine the properties of the fascicles is very important because only the pure fascicles can be further analyzed. The four properties of the fascicles are cancer, normal, bulk tissue, and cell line. To perform a purity check of a fascicle, the user simply double clicks the name of the fascicle in the Fascicles List; if the fascicle is pure, a dialog box similar to the one found in Figure 4.8 is displayed. In this example, fascicle brain35k_4 is confirmed to be a pure fascicle containing only cancerous libraries. Therefore, the creation of the other two ENUM tables, and two-associated SUMY tables, are allowed for this fascicle. The ENUM₂ table is created for the libraries that have the same property as the fascicle of this tissue type, but are not in the fascicle; another ENUM₃ table is created for all the libraries that have different properties from the fascicle of this tissue type. The SUMY₂ and SUMY₃ tables contain only the compact

Figure 4.7: Examine Fascicles
attributes of the fascicle, and the aggregate values are calculated according to the data in ENUM2 and ENUM3, accordingly. In Figure 4.8, ENUM2 contains the cancerous libraries that are not in fascicle brain35k_4, and ENUM3 contains all the normal libraries in the brain tissue type. Nevertheless, if a fascicle is non-pure, another dialog box is displayed, the formSUM button is ‘hidden’, and the analysis of this fascicle is terminated. When the user clicks the formSUM button, the ENUM and the SUMY tables are created for this fascicle.

![Image of GUI showing fascicle found in tissue type]

**Figure 4.8: Purity Check**

**Steps 6-7: Examine the Fascicles by Creating GAPs**

As mentioned above, each of the pure fascicles has three ENUM and SUMY tables for the libraries based on that property. By performing the GAP operation provided in the GEA, the user can determine the differences between two of the SUMY tables in the same or different tissue types. Figure 4.9 illustrates how to perform the GAP operation in the GEA. The user selects a set of SUMY tables to be displayed in the Summary Lists, which can be sorted by tissue type, fascicles, or category. For example, if the user wants to display all the SUMY tables that are in the brain tissue type, s/he must select brain in the “type” pull-down combo box in the Sorted-by section. (Another example is to display all the SUMY tables in the fascicle brain35k_4, by selecting brain35k_4 in the Fascicles pull-down combo box. This is the same for displaying the SUMY tables in each of the categories such as cancer in fascicle, cancer not in fascicle, or normal. The sorting is the same when it is applied to bulk tissue and cell line.)
When one of the three sortings is selected, the SUMY tables that satisfy the condition is displayed to both the Summary1 List and Summary2 List on the left of the window. To select the two SUMY tables to calculate the GAP, the user double clicks one of the SUMY tables in each of the list.

Figure 4.9: Create GAP

Figure 4.9 is an example of the calculation of a GAP table, namely brain35k_4canvsnor_gap; it calculates the differences between the SUMY\textsubscript{1} brain35k_4CancerFasTbl (cancerous libraries) and SUMY\textsubscript{3} brain35k_4NormalTable (normal libraries). This GAP table contains a list of tags that have significant differences in expression level between the cancerous tissue in fascicle and the normal tissue in the brain tissue type. When the user clicks the Find GAP button, the GEA calculates the GAP values for all the tags in the two SUMY tables, and saves the result to the GAP table brain35k_4canvsnor_gap in the database. However, according to the top-\textit{x} value defined by the user, only the top-10 tags with the highest GAP values are displayed on the Top Gap Values List, and a top-\textit{x} gap table is created to save this information. The information displayed for the top-\textit{x} tags includes the tag name, the tag number in brackets, and the GAP value. If the GAP value is positive, it means that the expression value in brain35k_4CancerFasTbl is higher than the expression value in brain35k_4NormalTable. If the GAP value is negative, it means that the expression value in brain35k_4CancerFasTbl is lower.
than the expression value in brain35k_4NormalTable. The usefulness of the buttons **Search Tag** and **Calculate Top Gap** is discussed in detail in Sections 4.4.4.1 and 4.4.3.1. The functionalities of the **Gap Comparison** button are discussed in Sections 4.3.3 and 4.3.4.

For each of the top tags, the GEA provides a simple but effective tool for visualizing the expression values’ distribution in the tag. Figure 4.10 shows the visualization of a top tag in a graph. This graph plots the expression value of the tag GAGGGAGTTT. The x-axis represents the libraries, and the y-axis represents the expression values of the libraries. The red dots on the top represent the expression values of the cancerous libraries in a fascicle, and the blue squares on the bottom plot the expression values of the normal libraries of the same tissue type. In this case, it is very obvious that the expression values in the cancerous libraries in a fascicle are much higher than the expression values in the normal libraries of the same tissue type. If a tag has a positive gap value and shows significant differences in expression values between cancerous and normal tissue, the tag warrants a more detailed clinical investigation. The user can save this result by clicking the **SAVE** button, or dismiss the graph by clicking the **CLOSE** button, which will bring the user back to the previous screen, as shown in Figure 4.9.

![Plot Tag](image)

**Figure 4.10:** Visualize the Distribution of a Tag
4.3.2 Case 2: Cancerous Brain Tissues Inside and Outside of the Fascicle

While Figures 4.2 and 4.3 identify two genes/tags which behave differently between cancerous and normal brain tissues, we conducted a similar analysis on cancerous brain tissues inside and outside of the fascicle. Using the tables produced earlier (as described in Section 4.3.1), we
created a new GAP table, that is, GAP$_2$ = $\text{diff}(\text{SUMY}_1, \text{SUMY}_2)$. Again, we removed the tags with overlapping ranges, sorted the remaining gap levels and plotted the top findings. Figure 4.11 shows an example of this. The cancerous libraries in the fascicle show a much lower level of expression than cancerous libraries outside the fascicle (i.e. 11 being the average). Identification of these kinds of genes may eventually lead to the discovery of different sub-types of brain cancer.

The example in Figure 4.12 is very similar to the GUI in Figure 4.9; the only difference is that we are now comparing the cancerous tissues inside of the fascicle against the cancerous tissues outside of the fascicle. In the GAP table, namely brain35k_4canvscnif_gap, we calculate the differences between the $\text{SUMY}_1$ brain35k_4CancerFasTbl (cancerous libraries inside the fascicle) and the $\text{SUMY}_2$ brain35k_4CancerNotInFasTable (cancerous libraries outside the fascicle). This GAP table contains a list of tags that have the greatest difference in expression level between cancerous tissue inside and outside the fascicle of the brain tissue type. When the user clicks the Find GAP button, the GEA calculates the GAP values for all the tags in the two $\text{SUMY}$ tables, and saves the result in the GAP table brain35k_4canvscnif_gap. The top-10 tags...
with the highest GAP values are displayed to the user and saved to a top-x gap table. If the GAP value is positive, it means that the expression value in brain35k_4CancerFasTbl is higher than the expression value in brain35k_4CancerNotInFasTable. If the GAP value is negative, it means that the expression value in brain35k_4CancerFasTbl is lower than the expression value in brain35k_4CancerNotInFasTable. In our experiments, the GAP values found between the cancerous tissue inside of the fascicle and normal tissue are often larger than the GAP values found between the cancerous tissue inside and outside of the fascicle. This suggests that the expression values of the cancerous tissues inside and outside of the fascicle are more similar than the expression values in normal tissues.

4.3.3 Case 3: Finding Genes always having a Higher/Lower Expression Level in the Cancerous Libraries

While the above examples focus on analyzing a single tissue (brain tissues), more complex analysis involving multiple tissue types can also be carried out within the GEA. In this study, we attempted to identify a list of genes that, for all tissue types, always has lower expression levels in cancerous tissues than in normal tissues. To do so, we created an appropriate GAP table for each tissue type, as shown in Sections 4.3.1 and 4.3.2. To compile the list of genes for each specific tissue type, we applied “selection” to keep only the tags with negative gap values, and then applied “projection” to retain only the tags. Finally, all these tag lists were intersected to give only those tags applicable to other tissue types. This example demonstrates the usefulness of the set operations.

