# Gene Expression Analysis in Multi-Agent Environment

H.C. Lam[1], M. Vazquez Garcia[2], B. Juneja[3], S. Fahrenkrug[3], and D. Boley[1]

[1] Department of Computer Science, University of Minnesota, Minneapolis, MN 55455
[2] Departamento de Sistemas Informaticos y Programacion, Universidad Complutense de Madrid, Spain
[3] Department of Animal Science, University of Minnesota, St.Paul, MN 55108

**Abstract.** This paper presents a multi-agent approach to gene expression analysis and illustrates the working steps using real dataset produced from a microarray experiment. The analysis can be conducted in a multi-agent environment with majority of the workload automated by agents at different stages of processing. Conceptually, the analysis process can be divided into three distinct steps: (1) data preprocessing, (2) statistical analysis, and (3) biological inference. In addition, each step involves different stages of processing. We implement a multi-agent system that provides process automation within each step using agents with different abilities. We also explore the parallelism of data processing using multi-agent system. The strength of our system is in its ability to support concurrent processing of gene data and the modular structural design that tailors to biologists' demands. We also discuss the possibility of applying Machine Learning technique to analyze gene expression data in a multi-agent environment.

**Keywords:** microarray, gene expression analysis

## 1 Introduction

Microarray technology has been very effective in producing large amount of gene expression data by measuring the transcription levels of thousands of genes simultaneously. A typical microarray experiment can generate up to millions of data points for analysis. However, this highly systematic data analysis process can be broken down into stages and each stage can be automated with very minimal human intervention. We consider one such case in which oligonucleotide microarrays were used to investigate stress responses from thousands of porcine islet genes.

Conceptually, there are three categorical data processing steps in gene expression analysis: (1) data preprocessing, (2) statistical analysis, and (3) biological inference, as illustrated in figure 1. Microarray data is preprocessed to ensure the quality and the comparability of the data. Statistical analysis is followed to find significant genes. The gene lists selected from statistical analysis will be input into various biological resources, for example, gene ontology databases, pathway databases, literature databases, and/or verified by biological experiments. The results inferred from these biological resources might agree with the experiment and the statistical analysis, or might not. In the latter case, it is possible that the information from the resources needs to be further curated or the statistical analysis needs some adjustment. The final results are then drawn after verification and logical interpretation from the researchers. The categories mentioned above can be automated using a multiple agents system.

In this paper, we present a multi-agent approach to conduct gene expression analysis using real dataset to emphasize that the majority of the analysis can be carried out using agents. The system supports concurrent gene data processing by implementing independent agent solutions and interface agent solutions to coordinate data flow. One of the common concerns and is often overlooked by researchers is the effect of intermediate data processing has on the data as it travels through the processing pipeline. We hope to address this problem using a multi-agent system and through a case study to help users in better selecting subsequent method of analysis.

This paper is organized as follows. Section 2 introduces multi-agent systems in the field of gene data analysis and provides a brief description of the data set used in this paper. Section 3 then presents the architecture of the multi-agent system in the context of gene expression analysis. In Section 4, we take a look at different agent's responsibilities in various stages of the analysis process. For Section 5, we illustrate an example of gene expression analysis using real data set to demonstrate automated processing by different agents in a coordinated manner to accomplish various tasks. In section 6, a brief discussion of using multi-agent learning technique to analyze gene expression data is provided. Finally, we conclude with Section 7 with discussion and open questions.

## 2 Related work

Multi-agent systems have been applied in various disciplines of computer science, from Artificial Intelligence to networking of dynamic systems. In the field of gene expression data analysis, the focus is placed on finding biological significant genes using a combination of ad-hoc approaches without considering

