CHAPTER 8
DISCUSSION, CONCLUSION AND FUTURE WORK

The autonomic computing has been inspired by the human Autonomic Nervous System. The similarity of the human autonomic nervous system of the body with the autonomic computer system gave birth to the term -Autonomic Computing (AC). One of the major characteristics of AC is self optimization.

'Supervised learning', 'Knowledge Discovery' 'Association rule mining', 'Classification' , 'data sets' are some of the popular and common terms in data mining. An optimization characteristic of AC overlaps with the concepts in data mining.

In this thesis four different scenarios were set, trying to understand how self optimization of the system can be achieved and improved, taking a step further towards the huge autonomic computing area.

I. The first one being SQL Query Dissembler, where, an approach to managing the execution of large complex queries in a database and identifying its impact on other smaller, possibly more important, queries is presented. The experiment shows that concurrent execution of large queries can have significant impact on the performance of other workloads (small queries), especially, as the points of contention between the workloads increase. This leads to the conclusion that there is a need to be able to manage the execution of these large queries in order to control their impact on small queries.

Currently, due to the fact that this approach is implemented outside database engine, there was no choice, but to use an expensive way to store the intermediate results. For intermediate results from inside a database engine, one could probably design a cheaper and faster mechanism to save the intermediate results. A possible solution would be to save the ROWID and COLUMNID information of a table instead of storing its real record values.

Another big improvement of saving the intermediate results from inside a
database engine is that it would avoid the overhead that is caused by the DBMS following the standard parsing, compiling, and optimizing procedure to execute a submitted SQL statement[73].

Future enhancement for this work is listed below:

1. Currently SQL query dissembler works for the small query set in the experiment workload which contains just read-only queries. It is desirable to consider update-queries (e.g. INSERT, UPDATE, and DELETE) in the workload. These queries tend to create more resource contentions on a database system and may cause data inconsistency problems.

2. To investigate a better way to execute the decomposed segments, preferably within a database engine so that internal query models, such as Query Graph Model, can be directly utilized.

3. With respect to SQL query dissembler, comparative study of autonomic components of some selected DBMSs to observe their autonomic maturity level need to be done. HI (Human Intervention) can be used to measure the autonomicity of the DBMS. Lesser the intervention more is the autonomicity and vice versa. There is a need to provide guidelines and benchmarks to evaluate the autonomicity degree in DBMSs.

4. Further research work also includes learning during query access plan selection; fully automated configuration and tuning of all system resources allocated to the database (memory, network, disk, CPU).

II. Second, is the online indexing for databases using query workloads. Considering the fact that there are a lot of loopholes associated with static indexing and sequential search, a new method called online indexing (dynamically indexes are assigned for products present in frequent item set) was proposed to remove the loopholes persisting in the current environment. The major problem associated with huge databases is indexing and retrieving the frequently occurring products as quickly as possible to reduce the searching time and to increase the performance level which can done with the help of online indexing and other parameters. For the concept
Study and analysis of autonomic components to improve the performance of some database applications

to be realized, parameter considered was minimum support. Apriori association rule mining algorithm was applied to establish relationships among the products in the frequent item set. The number of transactions considered for experiment was 500. Minimum support values were varied to identify the frequent itemsets in the dataset. The concept could be applied to high dimensional databases in real world applications. It was found that as the minimum support value increases the number of frequent itemsets grouped decreases and vice versa.

Further enhancement possible here could be:
1. Execution of the same problem statement using FP-growth algorithm. FP-Growth algorithm is supposed to give comparatively faster and efficient results.
2. The experiment could be carried out with the increased number of transaction. Also, the chosen minimum support value could be varied in the smaller range for more precise interpretation.

III. In OARM, frequent itemsets were generated using the apriori association rule mining algorithm. The genetic algorithm has been applied on the generated frequent itemsets to generate the rules containing positive attributes, the negation of the attributes with the consequent part consisting of single attribute or more than one attribute. The minimum support value chosen for experimentation was 20%. The dataset of the size 1000 and 5000 were used as input values. It was found that though there was a major change in the size of datasets used, the number of frequent itemset groups formed changed marginally. However, the density of the receipts on each group of these datasets varied drastically.

Future enhancements possible are:
1. Extending the association rule mining algorithm in this system to minimize the complexity of the genetic algorithm and scanning of database by applying Bayes theorem on the generated rule.
2. To have extensive analysis process as a part of the system in future to improve and measure the autonomicity of the system.

IV. Fourth, is the decision tree classifier with GA based feature selection. Here, the implementation and testing the performance of Decision Tree-based classifiers with and without GA was successfully carried out. The DT classifier was optimized using a Genetic Algorithm to select a subset of the features that were to be used in constructing an optimal decision tree. The efficiency of the decision tree constructed is solely based on the input training and testing samples. Time taken for classification in case of GA based decision tree is significantly better than the traditional one.

Future enhancements possible are:

1. Decision tree classifiers application could be made more responsive by using threads and parallel cloud computing.
2. An interesting extension to be explored is the possibility of additional feedback from ID3 concerning the evaluation of a feature set.
3. Currently the GA based feature selection is considering the features entirely by the information through GA operations, this can be changed by allowing the user to specify certain features which might be a dominating factor for the sample classification from user perspective.
4. Using the GA based selection for the remaining subset of features, comparing this with entirely selecting features based on GA is one more aspect to look into.