Figure 4.13 is an example that shows how to perform more complex analyses between multiple tissue types. It identifies a list of genes that always have lower expression levels in cancerous tissues in the fascicle than in the normal tissues for both brain and breast tissue types. First, the user must follow the steps, as stated in Section 4.3.1, to create a GAP table between the cancerous tissue in the fascicle and the normal tissue for the breast tissue type. Then, the user double clicks the GAP tables in the GAP Lists to select brain35k_4cancsnor_gap of brain and
breast25k_3canvsnor_gap of breast. The user selects the Intersect radio button in the compare section, and clicks the **Compare GAP** button to identify a list of tags that exist in both GAP tables, and then saves the results to a new compare gap table brainBreastIntersect1, as defined by the user. In this example, according to the Display-x value defined by the user, only the first 20 tags are displayed to the Gap Value List on the right-hand-side of the window. The GEA provides thirteen queries for further analysis of the result in the compare gap table:

1. Tags always have higher expression values in SUMY\textsubscript{a} in both GAP tables
2. Tags always have lower expression values in SUMY\textsubscript{a} in both GAP tables
3. Tags always have higher expression values in SUMY\textsubscript{b} in both GAP tables
4. Tags always have lower expression values in SUMY\textsubscript{b} in both GAP tables
5. All tags have non-null gap values in both GAP tables
6. Tags have higher expression in SUMY\textsubscript{a} of GAP\textsubscript{a}, not in SUMY\textsubscript{a} of GAP\textsubscript{b}
7. Tags have lower expression in SUMY\textsubscript{a} of GAP\textsubscript{a}, not in SUMY\textsubscript{a} of GAP\textsubscript{b}
8. Tags have higher expression in SUMY\textsubscript{b} of GAP\textsubscript{a}, not in SUMY\textsubscript{b} of GAP\textsubscript{b}
9. Tags have lower expression in SUMY\textsubscript{b} of GAP\textsubscript{a}, not in SUMY\textsubscript{b} of GAP\textsubscript{b}
10. Tags have higher expression in SUMY\textsubscript{a} of GAP\textsubscript{b}, not in SUMY\textsubscript{a} of GAP\textsubscript{a}
11. Tags have lower expression in SUMY\textsubscript{a} of GAP\textsubscript{b}, not in SUMY\textsubscript{a} of GAP\textsubscript{a}
12. Tags have higher expression in SUMY\textsubscript{b} of GAP\textsubscript{b}, not in SUMY\textsubscript{b} of GAP\textsubscript{a}
13. Tags have lower expression in SUMY\textsubscript{b} of GAP\textsubscript{b}, not in SUMY\textsubscript{b} of GAP\textsubscript{a}

![GAP Comparison](image)

**Figure 4.13:** Cancerous Tissue always has Lower Express Values than Normal
In the GAP calculation, the order of the SUMY table is very important because it determines the sign of the GAP value, which reveals that the GAP is either a positive or negative value. SUMY_a and SUMY_b represent SUMY_1 and SUMY_2 as defined in Section 3.2.2, for the calculation of the GAP table. In this example, SUMY_a represents the SUMY table of cancerous tissue in the fascicle in the brain or breast tissue type, and SUMY_b represents the SUMY table of normal tissue in the brain or breast tissue type. GAP_a and GAP_b represent the GAP tables of the brain and breast tissue types, respectively. Queries 1 to 5 apply to all comparison operations, including Union, Intersection, and Difference. However, queries 6 to 13 only apply to Union and Intersection, but not Difference. The example in Figure 4.13 displays the results of the analysis of query 2, which reveals that these tags always have lower expression values in the cancerous tissues in fascicle than in the normal tissue for both brain and breast tissue types. The GEA also identifies a list of tags that always has higher expression values in the cancerous tissue in fascicle than in the normal tissue for both brain and breast tissue type. The user simply selects query 1 from the query pull-down combo box, and the result returns all the tags with positive gap values in both tissue types. The list of tags identified in this step may suggest possible drug targets. As mentioned in Section 2.4, theoretically, if we could determine a list of genes that have been turned on or off in the disease tissues, and are able to determine an appropriate method for altering these expression values back to “normal,” the disease may eventually be cured.

4.3.4 Case 4: Finding Genes Unique in Each Type of Cancer

Although the similarities between the GAP tables in different tissue types are interesting, the dissimilarities can also provide interesting information. For example, in comparing the differences between the GAP tables derived from different tissue types, the results may provide information for a unique set of genes that have great differences between cancerous and normal libraries in one tissue type, but not in the other. These results may provide information on the different types of cancer possibly caused by different sets of genes. To accomplish this task, we apply the set minus operator to the GAP tables corresponding to various tissue types or types of cancers.
The same window as shown in Figure 4.13, is applied to accomplish this task; however, for this task, the user selects the Difference radio button in the compare section. The example in Figure 4.14 illustrates how the brain tissue type differs from the breast tissue type. By comparing the GAP tables, brain35k_4canvsnor_gap of brain against breast25_3canvsnor_gap of breast, a set of tags is identified, which is unique only in the brain, but not in the breast tissue type. A compare gap table brainBreastDiff1 is created to save this information. According to query2, selected by the user, only the tags that have lower expression values in the cancerous libraries in the fascicle than in the normal libraries are displayed to the Gap Value List. The user can also identify a set of tags that always has higher expression values in the cancerous tissue in fascicle than in normal tissue, by selecting query1.

![GAP COMPARISON](image)

Figure 4.14: Brain, Lower Expression in Cancer than Normal

### 4.3.5 Case 5: Verification with the ENUM Tables

In all the examples so far, the focus has been on the GAP tables and in the intensional world. As discussed in previous sections, the GEA can also provide meaningful operations in the extensional world. For example, through examining some of the results shown in Sections 4.3.1

55
to 4.3.4, we might decide to verify the results by returning to the extensional world and exploring the corresponding ENUM tables. We might wonder whether the outcome of the previous analysis would be affected by the removal of certain libraries. In this process, we might apply various relational algebra operations to prepare the appropriate ENUM tables, and eventually repeat or redo the kinds of operations shown in Sections 4.3.1 to 4.3.5. Figure 4.15 is the GUI that allows the user to create customized data sets for analysis (as discussed in Section 4.3.1.2). All the SAGE libraries in the database are listed in the Libraries List on the left of the window. To select the libraries to form a customized data set, the user simply double clicks a library, or highlights multiple libraries and then presses the >> button to copy the libraries to the Selected Library List on the right-hand-side of the window. If the user decides to remove the libraries from the Selected Libraries List, s/he double clicks the library, or highlights multiple libraries and then presses the << button. All the libraries in the Selected Libraries List are then assigned to a new user-defined tissue type. For the example show in Figure 4.15, when the user clicks the OK button, a new ENUM table for newBrain tissue type is created. More search functions and data manipulations in the extensional world are presented in Section 4.4.4.2.

<table>
<thead>
<tr>
<th>Customize Tissue Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select the libraries to form customize tissue type</td>
</tr>
<tr>
<td>1) Double clicks the libraries in the left list to transfer them to the right list</td>
</tr>
<tr>
<td>2) Give a name to the new tissue type.</td>
</tr>
<tr>
<td>3) Select OK button to create a table for the tissue type.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Select Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAGE_Duke_95-349</td>
</tr>
<tr>
<td>SAGE_Duke_BB542_normal_cerebellum</td>
</tr>
<tr>
<td>SAGE_Duke_cerebellum</td>
</tr>
<tr>
<td>SAGE_Duke_GBM_H1110</td>
</tr>
<tr>
<td>SAGE_Duke_H1020</td>
</tr>
<tr>
<td>SAGE_293-IND</td>
</tr>
<tr>
<td>SAGE_95-259</td>
</tr>
<tr>
<td>SAGE_95-260</td>
</tr>
<tr>
<td>SAGE_Duke_GBM_H1110</td>
</tr>
<tr>
<td>SAGE_Duke_BB542_normal_cerebellum</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tissue Type Name:</td>
</tr>
<tr>
<td>newBrain</td>
</tr>
</tbody>
</table>

Figure 4.15: Create Customize Data Set for User-defined Tissue Type

4.3.6 General Remarks

As presented in the case studies above, the GEA provides a rather expressive algebraic environment for supporting multi-step cluster analysis of gene expression data. The algebraic
framework ensures closure in that the output of a particular operation can form the input of other operations. In this way, with a creative combination of operations, a cancer/biology researcher can identify a very small number of genes that have a very high potential of being "interesting" targets for further clinical analysis. The GEA is not only able to find the differences between cancerous and normal tissues, but is also able to detect the differences between different control groups, such as between bulk tissues and tissues made of cell line.