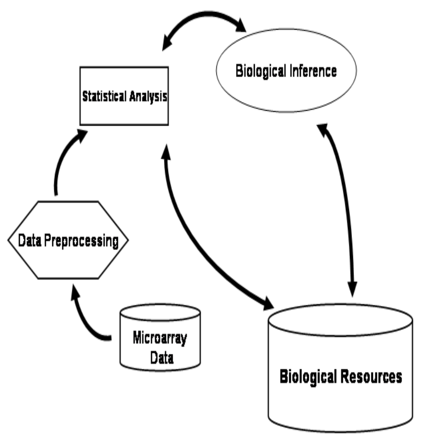

**Fig. 1.** Conceptual diagram of gene expression analysis

the underlying dependency of each data processing step. This is one of the reasons adopting a multi-agent system to tackle gene expression analysis in light of gathering the strength of each independent analysis method within a central processing environment to hunt for genes of interest. The other reason is that within a multi-agent environment, one only needs to interact with, in our case, the Query Agent, to setup the processing job. Thus, time and resources are better used and users of the system can focus on other tasks. Several research groups have come up with different agent based system to perform genomic study. BioMAS [13] is a multi-agent system primarily to handle gene information retrieval from different online sources and integrates them to find common pathways that are shared by human and chicken. GeneWeaver [3] is based on a multi-agent system in which each agent takes on a particular responsibility or expertise of genome analysis, such as assignment of functional descriptions to the proteins and performing homology searches. The agents coordinate their activities by sending messages to each other to accomplish overall tasks. A central control module is built into each agent in GeneWeaver system that decides how best to achieve its goal. Unlike our system, each agent only implements a designated algorithm to achieve its goal in processing the gene data. A Master agent is charged with decision making task and to set up a work flow path for the entire processing job. This way, each agent has a clear role and task and integrating of additional agents becomes transparent to existing agents in a multi-agent environment.

We used the dataset from the study of transcriptional profiling of stress-response in cultured porcine islets. This study used two color DNA microarrays to evaluate both physiological and molecular responses of isolated porcine islets to inflammatory cytokines and glucose culture conditions that model hyperglycemic and immunological stress encountered by islets after xenotransplanation [6]. A thorough understanding of genes expressed during islet isolation and culture could improve islet preparation, thus accelerating clinical application [6]. The long term goal of the study was to investigate the use of porcine islet as an alternative source of donor tissue for treating type 1 diabetes. The experiment used 39 microarray chips with more than fourteen thousands cDNA sequences hybridized to the probes on each chip. The three treatment conditions were glucose stress effect, cytokine stress effect, and the interaction of glucose and cytokine effect. The microarray chips were scanned using a ScanArray 5000 scanner. Raw data were generated by the Genepix 5.1 software (Axon Instruments Inc, Union City,CA).

## 3 Multi-agent System Overview

From a technical standpoint, gene expression analysis actually utilizes many resources and the result can be difficult to interpret. There is not a fixed way to do an analysis and most often researchers will try different strategies to analyze their data set. In light of this, our goal is to provide a system that satisfies this need by an adaptive multi-agent solution. The multi-agent architecture is shown in figure 2. The system is composed of various interface agents and individual working agents. There is a Master Agent (MA) which is the core of the system that is responsible for various tasks and the coordination of agents. The only human intervention to the system is when specific parameters are required to initiate the analysis and it is handled by the *query agent*. Once the parameters are available, the *master agent* will select the appropriate data processing path to perform the expression analysis. A data processing path in our case is a path that a given gene data goes through from preprocessing to biological verification in different stages of the process. Conceptually, these different stages make up the pipeline of gene expression analysis. We demonstrate a case study to illustrate the processing pipeline in Section 5. We first discuss what will take place in the system in the context of gene expression data analysis to provide a basic understanding of how the proposed system operates.

Once the microarray experiment is finished and the necessary data are acquired, the system starts with a user interacting with the query agent to input the parameter setting for the analysis. The query agent than passes the analysis parameters through the *User Interface Agent (UIA)* to the master agent. The master agent will then decide what data process-

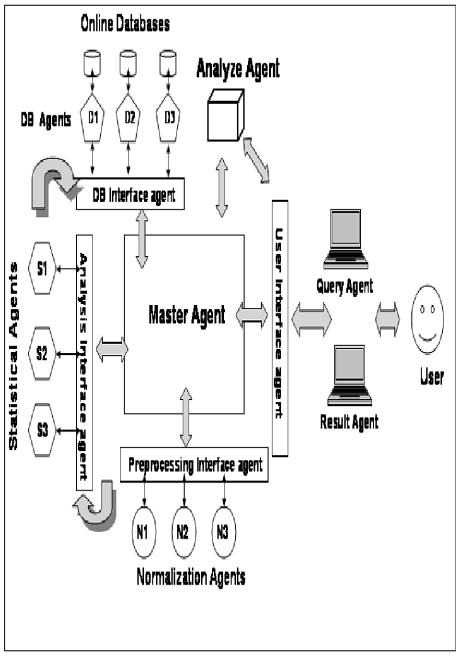

**Fig. 2.** Multi-agent system architecture

ing path the gene data will travel in the system. Once the data processing path has been determined, the *master agent* will hand the data to the *Preprocessing Interface Agent (PIA)* which in turn will send the data to a *normalization agent (N)* to perform data normalization if specifically requested. After normalization, the *PIA* collects the data output from the normalization agent and sends it to the *statistical analysis agent (S)* through the *Analysis Interface Agent (AIA)*. The statistical analysis agent will perform requested statistical analysis with the data and output a gene list with genes that are found to have statistical significant expression values. This gene list can then be sent to *DB interface agent (DBIA)*. The DBIA then invokes the proper DB agent for functional analysis. Once the result become available, DBIA passes the result back to the Master agent and it then will forward it to the user interface agent. The UIA sends the result to the *Result Agent* which will format and output it to the user. The user can then make the final verification and conclusion based on the result derived by the system.