4.4 Highlight Features in the GEA

So far we have discussed the GEA model and its operations, case studies, and efficiency issues. In the following, we focus on some special features implemented in the GEA. We first present the manipulation of range arithmetic. Then we describe a feature called the lineage feature, which helps the user manage (intermediate) the results from an analysis. Other interesting features include the manipulation of the top tags, and various search facilities for retrieving data that can be applying during the analytical process.

4.4.1 Range Arithmetic

Recall from Figure 3.3(a) that there is a Range column in a SUMY table. In this column, an entry is in the form \([\text{min}; \text{max}]\). To allow for more convenient manipulation of these range values, the GEA supports the well-known range arithmetic proposed by Allen [ALLEN83, ALLEN84]. This algebra can express any possibly indefinite relationship between two intervals; Table 4.1 exhibits the basic interval relations. It includes such predicates as A before B, A meets B, A overlaps B, A during B, and so forth, where A and B are two intervals. With Allen's algebra, the user can more easily extract a set of tags that satisfy a given interval in the Range column.
<table>
<thead>
<tr>
<th>Relation</th>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>A before B</td>
<td>b</td>
<td>AAA</td>
</tr>
<tr>
<td>B after A</td>
<td>bi</td>
<td>BBB</td>
</tr>
<tr>
<td>A meets B</td>
<td>m</td>
<td>AAA</td>
</tr>
<tr>
<td>B met-by A</td>
<td>mi</td>
<td>BBB</td>
</tr>
<tr>
<td>A overlaps B</td>
<td>o</td>
<td>AAAA</td>
</tr>
<tr>
<td>B overlapped-by A</td>
<td>oi</td>
<td>BBBB</td>
</tr>
<tr>
<td>A during B</td>
<td>d</td>
<td>AAA</td>
</tr>
<tr>
<td>B includes A</td>
<td>di</td>
<td>BBBB     BBBB</td>
</tr>
<tr>
<td>A starts B</td>
<td>s</td>
<td>AAA</td>
</tr>
<tr>
<td>B started-by A</td>
<td>si</td>
<td>BBBB</td>
</tr>
<tr>
<td>A finishes B</td>
<td>f</td>
<td>AAA</td>
</tr>
<tr>
<td>B finished-by A</td>
<td>fi</td>
<td>BBBB</td>
</tr>
<tr>
<td>A equals B</td>
<td>e</td>
<td>AAA</td>
</tr>
<tr>
<td>B equals A</td>
<td>e</td>
<td>BBB</td>
</tr>
</tbody>
</table>

Table 4.1: Basic Interval Relations

Search the range of the tag(s) that satisfy the range given by the user. The result will display the range of the satisfied tags. For example, for tag range search, enter AAAAAAAAAAAAAA-AAAAAAAACT. Please input only integer values for the range. Also, you could select only one summary table to search when you want to search any tags; else you could select multiple summary table.

Figure 4.16: An Example Window on Range Arithmetic
The example in Figure 4.16 searches the two SUMY tables referred to as brain25k_3NormalTable and brain30k_3CancerFasTab. In this example, the user wants to determine those tags having an expression level overlapping the level range from 10 to 700. In the result display window, the tag name, the tag number and the search result are shown. If the range of the tag does not overlap with the given range, “NO” is shown, as for Tag 568 in the first SUMY table. If the overlap relation is satisfied, then the actual range is shown. For instance, for Tag 573, the range is [20,616] in the first SUMY table. Finally, if the tag does not exist in the SUMY table, “NE” is shown.

The previous range arithmetic example illustrates how to search the specific set of tags in multiple SUMY tables. Next, we present how the GEA determines all the tags that satisfy the range conditions in a SUMY table. Figure 4.17 shows an example where the user wants to determine all the tags in the SUMY table brain25k_3NormalTable, and where their ranges overlap between the ranges of 5 to 700. The Search button will trigger this operation, and only the tags that satisfy these conditions are displayed in the Result List. Tags that do not satisfy these conditions are omitted. Again, the result returns the tag name, the tag number in brackets, and the range of the tag.

![Figure 4.17: Range Arithmetic -- Any Tag](image)
4.4.2 The Lineage Feature

Recall that a typical data mining exercise is a multi-step process, rather than a one-shot attempt. This cannot be truer for gene expression analysis. Typically, the user may need to use a non-trivial number of operations to analyze the data. As the number of operations performed becomes larger, the user may fail to remember what operations have been used to create previous intermediate results.

To address this problem, the GEA system provides a lineage feature. The idea is to keep historical information to help the user better organize, manage and search the previous results. More specifically, the database is carefully designed to record all necessary information, and a GUI interface is provided. As shown in the left-hand-side of Figure 4.18, the window display is similar to a Window-Explorer window, in that each folder can be expanded and collapsed by the user. This helps the user to narrow the focus to one folder at a time. For the example shown in Figure 4.18, the Clustering/brain folder contains information about all the brain-related fascicles. If the ENUM table corresponding to fascicle brain25k_3 is selected, the right-hand-side of the window provides detailed metadata about this fascicle.

For each of the operations performed, detailed information is captured to lead the user in recalling and understanding each operation, such as the operation calculated, what parameters are used, and some important user comments. This information reminds users of what they did, and suggests what they can do next. For example, the fascicle has 25000 compact dimensions, and is generated from the binary and meta files brainfile.b and brainfile.meta, respectively. The input parameters used to generate the fascicle were 6 (for the number of batches) and 3 (for the minimum frequency). Finally, the user records an earlier comment that special attention may need to be paid to the compact dimensions. Thus, this feature reminds the user of what was done before, so that the user can decide what to do next.

Apart from detailed information about tables, the lineage feature also indicates the relationship between tables. For example, the brain25k_3CancerFasTbl is a SUMY table based on brain25k_3. Similarly, the table b25canvscnf_gap1 is a GAP table based on the SUMY table.
brain25k_3CanNotInFasTbl. Recall that a GAP table is typically generated from applying a `diff()` operation to two SUMY tables. Thus, for example, if the GAP table b25canvscsnif_gap1 is generated from brain25k_3CancerFasTbl and brain25k_3CanNotInFasTbl, then b25canvscsnif_gap1 appears under both SUMY tables. (In Figure 4.18, because brain25k_3CanNotInFasTbl is not expanded, b25canvscsnif_gap1 is not shown under brain25k_3CanNotInFasTbl.)

Note that the GAP table b25canvscsnif_gap1, which is the result of an analytical operation, is also shown under the Analyze folder. The information under b25canvscsnif_gap1 shows the two aforementioned SUMY tables, which are used to generate the GAP table. Furthermore, there is an additional GAP table called b25canvscsnif_gap1_10. The latter is obtained by applying a selection operation to the original b25canvscsnif_gap1 to extract the top-10 positive gaps.

At any point in time, the user may select a particular table in a certain folder, and review its contents. This is the purpose of the View button. This allows the user to manage intermediate results effectively, and to create ideas for the next operation, either starting from the table just viewed, or somewhere else.

Notice that storing intermediate results consume space. Thus, to provide better resource management, the GEA provides two ways to remove intermediate results. First, the user may only choose to remove only the contents of a table, but keep the detailed metadata of the specific operation. This allows the user to free up some scarce storage space in the system. If the user wants to re-generate the content of the table, the stored metadata can be used directly.

The second deletion option is useful when the user has performed numerous operations, and concludes that none of the results along this path are interesting. Then, the user can remove the contents of a table, its detailed metadata, and all other tables generated from it. The Delete button allows the user to perform one of these two deletion options.
4.4.3 Manipulation of Top Gaps

4.4.3.1 Calculate Top Gap Table
To provide the user with more flexibility in the analytical processes, the GEA allows the user to start the analytical process from any step in the menu. Recall the example in Figure 4.9, where the user decides to display only the top-10 tags with the highest gap values (top gaps), while the GAP table is calculated. If the user later decides to obtain more top gaps (e.g. the top 20 tags) from the gap table calculated previously, there are two satisfactory methods. One solution applies the GUI in Figure 4.9; however, this is inefficient because a gap table with the same content is produced. Another solution, which is more efficient and consistent, creates a new window for the user to recalculate the top-20 tags for the existing gap table. Figure 4.19 shows a window for calculating the top gaps on a gap table. The user can select the set of gap tables to be displayed in the GAP list based on the fascicle or the tissue type they belong to. In this case, the gap tables that belong to the fascicle brain35k_4 are displayed, and the user calculates the top-30 tags in the GAP table brain35k_4canvsnor_gap. The gap table is selected, and the two SUMY tables derived from it are brain35k_4CancerFasTbl and brain35k_4NormalTable. By clicking the Calculate Top Gap button, the result of the top-30 tags are displayed on the bottom left-hand-side of the window, and a new top gap table is created to save this information. Each of the tags with the top gap value can be examined by obtaining its related biological information, also to visualize the distribution of the tag by plotting their expression values (as discussed in Figure 4.9). The functionality of the View Top Gap button is discussed next.