Our multi-agent system is built on top of the Bioconductor[2] platform where a wide range of statistical and graphical methods for the analysis of gene data can be incorporated. Besides taking advantage of already existing functionalities provided by the Bioconductor environment, our system provides an effective way for users to analyze their data through the use of multiple processing path which can carry out various analysis concurrently (see case study with data normalization as an example) whereas other software packages like GeneSpring[11] and GeneData Expressionist[10] lack. The inherit parallelism of gene data processing our system provides not only speeds up the process but also gives vital result comparison to help a user makes better decision along the processing pipeline.

The modular design proposed in this paper also makes it easy to incorporate new agents carrying new tasks in the future. The agents implemented so far are devoted to the statistical analysis of the microarray data, but future agents could easily be designed to make biological inferences based on information obtained from databases of functional descriptions of genes. This could be used to indicate which biological processes might be exposed by the experiments, or to explore the space of input parameters to the system proposed here, allowing the system to adapt to the specific experimental conditions
present.

## 4 Agent and their Responsibilities

Gene expression analysis is a daunting task because there are many ways to perform the same analysis and get different results. Our design objective is to provide a flexible system that supports concurrent processing of data stream, a modular structure that sustains the diverse requirements of researchers in bioinformatics, and most importantly, user friendly by making the majority of the analysis automated. In this section, we outline the system components and their responsibilities.

### 4.1 Query Agent

The query agent is responsible in collecting users input and forwarding it to the UIA. The UIA then forwards the input to the master agent. Once the required information has been gathered, the query agent arranges the sequence of options chosen by the user as a colon delimited User Input Sequence (UIS): (BEGIN:NORMTYPE:W:NORMALG:G:SAVENORM : Y:STATS:E:SAVESTAT:Y:CHKDB:Go:END) and forwards it to the UIA. This is the only part where human intervention with the system takes place, once the master agent received the user input sequence, the system starts processing user's request.

### 4.2 Master Agent

The Master Agent is the central control unit of this multi-agent system. It distributes work to different agents, coordinates activities among agents, interacts with interface agents, and store agents information in its agent repository. The design logic behind the use of a central control agent is to enforce the objective where working agent works (data intensive work) and decision making agent only responsible for making decision and setting up work path. This way, the system is more well defined and structured. In order to distribute work to different agents, master agent needs to process the UIS coming from the UIA. If a user has specified for: (1) data normalization, (2)then perform statistical analysis, and (3)finally checking online functional databases for annotation purposes, the master agent will derive a data flow path that only the end result from (3) will be sent back to itself which will then be passed to the UIA. There is no intermediate involvement from the master agent. On the other hand, if a user prefers only data normalization done to the gene expression data, the master agent only interacts with the PIA and returns the result to the UIA when it is available.

Base on the user input sequence, master agent can coordinate various tasks among different agents in the system. It does so by issuing Master Command Sequence (MCS) to the interface agents. For example, if a user has specified using multiple statistical agents (T-Test, ANOVA, and E.Bayes) to perform the analysis on the same data set, then the master agent will issue a MCS: (BEGIN:AIA:S1:Y:S2:Y:S3:Y:END) to the AIA which in turn will activate the requested statistical agents.

One of the goals of our design is to provide a modular structure to the system without incurring unnecessary complexity to the system. Therefore, to tailor to different users' requirement, a working agent can be added to the system by registering with the master agent. The working agent only communicates with its corresponding interface agent after registration process. This is attractive since there is no need to change the overall system structure. The interface agents provide indirect communication between the working agents and the master agent. This allows the addition or removal of working agents without interrupting the system.

In order to maintain control of all the agents in the system, the master agent also implements an agent repository database which keeps all the available agents' information. The agent information is consisted of but not limited to: Agent Name(unique), Agent Service, Agent Status, and Agent Date. A new working agent can register with the master agent by providing its name and the service it implements. The working agents are dynamic system components, which means that they can either be registered with or deregistered with the master agent. To deregister, the corresponding interface agent needs to send an $agent - disable$ request to the master agent in order to update the $Agent - Status$ information in the repository.

### 4.3 Normalization Agents

Normalization agents(N) is a type of working agent that communicates directly with Preprocessing Interface Agent (PIA). The duty of a normalization agent is to perform the requested data normalization process. Each normalization agent in the system can be viewed as a unique working agent and each of these agents implements a single normalization algorithm. In figure 2, one can see three normalization agents (N1,N2,N3) and each corresponding to different normalization process and the agents are independent of each other.