**Figure 4.19: Calculate Top Gap Table**
4.4.3.2 View Top Gap Table

As mentioned earlier, when the top gaps are calculated, a new top gap table is created for saving the information for future reference. A window is developed for viewing the information in the top gap tables, as shown in Figure 4.20. Again, for better data organization, the GEA allows the user to express her selection as general or specific. Since the top gap tables belong to the gap table, and the gap tables are calculated from the fascicles, in general, if a user selects a fascicle, then the top gap tables belonging to it are displayed in the Top GAP list. In a more specific case, all the gap tables calculated from this fascicle are also displayed in the GAP List; by selecting a gap table, all the top gap tables belonging to it are displayed in the Top Gap List. In Figure 4.20, the user views the information in the top gap table brain35k_4canvsnor_gap_30, and when it is clicked on, its information is displayed in the bottom right hand corner of the window. When the user clicks the **View Top Gap** button, the top-30 gaps’ values in the Top Gap table are displayed in the Top Gap Values list. The user is then able to examine each of the top gaps’ biological information and visualize the expression values in a graph.

![Figure 4.20: View Top Gap Table](image)

Summary 1:
- brain35k_4CancerFasTbl

Summary 2:
- brain35k_4NormalTable

**CAP Name:** brain35k_4canvsnor_gap

**Top X:** 30
4.4.4 Search Operations

As mentioned in the previous sections, a number of search operations are available to the user for extracting the useful information from the database. There are two sets of search operations provided in the GEA. One set of operations is used to extract the information from the Expression Analysis Database. Another set of operations is for general searching, which is used to extract the information from SAGE Database.

4.4.4.1 Expression Analysis Database

The menu in Figure 4.21 shows the operations available under the submenu of the Expression Analysis Database in the Search menu of the GEA. Databases that can be searched for, include UNIGENE, PUBMED, PFAM, KEGG, SWISSPROT, GENEBank, and OMIM.

However, this part of the operation is not fully implemented in the GEA. For details on how to integrate this into the GEA, please refer to Section 5.2. A prototype in Figure 4.22 illustrates how the Express Analysis Database can be incorporated into the GEA, and what type of information can be retrieved. There are three pieces of information that the user can obtain from this prototype.

1. Translate the tag to its corresponding gene: the user inputs the name of a tag and clicks the search button located on the top of the window; the GEA returns the corresponding gene name.

2. To determine the corresponding protein sequence for a gene: the user inputs the gene name obtained from 1, and clicks the search button in the middle of the window to trigger the search in the database; the GEA returns the related protein sequence.

3. Search publications related to the gene: the user inputs the gene name, and clicks the search button in the bottom of the window; a number of publications that are written to describe this gene are returned to the user for further studies.
Figure 4.21: Search Expression Analysis Database

Figure 4.22: EADB Search
The example shows that the tag CCTTGAGTAC corresponds to the gene aldolase C. When searching the information related to the gene aldolase C, its protein sequence and two related publications are returned. By completing these simple searches, many valuable pieces of biological information can be retrieved, and thus the user may be able to make better observations when s/he pieces the information together.

4.4.4.2 General Database Searches

The gene expression data contains a large amount of knowledge; however performing a manual search on the data is almost impossible due to its size and complexity. Therefore, a set of general search facilities are implemented in the GEA to obtain information such as Library Information, Tissue Type Information, Range Search for Tag, Range Search for Library, and Tag Frequency. In the following sections, each of these search facilities is explained in detail.

Search SAGE Library Information

A relation is created for storing the information of the SAGE libraries during the database design. For each of the libraries, the information includes:

- The name of the library.
- The library ID, which is a numerical value from 1 to 100.
- The tissue type, which is one of the following: brain, breast, prostate, ovary, colon, pancreas, vascular, skin, and kidney.
- The neoplasatics stage has the tissue that is normal or cancer.
- The tissue source, which means that when is the tissue is extracted, one is bulk tissue, which is extracted from the human body, and the other one is cell line, which grows in a test tube.
- The total number of tags in a library is the sum of the tag count.
- The unique number of tags in a library is the sum of the number of tags.

An example in Figure 4.23 illustrates the search for library information in the GEA. There are two ways for searching this library information: one is by the library ID, and the other is by the
library name. The library information mentioned above is displayed in the text box on the right-hand-side of the window. This example shows that the search is based on library number 29.

![Figure 4.23: Search Library Information](image)

**Search Tissue Type Information**

In many situations, the user may want to know which libraries belong to certain tissue types (including system-defined and user-defined tissue type); for example, when the user creates customized data sets or examines fascicles, and so forth. To make the retrieval of information easier, the GEA displays a window similar to the one shown in Figure 4.24, to perform this search; the design of the window is also very similar to the one in Figure 4.4. When the user simply double click one of the tissue types in the Tissue Type List, all the libraries belonging to this tissue type are displayed on the right-hand-side of the Libraries in Tissue List. Also, the GEA counts and displays the total number of libraries in this tissue type. Although this function seems very simple, it is a very time consuming operation when it is performed manually; however, automating this process can save a lot of time and energy. When the user clicks on a library in the Library in Tissue List, the library information is returned in a new window similar to the one in Figure 4.23. This example shows the user searching for information on the brain tissue type.
Search Frequency for the Tag
Each library contains more than sixty thousand tags, and each tag may contain a different expression value in each library. If the user wants to extract an expression value for a tag in a library, a manual process may still be possible; however, this task will become very complicated if the user wants to extract the expression values for a set of tags in multiple libraries. To solve this problem, a search of frequency for the tag operation is introduced.

An operation that searches the expression values for a set of tags in multiple libraries is available in the general search. First, the user selects the radio button Tag Range in the Tags section, and enters the first and the last tag he wants to search. Then, the user selects the radio button Single or Multiple Libraries in the Libraries section. If the user wants to search the expression value of the tag in all the existing libraries, s/he can simply select the radio button All in the Libraries section; otherwise, the user must highlight the libraries that s/he wants to search. Figure 4.25 is an example which shows the extraction of the expression values in the tags from AAAAAAAAC to AAAAAACCC, in the libraries SAGE_293-IND, SAGE_95-259, and SAGE_95-260. The result is displayed in the Result List, and the retrieved information, including Tag Name, Tag Number in brackets, and the expression values of the tag in each SAGE library. An example, Tag AAAAAAAAG, has a tag number of 3, and the expression values for the selected libraries are 26, 0, and 7, respectively.
In the GEA, the user can also search expression values for a single tag instead of a multiple number of tags at once. An example in Figure 4.26 shows a user searching the expression values for a tag AAAAAAAAAC, in the libraries, SAGE_293-IND and SAGE_95-259. The result shows that the tag number for AAAAAAAAAC is 2, and the expression values for the selected libraries are 13 and 8, respectively.
4.4.5 Checking Features

So far, we have discussed some of the special features implemented in the GEA. The discussion now focuses on the enhancement of the toolkit by incorporating a considerable amount of checking features that better support usability and user interaction. A piece of high-quality software should provide the user with as many hints and useful information as possible, especially for the user who is not familiar with the product and what problems are encountered. Therefore, during the design phase, a number of design decisions must be considered, and possible solutions need to be established. The GEA is programmed to provide three checking features for the user; each of these provides the user with simple and useful information to help him understand and solve any problems encountered while using the toolkit.

4.4.5.1 Error Checking

Error checking is one of the most important elements in assisting the user using the GEA. This is a proactive feature, which displays a warning message that advises the user that an illegal action is committed or incorrect information is provided. The dialog box is chosen to carry out this mission. Each dialog box provides the user with different information for helping him determine the error, or prevent making mistakes. Error-checking features may include checking for: information provided is insufficient, data violates domain definition, performed illegal operation, and so forth. Figure 4.27 illustrates an example of when a login failed; a hint is given and directs the user to pay special attention to the password and type (access privilege), and not the user name.