### 4.4 Statistical Analysis Agents

Statistical analysis agent (S) is a type of working agent that implements a single statistical method. The input for this agent is from the AIA where it can accept data from the master agent directly or from the PIA if a user has specified data normalization beforehand. The output produced by this type of agent is formatted as a list. Depending on what type of statistical analysis a user has specified, the gene list will have different statistical values (P-value,t-value,B-value,etc) attached to each significant gene found. The output (gene list) produced by the statistical agent is then passed to the DB interface agent for further verification and analysis or directly back to the master agent through the AIA. Again, our system allows additional statistical agents to be implemented in order to provide diverse choices for analysis and create more data paths to accommodate users as different needs should arise.

### 4.5 DB Agents

The DB Agents are agents that interface with various online gene functional databases. It can supply the annotation for the genes output from the statistical agents or directly requested from the master agent if given a gene list. The purpose of using DB agents is to provide biological interpretation of genes. For example, an ontological analysis based on certain statistical models [14] can be implemented independently. This is helpful in finding over-represented gene ontology group in a group of statistical significant genes.

### 4.6 Result Agent

Result Agent is responsible for presenting the result generated by the system to the user. It communicates with the UIA directly. The result format is usually a list of genes found or a table of genes with statistical values attached, or a list of annotated genes.

### 4.7 Analyze Agent

Analyze Agent is an agent that interfaces with the master agent directly and can provide important services to users in order to help them make better interpretation of the result. The services include but are not limited to: comparing gene lists obtained by using different working agents, cluster genes, or even perform Principle Component Analysis on the gene data. For instance, if a user needs to compare different gene lists produced by using different preprocessing methods using correlation measurement, he or she simply takes advantage of the system by using query agent to specify more downstream analysis on the data.

### 4.8 Interface Agents

There are four types of interface agents implemented in the system: (1) Preprocessing Interface Agent (PIA), Analysis Interface Agent (AIA), DB Interface Agent (DBIA), and User Interface Agent (UIA). The purpose of using interface agents are twofold: (1) keep working agents independent from each other, (2) provides an indirect communication service between the master agent and the working agents. This way, the addition or removal of normalization agents or statistical agents in the system can be done seamlessly. We only need to update the master agent's knowledge about the new addition. The idea behind this design is to allow flexibility in the system while at the same time addresses different user's need by integrating new agents to the system. The PIA has the capability of directly sending any output data from its corresponding working agents to the AIA for analysis. And the AIA has the same feature which allows the output from statistical agents to be sent to the DBIA.

### 4.9 Communication

So far, we have seen the agents' responsibilities and how they interact in the system. In this section, we will briefly discuss the message format used for communicating with other agents in the system and the UIS and MCS format. There are two aspects of messaging in our multi-agent environment. The first one is the message format structure which is prepared by the working agents, the second is the message transport structure which can be prepared by the interface agents or working agents. An agent message has the sender field, receiver(s) field, message field (including specified parameters), and time stamp field. The interface agents prepare the transport message which encapsulates an agent message. Transport message only has the sender field and the receiver field. The intended receiver is either a master agent or an interface agent which will decode the message for processing. The format of UIS is a name and value(s) tuple where value(s) can be a single value or multiple values delimited by colon. The tuples can be concatenated to form a sequence enclosed in the BEGIN and END tags. The name field is the operation name, for example, normalization algorithm (NORMALG), save normalized data (SAVENORM), and statistical analysis (STAT). The value(s) field takes a single letter to indicate the desired choice based on the question presented by the query agent. As for MCS, the formulation is also based on a name and value(s) tuple like the UIS except the name field is used for designated agent name instead of operation name. It also must be enclosed in the BEGIN and END tags. The UIS and MCS are very simple messages that can be encapsulated into message format structure and transport structure discussed above that traverse in the system.

## 5 Gene expression analysis case study

In this section we present a case study scenario to illustrate the working steps of the multi-agent system using the dataset described in section 2. We look at genes that are expressed under the cytokine stress effect. Suppose a user entered the necessary parameters and the query agent created a UIS: (BEGIN:NORMTYPE:W:NORMALG:G:L:P:SAVENORM:Y :STATS:E:SAVESTAT:Y:END) and forwarded it to the UIA agent for processing. According to the input sequence, the requested services are: whole array normalization, select multiple normalization algorithms (Global, LOWESS, PinTip), save normalized results, perform EBayes analysis on the normalized data, save the EBayest results. The extra parameters required to do the normalization will be requested by the query agent subsequently. The corresponding MCS is: (BEGIN:PRE:N1:Y:N2:Y:N3:Y:AIA:S1:Y:END). In figure 3, a snapshot of the conceptual data flow path is shown:

### 5.1 Data normalization

The purpose of doing normalization is to adjust for any bias which is caused by microarray technology rather than biological differences between the RNA samples or printed probes on microarray chips [17]. In other words, normalization is to minimize systematic variation in the measured gene expression levels of hybridized mRNA samples so that biological differences can be more easily distinguished. For two color DNA microarray, the most common systematic bias is the color-effect produced by the difference in intensity response between the two dyes (Cy3 and Cy5) as a function of the level of expression. Cy5 (red) tends to be relatively more intense at higher and lower intensity levels than Cy3 [26]. Besides color-effect, other biases can come from variation between spatial positions on a slide or between slides. The robotic printing can also contribute to biased output due to pin-tip effects

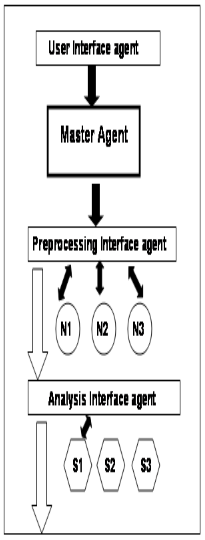

**Fig. 3.** Snapshot of gene expression analysis path

[26]. There are many approaches (Within-print tip normalization, Scale normalization, Composite normalization, Intensity dependent normalization, Global normalization, Local normalization) [18] to normalize expression data based on different hypotheses. Here, we outline some popular normalization schemes provided by the normalization agents. If a user has specified using global normalization, a supplement parameter $c$ is needed. The global normalization is done using the equation $log(R/G) - c$ where $R$ represents the red intensity values and the $G$ represents the green intensity values and $c$ is the mean or medium of the intensity log ratios $M$, where $M$ is the ratio of red foreground value over green foreground value. On the other hand, if a user has chosen the LOWESS normalization scheme, he needs to supply the smoother span parameter which indicates the proportion of points that control the smoothness of the curve at each value [5]. This smoothing technique suggested by Yang et al [26] is a function $L$ of the average intensity $A$ fitted to the scatter plot and then is used to normalize the log-intensity ratio $M$ by computing the difference $M - L(A)$. It emphasizes the contributions of data from array elements that are far from each point, thus generating a smoothing effect for the data points. One can also specify the subgrid normalization algorithm. Subgrid normalization tries to eliminate the biases due to the differences of the print tips in the printing of microarray chips by using a LOWESS fit to the MA plot for each block in the array independently [26].

## 5.2 Finding statistical significant genes

Microarray experiments usually produce enormous amount of data points for biologists. The statistical task is hopefully to efficiently and effectively reduce the amount of data into a significant group of data representing meaningful gene functions. Many sophisticated statistical methods have been developed and can provide useful analytical tools for researchers to find genes that have biological significant [4,12,16,25,24]. The most simple statistical analysis is the twofold changes method. This method does not account for the possibility of variations of log ratios for different genes. The Student $t$ statistics is another common method in use for finding significantly expressed genes. There are several versions of the t-test, depending on the sample size and the assumption of expression levels having equal variance under different conditions. Another popular method is the Empirical Bayes [20] method which we will be using in this section as an example. We generated gene lists using statistical agent implementing Empirical Bayes (EB) method discussed in [19,21,22,23]. The general idea behind this approach is by taking advantage of information sharing existing among genes. The variability of genes can be utilized in the EB statistical model to cultivate gene information. The first step is to build a design matrix and the contrast matrix (supplied by the user). Design matrix is based on the experiment setup and the contrast matrix is the comparison of interests (samples/treatments/conditions). The second step is to fit a linear model curve that estimates the parameters using marginal distributions of the observed statistics. The final step is to reformulate the posterior odds statistic in terms of a moderated t-statistic where posterior standard deviations are used. The details of this method can be found in [8,20]. The three different gene lists using S3 implementing the EBayes method after preprocessing (median, LOWESS, and print-tip) from the normalization agents correspond to genes that show cytokine stress effect. We show the top 10 genes from each gene list in Table 3, Table 4, and Table 5. The first column is the gene's index. The M-value is the log2 fold change and the A-value is the average expression level for that gene across arrays. The $t$ value is the ratio of the M-value to its standard error. The P-value is obtained from the $t$ values. The B-value is the B-statistic which is the log-odds that a gene is differentially expressed. It is the ratio between the probability that a given gene is differentially expressed over the probability that a given gene is not differentially expressed. A B-statistic of zero means a 50-50 chance that a gene is differentially expressed [6].