Figure 4.27: Error Checking
4.4.5.2 Redundancy Check

Some of the operations require a rather long execution time; therefore, unnecessary redundancy should be eliminated to assure efficient performance of the GEA. Also, redundant operations may destroy useful data by over writing, erasing, or partially modifying the existing data. For example, building a table for each tissue type demands the longest execution time in the GEA; unnecessary reconstruction of the table will seriously affect the performance of the toolkit. However, sometimes reconstructing the table is necessary because of new information updates; therefore, appropriate user judgment is required to make this decision. An example is shown in Figure 4.28 which illustrates the above argument. When a user wants to create a table for the brain tissue type which is already created in the database, a dialog box is displayed and asks the user whether s/he wants to replace the existing table. If the new information is not available to add to the table, this operation is considered as redundant, and the No button should be selected, or else the Yes button should be selected to refresh the table with up-to-date data.

![Figure 4.28: Redundancy Check](image)

4.4.5.3 Confirmation Check

A confirmation check is another very important feature that assists the user in working with this toolkit. Occasionally, the user may be confused and wonder whether an operation is carried out correctly or not. A confirmation message is returned to the user after certain operations are successfully executed. For example, the dialog box in Figure 4.29(a) shows that the system returns the confirmation message “rng is added to the system” to the user after s/he decides to add a new user “rng” to the system. This notifies the user that the Add User operation was properly carried out. Another type of confirmation check confirms and assures the actions of the
user, which makes sure the user really wants to commit this operation, and not just by accident. An example is shown in the dialog box Figure 4.29(b); if the user selects the exit operation in the GEA, the system checks with the user. If user decides to exit, the Yes button should be selected, or else the No button should be selected to terminate this action.

![Confirmation Check](a)

![Confirmation Check](b)

Figure 4.29: Confirmation Check

4.5 Other Features in the GEA

In the previous section, we discussed many features that are provided in the GEA that help the biology/cancer researchers identify a set of candidate genes and perform clustering analysis. Other supplementary features such as authentication, data management, administration, configuration, and help are also implemented in the GEA. These additional features can make the toolkit more complete and self-contained. We present these features in Appendix III and explain the usefulness of them.

4.6 Complications faced during the Implementation

4.6.1 How to Store the SAGE data to the Database?

During the development of the GEA, there are quite a few implementation issues that needed to be addressed. In the following, we provide an overview of some of the more interesting ones. Storing the gene expression data in the database is a fundamental operation of the GEA. Recall from the previous section that the SAGE data consists of 100 libraries, and each library contains more than 60,000 attributes, giving rise to an extremely high dimensionality situation. Like most
commercial Database Management Systems, DB2 can only handle up to hundreds of columns. Our solution to this problem is to “rotate” the table in the physical view. That is, while in the conceptual view, tags correspond to columns, and in the physical view, tags are stored as rows. This rotation forces the adjustment of certain standard relational operations. For example, if we apply a sum operation on a tag in the conceptual view, the value gives the total expression level of the tag in all libraries. However, in the physical view, the sum operation corresponds to a summation of all the entries in the row corresponding to the tag. Figure 4.30 shows how the database is rotated from the conceptual structure in (a) to the physical structure in (b).

<table>
<thead>
<tr>
<th>LibraryName</th>
<th>Tag(_1)</th>
<th>\ldots</th>
<th>Tag(_n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Library(_i)</td>
<td>Expr(<em>{\text{level}})(</em>{i1})</td>
<td>\ldots</td>
<td>Expr(<em>{\text{level}})(</em>{in})</td>
</tr>
<tr>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>Library(_m)</td>
<td>Expr(<em>{\text{level}})(</em>{m1})</td>
<td>\ldots</td>
<td>Expr(<em>{\text{level}})(</em>{mn})</td>
</tr>
</tbody>
</table>

(a)

<table>
<thead>
<tr>
<th>Tag</th>
<th>Library(_i)</th>
<th>\ldots</th>
<th>Library(_m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag(_1)</td>
<td>Expr(<em>{\text{level}})(</em>{11})</td>
<td>\ldots</td>
<td>Expr(<em>{\text{level}})(</em>{1m})</td>
</tr>
<tr>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
<td>\ldots</td>
</tr>
<tr>
<td>Tag(_n)</td>
<td>Expr(<em>{\text{level}})(</em>{n1})</td>
<td>\ldots</td>
<td>Expr(<em>{\text{level}})(</em>{nm})</td>
</tr>
</tbody>
</table>

(b)

**Figure 4.30:** (a) Conceptual Structure of the SAGE Data  
(b) Physical Structure of the SAGE Data
4.6.2 Limited Support from JDBC

Using JDBC as the API between DB2 and Java, is very convenient. However, there are some DB2 commands which are not supported by the JDBC driver, thus making the implementation more complicated. The DB2 commands, such as LOAD, EXPORT and others used for file manipulation, are not supported. A solution is to call these commands indirectly using the direct command execution to DB2. One disadvantage occurs after Java sends out this command; there is no way to keep track of when this command will finish its execution. Therefore, the approach we adopt is the WAIT method in Java in order to pause the Java program execution for a few seconds after the LOAD command is sent. Through this, we make sure that the table is properly loaded before any operation is invoked on the table.

4.6.3 Database Design

The database of GEA consists of a set of relational tables, and the information in the tables are sufficient to record the operations performed, and essential to support the operations. For completeness, and the references cited in this thesis, the schema of each of the tables in the database is presented in Appendix IV. The primary key for each of the tables is underlined.
Chapter 5

Conclusion and Ongoing Work

5.1 Conclusion

Gene expression analysis is one of the most active research areas in genomic analysis, and gene expression data are being produced at a phenomenal rate. The GEA presented in this thesis aims to provide a toolkit for cancer/biology researchers to conduct cluster analysis more effectively. A key feature of the GEA is the underlying algebraic framework, which allows intermediate results to be manipulated easily, via such structures as SUMY and GAP tables, and such operations as $\text{mine}()$, $\text{populate}()$, $\text{diff}()$, and so forth. On the one hand, this framework is consistent with the relational model, and provides integrated genomic analysis linking other databases, such as UNIGENE, KEGG, PUBMED, to name a few. This framework also effectively supports multi-step cluster analysis of gene expression data.

Apart from providing a description of the algebraic framework, we show in this thesis a few case studies using the GEA for identifying candidate genes for cancer research. On the system side, we introduce a lineage feature, which helps the user organize intermediate results more
effectively. We also study performance issues, particularly with respect to the high dimensionality of gene expression data.

5.2 Ongoing Work: Integrated Genomic Analysis

In the Chapter 4, we described the implemented GEA system. In this section, we describe ongoing work, specifically, how the GEA can be linked to other databases for integrated genomic analysis. Recently, the BC Genome Sequence Centre has started a project to build a repository integrating databases of various types. These databases include the following:

- **UNIGENE** (www.ncbi.nlm.nih.gov/UniGene), which can be used to map a tag to a gene;
- **SWISSPROT** (www.ebi.ac.uk), which can be used to relate genes to proteins;
- **PFAM** (www.pfam.wustl.edu), which classifies proteins into functional families;
- **KEGG** (www.genome.ad.jp/kegg), which gives information on genetic and biochemical pathways;
- **GENBANK** (www.ncbi.nlm.nih.gov/Genbank/GenbankOverview.html), which contains information on nucleotides;
- **PUBMED** (www.ncbi.nlm.nih.gov/entrez/query/static/overview.html), which is a literature and patent database;
- **OMIM** (http://www.ncbi.nlm.nih.gov/omim), on human diseases
- Other databases drug interactions, bimolecular interactions, and more.

In the following, we show a few examples of how these databases can be linked with the GEA to provide even more information to the user. The key point is that because the GEA model is consistent with the relational model, all the linkages between the GEA and these other databases can happen in a traditional querying environment, mainly through join queries.
5.2.1 UNIGENE

In the first example, recall that the GEA provides analysis on the basis of tags. To understand the biological meaning of a tag, the first step is to use the UNIGENE database. The UNIGENE information is obtained by partitioning the GENBANK sequences into a non-redundant set of gene-oriented clusters. Each UNIGENE cluster contains sequences that represent a unique gene, as well as related information, such as the tissue type, in which the gene has been expressed, and its map location. To look up the gene corresponding to a tag, the following relational algebraic expression suffices: \( \text{GeneRel} = \pi_{\text{Unigene.gene}} (\sigma_{\text{TagRel.tag}=\text{Unigene.tag}} (\text{TagRel} \bowtie \text{Unigene})) \), where TagRel is a relation with a column on tags (e.g., a SUMY or GAP table), and Unigene is the relation containing a mapping from tags to genes.