### 5.3 Secondary Analysis with Analyze Agents

Given three gene lists generated using different normalization agents in the previous section, most often a researcher wants to compare the lists in order to make a better judgement about the gene list quality or draw out common genes that show significant expression among the lists. In most experiments, majority of genes are not differentially expressed and based on statistical threshold, so using the top 50 statistical significant genes from the gene lists, the analyze agent provides a correlation measure of the M-value and the A-value of the three lists of genes. In table 1 and table 2, we can see that the M-value and the A-value using LOWESS and Pin-Tip methods are quite closely correlated, suggesting that these two types of normalization can be used to preprocess the data without affecting downstream analysis too much. It is often wise to perform intermediate analysis and comparison before proceeding further because the choices made earlier can often affect subsequent result interpretation in gene expression analysis and our system let users do just that.

**Table 1.** Correlation measure of M-value using different normalization

| $Method$ | $Median$ | $LOWESS$ | $Pin - Tip$ |
|---|---|---|---|
| Median | 1.0 | 0.51 | 0.421 |
| LOWESS | | 1.0 | 0.896 |
| Pin-Tip | | | 1.0 |

**Table 2.** Correlation measure of A-value using different normalization

| $Method$ | $Median$ | $LOWESS$ | $Pin - Tip$ |
|---|---|---|---|
| Median | 1.0 | 0.44 | 0.301 |
| LOWESS | | 1.0 | 0.816 |
| Pin-Tip | | | 1.0 |

## 6 Multi-agent learning in gene expression data analysis

Multiagent Learning is at the intersection of Multiagent Systems and Machine Learning, both are the subfields of Artificial Intelligence. Machine Learning algorithms have been applied in different bioinformatics areas with encouraging outcomes and results. Traditional Machine Learning typically involves a single agent that is trying to maximize some utility function. Examples of traditional Machine Learning tasks include function approximation, classification problem, and parameter maximization problem given empirical data. With multi-agent system, multiple agents interacting together to perform a single task or different tasks toward a common goal, thus, extending the problem domain to include distributed learning in which many agents learn separately to achieve a common objective. This is especially true for multi-agent system where global behavior essentially emerges. Therefore, learning is a crucial part of autonomy in a multi-agent framework.

With gene expression data, Machine Learning approaches shine because the amount of gene data is large but little theory to interpret them. One of the examples of learning in gene expression is the work by Friedman [9] where a Bayesian network can be learnt using gene data. A Bayesian network is constructed using the conditional probability distribution of the network nodes. The network is composed of random variables which model the nodes in the network. In gene expression, the random variable nodes are the genes' expression level. In a multi-agent system setting, learning a Bayesian network can be done by each agent learning a subnetwork in the system and the results can be merged by the master agent of the system. One of the main goals of Bayesian network is to distinguish between causality and association and to explore and understand the elusive nature of gene regulatory network in living organisms.

## 7 Conclusions and Future work

To deal with massive amount of data produced by microarray experiment, we have proposed using a multi-agent system. This is because: (1) concurrent processing of gene expression analysis can be implemented by a multi-agent system, (2) the complexity of the system has been reduced to individual components making up the system by using modular design. One can register an agent in the system to fulfill the data analysis goal. Another advantage of using a multi-agent system to perform gene expression analysis is that it allows researchers to compare different outcomes produced by using different data path in the system pipeline, as we have shown in this paper. Although we have shown that the majority of gene expression analysis can be automated, the biggest question remains in selecting the right tools to perform the analysis. In other words, there is no unified method available for performing gene data analysis. For example, a common question before performing any data analysis task is to ask whether we need to preprocess the data, and if so how. Another relevant and challenging question is how to determine which statistical method to use to find significant genes given the massive amount of data. And most often, biologists or researchers only want to perform partial analysis and obtain the intermediate results before determining what other downstream analysis method to use. Although the current system is a work in progress, it has nonetheless provided a flexible architecture by offering users different choices to conduct gene expression analysis or to perform intermediate processing with the data. Adapting a multi-agent approach has shown that even though gene expression data processing is an inherently sequential task, it allows users to explore different data processing path concurrently and aids users in making the most appropriate decisions by providing intermediate results if requested.

A main extension to the system will be to accommodate a wide range of queries so that more specific data analysis can be done. Also, the need to implement agent with ability to provide informative information to help users to better understand the result during and after the analysis is also vital. We have also planned to switch to a graphical user interface where biologists are more accustomed to to provide easy navigation. As for implementing Machine Learning algorithms for multi-agent system, the difficulty lies in making modification to the existing Machine Learning algorithm designed for single agent learning. This is because most existing learning algorithms are used directly to a single agent within a multi-agent system [1]. At the present time, we are making sure our system's underlying functionalities are sound and effective and to allow users to provide valuable feedback. As for integrating Machine Learning algorithms into our system, we are currently working on building agents with learning capability and hopefully will be available for testing in the near future.