5.2.2 SWISSPROT

The user can then map the genes to protein sequences, via the SWISSPROT database. The SWISSPROT database contains not only the annotated protein sequences, but also up-to-date biochemical information accompanying the sequence data. For instance, taking the result relation GeneRel from the previous example, the expression: \( \text{ProtRel} = \pi_{\text{Swissprot.sequence}} (\sigma_{\text{GeneRel.gene}=\text{Swissprot.gene}} (\text{GeneRel} \bowtie \text{Swissprot})) \) retrieves the associated proteins.

5.2.3 PFAM

Many proteins from different origins may share similar functions. The PFAM database provides data for the user to determine other similar protein sequences in the same protein family; the comparison of their similarities and differences is sure to be useful. For two similar protein sequences in the same family, two proteins may be revealed to share the same function because they have the same domains, even though they occur in different compartments of the cell, or even in different organs. For each new protein sequence, PFAM data can be used to classify it in an existing protein family, even if the homology is weak. It is a database of multiple alignments...
of protein domains or conserved protein regions. The alignments represent some evolutionary conserved structure, which has implications for the protein’s function. A detailed examination to the protein sequences will give the user useful insight to questions such as, ‘if the protein sequences are in the same family, do they have the same function?’ Again this can be done as a joint query using ProtRel obtained above.

5.2.4 KEGG

Another useful piece of biological information that can be derived using the protein sequence is the genetic and biochemical pathway of the gene. By using the Kyoto Encyclopedia of Genes and Genomes (KEGG) database, we can determine the biochemical pathways the gene is involved in, and the roles it plays. Also the KEGG will help to identify other genes and molecules in the same pathway, for which the comparison between these networks of genes will be useful. The KEGG is a database containing molecular and cellular biology information such as information pathways that consist of interacting molecules or genes. It also provides links from gene catalogs produced by genome sequencing projects. This information can be obtained by joining ProtRel with the KEGG database.

5.2.5 GENBANK

For each of the candidate genes found, the user can identify its DNA sequence using the GENBANK database. By comparing their sequences, if the corresponding DNA sequence shows a high degree of similarity to a newly discovered gene, it might become possible to determine the possible function of the gene. Also, diseases can be caused by mutations in DNA sequences, because ‘normal’ DNA sequences are different from diseased DNA sequences. The GENBANK database is the National Institutes of Health genetics sequence database, an annotated collection of all publicly available DNA sequences, which consists of an ordered sequence of nucleotides. This database gathers the most up to date and comprehensive DNA sequence information on more than 55 thousand different organisms from various laboratories and projects.
5.2.6 OMIM

To determine what human diseases a gene can cause, the Online Mendelian Inheritance in Man (OMIM) database can be integrated with the GEA. The OMIM database is a catalog of human genes and genetic disorders, with links to literature references, DNA sequence records, gene maps, and clinical synopses. The GEA can use the information on the OMIM to answer questions, such as ‘what human genes are related to hypertension’, and ‘which of those genes are on chromosome 17’?

5.2.7 PUBMED

These examples can go on and on, but the final example is a less conventional one. For a particular gene or protein sequence, many studies may have been published. It is important for the user to be aware of these publications. To this end, the PUBMED database can be used. It includes bibliographic information, such as journal names, page numbers, and so on. The results from querying this database using GeneRel or ProtRel produce a list of publications, linking the user to full-text journal articles at the appropriate web sites.
Bibliography


[HK01] J. Han, M. Kamber: Data Mining Concept and Techniques, pg. 134, 2001


[HTM] The Human Transcriptome Map is available at http://bioinfo.amc.uva.nl/HTM/


[LGG01] N. M. Luscombe, D. Greenbaum, and M. Gerstein. What is Bioinformatics?


[NH94] Raymond T. Ng and Jiawei Han. Efficient and Effective Clustering Methods for Spatial Data Mining. In Proc. VLDB, pp. 144-155, September 1994.


Appendix I

Graphs

Figure AI.1: Microarray

Size of the SWISS-PROT Database

Figure AI.2: SWISS-PROT Database Statistic
Figure A1.3: The Procedures to Create Gene Expression Profile -- SAGE
Appendix II
Complications Encountered
When Port to mySQL

In this appendix, we discuss some of the complications encountered when porting the database from DB2 to mySQL. After a detailed investigation, we discovered that the complications might be contributed by three areas, which include JDBC, mySQL, and GEA itself.

As mentioned in Chapter 4, JDBC is the application programmer’s interface for connecting DB2 and Java in the GEA application. JDBC has many dedicated drivers, and each adapts to different DBMS, such as Oracle, DB2, and mySQL. However, we discovered that the JDBC driver for mySQL is not fully functional, or tested, yet. Moreover, the author claims there may be defects in the JDBC driver for mySQL. Since the driver is not defect free, many unknown and unexpected problems may be encountered during porting.

MySQL is a DBMS that is publicly available for downloading from the Internet, and many individual users are taking advantage of this. Therefore, we would like to modify the GEA to operate on mySQL, so that more users can employ the GEA despite the high cost of commercial DBMS. However, mySQL has some discouraging limitations. First, mySQL does not support Feign Key Constrain, which is implemented in the GEA database structure. Second, mySQL does not support the data export utility. This utility is critical to the GEA because massive amounts of data manipulation are required, and the performance of this application is significantly affected. Finally, mySQL has a very different syntax for load utility and a slightly varied SQL syntax, which requires the modification of many parts of the code for the GEA application.
Due to the complications involved in the JDBC driver and mySQL DBMS, to modify the GEA to support mySQL, major modifications to the java code are needed. Currently, GEA does not have any compile errors. However, some unknown errors occur in runtime when connected to mySQL; we suspect both the JDBC driver and mySQL create these problems.

As complications are caused by external applications, the GEA has difficulty accommodating them. After discussions and evaluations, we suggest deferring DBMS porting to future work. We conclude that at the current stage, it is not feasible to attempt this.
Appendix III
Supplementary Features

In Chapter 4, we discussed many special features provided in the GEA that help biology/cancer researchers identify a set of candidate genes and perform clustering analysis. In this appendix, we discuss some of the supplementary features that enhance the GEA and allow it to be more complete. The supplementary features included are authentication, data management, administration, configuration, and help. These additional features can strengthen the toolkit; make it more complete and self-contained.

AIII.1 Authentication Features

The authentication feature in the GEA is implemented for two specific reasons. First, the GEA supports multiple user access; the user can only access the data under her own working space. The user can work at her own pace; in order to extract appropriate information and deliver it to the user, the data and operations performed by the user are identified in the database based on the unique identification of the user. Second, the GEA can be installed in any open environment, such as a network that can be accessed by anyone. In an unsecured environment, the gene expression data and its results are vulnerable to malicious users, who may steal, delete, and modify valuable information. To prevent unauthorized access and malicious attacks on the data in the GEA, it is important to provide additional authentication features.

AIII.1.1 Login, Logout, and Exit
Every user who wants to access the GEA must first login to it. The access privilege of the user is based on his/her user name and password. In the GEA, there are two levels of access: one is the system administrator, who is authorized to have full access privileges to all operations. The other is the system user, who has a lower level of access, and can only access certain assigned operations. The administrator is responsible for assigning passwords and authorizing access privileges to each user. Figure AIII.1 is an example of a login window; users are required to input their user names, passwords and levels of access, which is either “administrator” or “system user”. The GEA searches the database to verify the login information when the user clicks the OK button. Also, before the user is successfully logged in to the GEA, only the File and Help Menus are available to the user in the main menu window.