**Table 3.** 10 most distinctive genes via Median Normalization procedure

| $gene\#$ | $M-value$ | $A-value$ | $t-value$ | $P-value$ | $B-value$ |
|---|---|---|---|---|---|
| 4326 | -3.043893 | 10.735360 | -12.002565 | 8.660703e-16 | 33.643189 |
| 8601 | -1.961354 | 10.153340 | -10.882038 | 6.875208e-14 | 29.017720 |
| 11442 | -1.844612 | 9.968873 | -9.862251 | 5.016434e-12 | 24.695816 |
| 11373 | -2.204794 | 12.343943 | -9.550642 | 1.596294e-11 | 23.359504 |
| 13430 | -1.771025 | 10.321533 | -8.700099 | 6.661173e-10 | 19.690802 |
| 13433 | -2.058296 | 13.501244 | -8.547132 | 1.130084e-09 | 19.029552 |
| 12697 | -1.861232 | 11.305835 | -8.303853 | 2.996401e-09 | 17.978234 |
| 583 | -1.668582 | 9.708283 | -7.987291 | 1.135252e-08 | 16.612326 |
| 1598 | -1.436670 | 9.914980 | -7.824090 | 2.143097e-08 | 15.909770 |
| 12276 | -1.583127 | 10.638545 | -7.788921 | 2.268090e-08 | 15.758569 |

**Table 4.** 10 most distinctive genes via LOWESS Normalization

| $gene\#$ | $M-value$ | $A-value$ | $t-value$ | $P-value$ | $B-value$ |
|---|---|---|---|---|---|
| 4326 | -2.958899 | 10.735360 | -13.512274 | 1.782144e-21 | 46.50845 |
| 8601 | -1.911550 | 10.153340 | -11.921551 | 4.553869e-18 | 38.51401 |
| 11373 | -2.026666 | 12.343943 | -11.485672 | 3.237031e-17 | 36.28743 |
| 11442 | -1.820907 | 9.968873 | -10.727502 | 1.510529e-15 | 32.39173 |
| 13433 | -1.877986 | 13.501244 | -10.009076 | 6.068162e-14 | 28.68830 |
| 13430 | -1.724227 | 10.321533 | -9.960175 | 6.599055e-14 | 28.43627 |
| 12697 | -1.713848 | 11.305835 | -9.334747 | 1.686173e-12 | 25.21914 |
| 4963 | -1.725450 | 10.658531 | -8.860958 | 1.896474e-11 | 22.79617 |
| 13434 | -1.915059 | 10.211103 | -8.801144 | 2.323510e-11 | 22.49158 |
| 5512 | -1.569649 | 13.764983 | -8.675243 | 4.103380e-11 | 21.85159 |

**Table 5.** 10 most distinctive genes via Pin-tip Normalization

| $gene\#$ | $M-value$ | $A-value$ | $t-value$ | $P-value$ | $B-value$ |
|---|---|---|---|---|---|
| 4326 | -2.865264 | 10.735360 | -14.537376 | 2.263504e-22 | 48.89924 |
| 8601 | -1.943732 | 10.153340 | -12.806404 | 4.836380e-19 | 40.99271 |
| 13430 | -1.749680 | 10.321533 | -12.233605 | 5.480581e-18 | 38.29937 |
| 12697 | -1.752522 | 11.305835 | -11.931101 | 1.635944e-17 | 36.86371 |
| 11442 | -1.778762 | 9.968873 | -11.911700 | 1.635944e-17 | 36.77135 |
| 11373 | -1.984428 | 12.343943 | -11.725196 | 3.466326e-17 | 35.88173 |
| 12276 | -1.637334 | 10.638545 | -10.836023 | 2.620196e-15 | 31.60344 |
| 6186 | -1.582469 | 9.908442 | -10.447117 | 1.645690e-14 | 29.71719 |
| 4963 | -1.763556 | 10.658531 | -10.191223 | 5.363449e-14 | 28.47277 |
| 13433 | -1.674590 | 13.501244 | -10.003111 | 1.255348e-13 | 27.55683 |

## 8 Acknowledgement

This publication was supported by National Science Foundation (NSF 0534286).

## References

1. E Alonso, M Iverno, D Kudenko, M Luck, and J Noble, Learning in Multi-Agent System. Third Workshop of UK's Special Interest Group on Multi-Agent System, 2001.
2. Bioconductor Open Source Software for Bioinformatics, www.bioconductor.org
3. K Bryson, M Luck, M Joy and D T Jones, GeneWeaver: A Novel Genome Annotation System Based on Software Agents. Abstract/poster at the Seventh International Conference on Intelligent Systems for Molecular Biology, 1999.
4. Y Chen, E R Dougherty, and M L Bittner, Ratio-based decisions and the quantitative analysis of cDNA microarray images. J.Biomed. Opt. 2, 1997, pp. 364-367.
5. W S Cleveland: LOWESS, A program for smoothing scatterplots by robust locally weighted regression. The American Statistician, 1981.
6. C M T Dvorak, M Hardstedt, H Xie, M Wang, B J Hering, M P Murtaugh, and S Fahrenkrug, Transcriptional profiling of stress-response in cultured porcine islets. Manuscript, 2005.
7. B Efron, R Tishirani, J D Storey, and V Tusher, Empirical Bayes analysis of a microarray experiment. J. Amer. Statist. Assoc., 96, 2001 pp. 1151 - 1160.
8. Foundation of Intelligent Physical Agents (FIPA), www.fipa.org
9. N Friedman, M Linial, I Nachman, and D Pe'er, Using Bayesian Network to Analyze Expression Data. In Proceedings of the Forth Annual Conference on Research in Computational Molecular Biology RECOMB. 2000 pp. 127-135.
10. GeneData Expressionist, www.genedata.com/productoverview/expressionist/
11. GeneSpring, www.chem.agilent.com
12. T Ideker, V Thorsson, A F Siehel, and L E Hood, Testing for differentially expressed genes by maximum likelihood analysis of microarray data. J. Comput. Biol. 7, 2000, pp. 805-817.
13. L Jin, K V Steiner, C Schmidt, G Situ, S Kamboj, T Hlaing, M Conner, H Kim, M Emara, and K Decker, A Multiagent Framework to Integrate and Visualize Gene Expression Information. IEEE ICDM Workshop on MADW and MADM. 2005
14. P Khatri, and S Draghici, Ontological analysis of gene expression data: current tools, limitations, and open problems. Bioinformatics. vol. 21, no. 18 .2005.
15. M L Lee, Analysis of Microarray Gene Expression Data. Kluwer Academic Publishers. 2004.
16. M A Newton, C M Kendziorski, C S Richmond, F R Blattner, and K W Tsui, On differential variability of expression ratios: improving statistical inference about gene expression changes from microarray data. J.Comput.Biol, 2001, pp. 37-52.
17. J Quackenbush, Microarray data normalization and transformation. Nature genetics supplement vol 32, 2002
18. G K Smyth, Y H Yang, and T Speed, Statistical Issues in cDNA Microarray Data Analysis. Functional Genomics: Methods and Protocols, 2002.
19. G Smyth, N Thorne, and J Wettenhall, LIMMA package user guide.
http://www.maths.lth.se/help/R/.R/
library/limma/doc/usersguide.html, 2004.
20. G K Smyth, Linear models and empirical Bayes methods for assessing differential expression in microarray experiments. Statistical Applications in Genetics and Molecular Biology 3, No. 1, Article 3, 2004.
21. G K Smyth, J Michaud, and H Scott, The use of within-array replicate spots for assessing differential expression in microarray experiments. Bioinformatics Vol. 21, 2005, pp. 2067-2075.
22. G K Smyth, and T P Speed, Normalization of cDNA microarray data. Methods 31, 2003, pp. 265-273.
23. G K Smyth, Limma linear models for microarray data. In: Bioinformatics and Computational Biology Solutions using R and Bioconductor, R. Gentleman, V. Carey, S. Dudoit, R. Irizarry, W. Huber (eds.), Springer, New York, 2005, pp. 397-420.
24. J G Thomas, J M Olson, S J Tapscott, and L P Zhao, An efficient and robust statistical modeling approach to discover differentially expressed genes using genomic expression profiles. Genome Res., 11, 2001, pp. 1227-1236.
25. V G Tusher, R Tibshirani, and G Chu, Significance analysis of microarrays applied to the ionizing radiation response. Proc. Natl Acad. Sci. USA, 98, 2001, pp. 5116-5121.
26. Y H Yang, et al, Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. Nucleic Acids Research, Vol 30, No. 4, 2002.

## 9 Author Bios

**H.C. Lam** is a graduate student at the University of Minnesota, Minneapolis. USA.

**M. Vazquez Garcia** is a graduate student at the Departamento de Sistemas Informaticos y Programacion, Universidad Complutense de Madrid, Spain.

**B. Juneja** is a graduate student at the Department of Animal Science, University of Minnesota, St.Paul. USA.

**S. Fahrenkrug** is Associate Professor of Genetics at the Department of Animal Science, University of Minnesota, St.Paul. USA.

**D. Boley** is Professor of Computer Science and Engineering at the Department of Computer Science and Engineering, University of Minnesota, Minneapolis. USA.