![Login Window](image)

**Figure AIII.1: Login Window**

When a user is successfully logged in to the GEA, according to the authorized privilege of the individual, specific operations are granted to the user in the main menu. Figure AIII.2 is an example that shows that the system user “Jessica” is successfully logged in to the GEA. She is allowed to access most of the operations, except for the operations under Administration and Setting, which can only be accessed by a user with “administrator” privileges.
However, when a login fails due to an incorrect user name, password or level of access, a warning dialog box with detailed information is displayed to notify the user. The user can re-enter her login information and try again. Also, when the user can not login to the GEA, no access to other operations are granted in order to prevent unauthorized access to the data in the toolkit. Figure AIII.3 illustrates an example of a faulted login. When the user finishes working on the GEA, logout is mandatory to ensure all data are properly saved to the database and access to operations is terminated. After a user is logged out, another user can login to the GEA using his username and password. The user can also select Exit to shut down the GEA.
AIII.2 Data Management

The gene expression data is generated at a pronominal rate; the data set used for analysis in the GEA also requires periodic updates to reflect changes in the real world. A set of database management operations is available to help the user efficiently manage her database; the menu option is displayed in Figure AIII.4. A detailed discussion of the history of operations can be found in Chapter 4. The GEA also provides the operation for the user to modify data in a file.

![Figure AIII.4: Database Manage](image)

AIII.2.1 Initialize, Load, and Delete Database

The initialize operation provides the user with the ability to reinitialize the entire database by deleting all the user-created tables, as well as the data in the system-required tables. Only the GEA system required table remains with some default information, and no data is populated to any of the tables. When the error removal (data cleaning) process is completes (as discussed in Section 4.2), the user can apply a load operation to populate the library information to the Libraries table and the cleaned gene expression data to the TAGS table. The delete operation is used to remove the data in both Libraries and TAGS tables, which update the analytical database.
In general, the load and delete operations are used to maintain the gene expression data in the database.

### AIII.2.2 File Edit Operations

For the completeness, the GEA includes a simple text editor for the user to edit text documents with. The simple text editor provides the functions of open, edit, and save. Similarly to an ordinary text editor, it provides a simple text area for displaying text, and a self-explanation menu. The GUI in Figure AIII.5 shows the menu options available in the text editor; its text area is displayed as empty.

![Simple Text Editor](image)

**Figure AIII.5: Simple Text Editor**

By selecting the Open operation in the pull-down menu, a browse window is opened for the user to select the document that s/he wants to open for editing. The example in Figure AIII.6 shows the user opening the document `sageName.txt` in the directory `SageLibrary`. To select the document to open, simply double click the name of the document in the browse window, or highlight the document and then select the **Open** button.
When the document is opened for editing, its contents are displayed in the text area of the simple text editor. Figure AIII.7 shows the content of sageName.txt in the text editor. The sageName.txt saves the information of SAGE libraries; the user can change the content of the document by adding, deleting and modifying the file. The basic editing functions provided are cut, copy, paste, select all, and recent cuts. These functions allow the user to easily manipulate text documents while applying analytical operations in the GEA.
AIII.3 Administration Features

A set of administrative features is implemented in the GEA to manage user information. The system administrators who hold higher access privileges are given the tools to manage other system users who have lower access privileges. The operations available for administration users are illustrated in Figure AIII.8, and include the following:

1. Setup account for new users, and assign appropriate access privilege
2. Remove a user account from the GEA database
3. For existing users, the administrator is allowed to modify account information for the user, such as password and user access privilege.

Figure AIII.8 Administration Feature

AIII.3.1 Add New User Account

The administrator has the authority to create a new user account in the GEA. The GUI for Add New User Account is very similar to the GUI for Login, which is shown in Figure AIII.9. The administrator needs to assign a user name, a password, and an appropriate access privilege to the new user account. Figure AIII.9 shows the administrator creating a new user account for “cfu”, where the password is encrypted by the symbol “*” in the password text box, and a system user
access privilege is assigned. By clicking on the Add button, a new user account is created in the database for the new user.

![Add User](image)

Figure AIII.9: Create New Account

### AIII.3.2 Delete User Account

The administrator can also delete user accounts from the system due to various reasons, such as when a user account is expired or the user violates the rules of the system. A delete user account window is shown in Figure AIII.10; all the user names are listed in the User List on the left-hand-side of the window. A double click selects a user who requires deletion from the GEA, and information, including access privilege and user name is displayed on the right-hand-side of the window. Alternatively, the administrator can enter the information directly, and then click the Delete button to remove the account from the database. A confirm message prompts the user if the account is successfully deleted, or else an error message is displayed to the administrator.

![Delete User](image)

Figure AIII.10: Delete User Account
AIII.3.3 Modify User Information

For each existing user account, the administrator has the authority to modify the information of these accounts. An administrator could also change the password of a user account. As shown in Figure AIII.11, the administrator changes the password for the system user "cfu"; if the administrator changes the Type from U to A, the system user "cfu" is promoted to an administrator and obtains the administrator access privilege. When the administrator clicks the Modify button, the modified information is updated in the database.

![Modify User Information](image)

Figure AIII.11: Modify User Information

AIII.4 Configuration Feature

To allow the GEA to function properly after installation, a configuration process must occur. This process is often not the favorite task of a user. A typical configuration process may involve setting a class path, a library path, and other specifics. Only a user with administrator privileges has the authority to change the configuration data of the system. The configuration operation is under the Setting menu.

Since the GEA uses the database management system DB2, some information regarding the DB2 requires correct configuration. For the convenience of the user, a GUI is implemented to acquire the information and incorporates it into the toolkit. Figure AIII.12 is an example of the
configuration window for the GEA. The information required for the configuration includes the
following:

1. UserID and password for access DB2;
2. The name of the current database used by the GEA;
3. The search path for the DB2 executables.

The user can select the **Default** button to set the information to the default values, or supply new
information and select the **Apply** button for reconfiguration. The configuration operation is often
carried out during the installation of the software.

![Configuration Window](image)

**Figure AIII.12: Configuration Window**

### AIII.5 Help Feature

A Help operation is provided in the GEA to assist the user when problems are encountered; it
also serves as a tool to help the user understand and use the toolkit. The user can find detailed
explanations of the GEA, the Fascicle algorithm, and some background information of
bioinformatics and data mining. An index is also provided to the user for searching. This index
includes technical terms and operations available in the GEA. Figure AIII.13 shows the help
features provided in the GEA.
Figure AIII.13: Help Features
Appendix IV
Database Design

1. This table provides the compact dimension threshold for each tissue type in SAGE.
   CDInfo (Type String, Threshold integer)

2. This table stores the result of the union or intersection between two GAP tables.
   CompFile_U_I (TagName String, TagNo integer, GapValue1 double, GapValue2 double)

3. This table saves the differences between two GAP tables.
   CompFile_D (TagName String, TagNo integer, GapValue double)

4. This table is reserved for the Expression Analysis Database.
   EADB()

5. This table keeps track of all the fascicles created by the user.
   FasFile (UserName String, FasName String, Type String, FasCD integer, FasBinary String, FasMeta String, FasBatch integer, FasMin integer)

6. This table keeps track of the property information of each fascicle:
   FasInfo (UserName String, Fascicle String, FasName String, Cancer integer, Normal integer, BulkTissue integer, CellLine integer)

7. This table keeps track of the libraries contained in each fascicle.
   fasLib (UserName String, Fascicle String, Lib_ID integer)
8. This table keeps information of gap comparison.
   GapCompInfo (UserName String, CompFile String, Type String, Gap1 double, Gap2 double, CompType String)

9. This table saves the gap information including type and the summary tables used to calculates it.
   GapInfo (UserName String, GapName String, Type String, Flag integer, Sum1 String, Sum2 String)

10. This table saves the gap values calculated between two Summary tables.
    GapTable (TagName String, TagNo integer, GapValue double)

11. This table provides the information to link gene to protein.
    GenePro (Gene String, Protein String)

12. This table provides the information to link gene to publications
    GenePub (Gene String, Pub String)

13. This table stores the libraries information.
    Libraries (Lib_ID integer, Lib_Name String, Type String, CANNOR integer, BTCL integer, Tag integer, Utag integer)

14. This table saves the information of the total number of library and tags of the SAGE data.
    SageInfo (Totag integer, ToLib integer)

15. This table keeps the information of the SUMMARY tables.
    SumInfo (UserName String, SumTable String, Fascicle String, Category String, Sign integer)

16. This table provides the information of the libraries consist in each summary table.
SumLib (UserName String, SumTable String, Lib_ID integer)

17. This table saves the aggregate values, including min, max, range, average, and standard deviation for each of the tags in a summary table.

   SummaryTable(TagName String, TagNo integer, minimum integer, Maximum integer, Range integer, Average double, STDV double)

18. This table stores the system configuration information.

   SysConfig(DB2ID String, DB2PWD string, DB2DB String, DB2PATH String)

19. This table saves the gene expression data (SAGE).

   TAGS (TagName String, TagNo integer, Lib_a integer, ..., lib_b integer)

20. This table provides the information to link tag to gene.

   TagInfo (TagName String, TagNo integer, Gene String)

21. This is the ENUM table for each of the tissue type.

   TissueTypeTable (TagName String, TagNo integer, Lib_a integer, ..., Lib_m integer)

22. This table keeps the topGapFile information and the top gap created.

   TopRec (UserName String, TopGapFile String, GapName String, TopNo integer)

23. The table saves the top gap values for each GAP table.

   TopGapFile (TagNo integer, TagName String, GapValue double)

24. This table stores the libraries in each tissue type and its order.

  TypeInfo (Type String, Lib_ID integer, Order integer)

25. This table provides the information of the ENUM table name and its metadata file.

   TypeCreateInfo (UserName String, Type String, TableName String, Flag integer)
This table saves the userID and its password.
Userinfo (Type String, UserName String, Password String)
Algorithm
A process, or set of rules, usually one expressed in algebraic notation, now used especially in computing, machine translation and linguistics.

Amino acid
The basic building block of proteins. All proteins are made of one or more chains of amino acids. There are 20 different kinds of amino acids that are used to make proteins. They all have different but closely related chemical structures.

Base
A single unit of DNA. There are four possible bases, represented by the letter A, C, G, and T. A piece of DNA is made of a series of bases linked in a chain.

Bulk tissue
A group of cells, which were taken directly out of tissue in a person’s body. Bulk tissue may contain several different types of cells in different amounts.

Cell line
A cell which can grow indefinitely in a test tube or petri dish, making an infinite number of copies of itself.

Chromosome
Self-replicating structures of cells that carry in their nucleotide sequences of the linear array of genes. There are 23 chromosomes for the entire human genome.
Cluster
The grouping of similar objects in a multidimensional space. Clustering is used for constructing new features, which are abstractions of the existing features of those objects.

Clustering
Data mining techniques that apply clustering algorithms find groups of items that are similar.

Configuration
(In software) The complete ordering and description of all parts of a software or database system. Configuration management is the use of software to identify, inventory and maintain the component modules that together comprise one or more systems or products.

Co-regulated
Gene that are co-regulated are always expressed at the same rate.

Data mining
An information extraction activity whose goal is to discover hidden facts contained in databases. Using a combination of machine learning, statistical analysis, modeling techniques and database technology, data mining finds patterns and subtle relationships in data and infers rules that allow the prediction of future results.

Distance function
Functions calculate the distance between two points. A common distance function is the Euclidean distance.

Entropy
A way to measure variability other than the variance statistic. Some decision trees split the data into groups based on minimum entropy.
DNA
It stands for deoxyribonucleic acid. The chemical that forms the basis of the genetic material in virtually all organisms.

DNA microarray
The microarray measures the relative levels of mRNA abundance between different tissue samples.

Expressed Sequence Tags (EST)
EST stands for Expressed Sequence Tags. This method collects gene expression data in complete segments of mRNA.

Expression (gene or protein)
A measure of the presence, amount, and time-course of one or more gene products in a particular cell or tissue. Expression studies are typically performed at the RNA (mRNA) or protein level in order to determine the number, type, and level of genes that may be up-regulated or down-regulated during a cellular process, in response to an external stimulus, or in sickness or disease.

Expression profile
The level and duration of expression of one or more genes, selected from a particular cell or tissue type, generally obtained by a variety of high-throughput methods, such as sample sequencing, serial analysis, or microarray-based detection.

Euclidean distance
A straight-line distance between two points.
Gene
Classically, a unit of inheritance. In practice, a gene is a segment of DNA on a chromosome that encodes a protein and all the regulatory sequences (promoter) required controlling expression of that protein.

Gene expression
The conversion of information from gene to protein via transcription and translation.

Gene expression profile
The genetic makeup and expression profile of a cell determines its fate as a normal or cancer cell. Each type of cell expresses a unique set of genes, and the degree to which those genes are expressed can be quantified and displayed in a gene expression profile.

Genome
The complete genetic content of an organism.

Hierarchical clustering algorithm
Methods to represent similarity matrix as a graph, form a separate cluster around each node, and traverse the edges in decreasing order of similarity, merging two clusters according to some criterion. For example, Dendrograms used this representation.

Homology
Two or more biological species, systems or molecules that share a common evolutionary ancestor. Two or more gene or protein sequences that share a significant degree of similarity, typically measured by the amount of identity (in the case of DNA), or conservative replacements (in the case of protein), that they register along their lengths.

Housekeeping genes
Genes that are always expressed due to their constant requirement by the cell.
Leukemia
Cancer of the bone marrow.

Library
A large collection of compounds, peptides, cDNAs or genes which may be screened in order to isolate cognate molecules.

Lymphoblastic Leukemia
A type of blood cancer.

Lymphocyte
A type of white blood cell.

Lymphoma
Cancer of the lymph nodes.

mRNA
Messanger RNA; the complementary RNA copy of DNA formed from a single-stranded DNA template during transcription that migrates from the nucleus to the cytoplasm where it is processed into a sequence carrying the information to code for a polypeptide domain.

Multiple sequence alignments
A Multiple Alignment of k sequences is a rectangular array, consisting of characters taken from the alphabet A, that satisfies the following conditions: there are exactly k rows; ignoring the gap character, row number i is exactly the sequence si; and each column contains at least one character different from ".-".

Mutation
An inheritable alteration to the genome that includes genetic (point or single base) changes, or larger scale alterations such as chromosomal deletions or rearrangements.
Myeloid Leukemia
A type of blood cancer.

Neoplastic state
Whether a cell is cancerous, pre-cancerous, or normal.

Nucleotide
A nucleic acid unit composed of a five-carbon sugar joined to a phosphate group and a nitrogen base.

Pattern
Molecular biological patterns usually occur at the level of the characters making up the gene or protein sequence.

Pathway
Bioinformatics strives to define representations of key biological data types, algorithms and inference procedures, including sequences, structures, biological pathways and reactions. A pathway analysis of whole genomes including identifying common patterns across species and species differences.

Parameters
Parameters are user-selectable values, typically experimentally determined, that governs the boundaries of an algorithm or program.

Protein
The chemical building blocks from which our cells, organs, and tissues like muscle are made. Proteins also serve double-duty as hormones, enzymes and antibodies, which help our bodies fight off invading germs. Proteins are made of long chains of even smaller building blocks called amino acids.
Protein family
Sets of proteins that share a common evolutionary origin reflected by their relatedness in function, which is usually reflected by similarities in sequence, or in primary, secondary or tertiary structure. Subsets of proteins with related structure and function.

SAGE Library
The data product of the SAGE technique is a list of tags, with their corresponding count values, which are the expression levels of the tags.

Tag
A tag represents the transcription product of a gene; it is a nucleotide sequence of 10 base pairs (which is a sequence of the 4 bases A, C, G, and T). A tag has a many-to-one relationship with a gene.

Tag-to-gene mapper
A tool to identify its corresponding gene for a tag.

Tissue
Section of an organ that consists of a largely homogenous population of cell types. Since many organs are multifunctional, they have developed highly specialized cell types to perform different functions. Identifying the section of an organ that is homogenous for a particular cell type ensures that the gene expression profiles extracted from those cells will accurately resemble the class of cells that make up the tissue.

Transcript
The single-stranded mRNA chain that is assembled from a gene template.

Transcription
The assembly of complementary single-stranded RNA on a DNA template.
Transcription factors
A group of regulatory proteins that is required for transcription in eukaryotes. Transcription factors bind to the promoter region of a gene and facilitate transcription by RNA polymerase.

Translation
The process of converting RNA to protein by the assembly of a polypeptide chain from an mRNA molecule at the ribosome.

Total Tags
The sum of all the count values of all the tags in a library.

Unique Tags
The number of different tags that were detected in the sample.

Visualization
Visualization is the process of representing abstract scientific data as images that can aid in understanding the meaning of the data.