BIOINFO Genetic Programming
Volume 1, Issue 1, 2011, pp-01-04
Available online at: http://www.bioinfo.in/contents.php?id=280

Real life problem Solved using Genetic Algorithm and Data Warehouse

H.S. Fadewar[1] and G.N. Shinde[2]

[1] Sinhgad Insitute of Management and Computer Application, Narhe, Pune.
fadewar_hsf@yahoo.com
[2]Principal, Indira Gandhi Mahavidyalaya, CIDCO, Nanded

Abstract :

In this paper, we explore the use of an genetic algorithm for materialized view selection based on multiple global processing plans for queries so as to achieve both good query performance and low view maintenance cost.

Keywords : Data warehouse , Materialized view , Genetic Algorithm , Query , MVPP etc.

1. INTRODUCTION

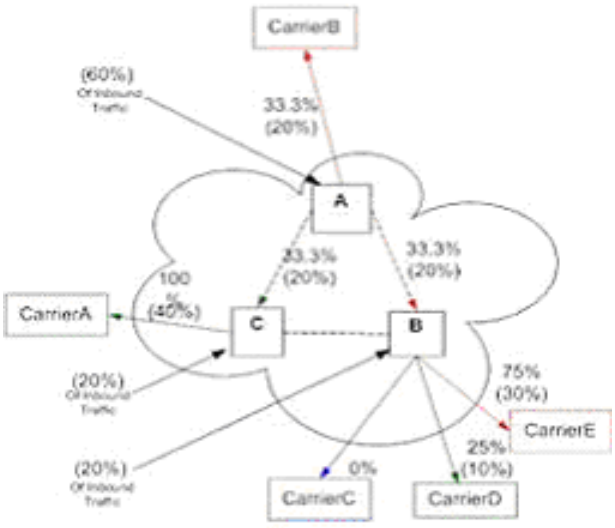
DATA warehousing is an approach to the integration of data from multiple, possibly very large, distributed, heterogeneous databases and other information sources. A data warehouse (DW) is a repository of integrated information available for querying and analysis. To avoid accessing the original data sources and increase the efficiency of the queries posed to a DW, some intermediate results in the query processing are stored in the DW. These intermediate results stored in a DW are called materialized views. On a sufficiently abstract level, a DW can be seen as a set of materialized views over the data extracted from the distributed heterogeneous databases. There are many research issues related to DWs [1], among which materialized view selection is one of the most challenging ones. On one hand, materialized views speed up query processing. On the other hand, they have to be refreshed when changes occur to the data sources. Therefore, there are two costs involved in materialized view selection: the query processing cost and the materialized view maintenance cost. The question we are interested in is what views should be materialized in order to make the sum of the query performance and view maintenance cost minimal. The materialized view selection involves a difficult trade-off between query performance and maintenance cost.

- Materializing all the views in a DW can achieve the best performance but at the highest cost of view maintenance.
- Leaving all the views virtual will have the lowest view maintenance cost but the poorest query performance. The word "virtual" here means that no intermediate result will be saved in the DW.
- We can have some views materialized (e.g., have those shared views materialized), and leave others virtual. In this way we may achieve an optimal (or near optimal) balance between the performance gain and maintenance cost.

1.1. Real world Uses Genetic Algorithm:

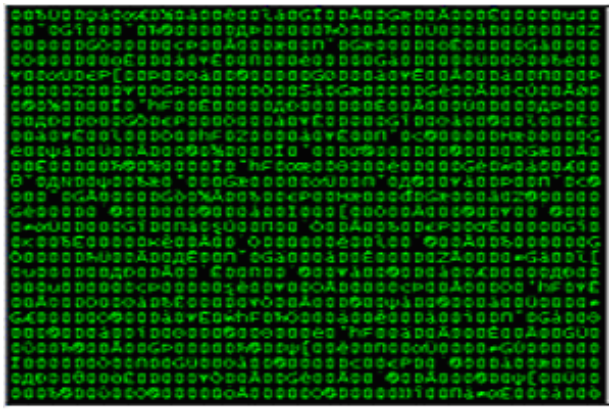
Optimized Telecommunications Routing:

Do you find yourself frustrated by slow LAN performance, inconsistent internet access, a FAX machine that only sends faxes sometimes, your land line's number of 'ghost' phone calls every month? Well, GAs are being developed that will allow for dynamic and anticipatory routing of circuits for telecommunications networks. These could take notice of your system's instability and anticipate your re-routing needs. Using more than one GA circuit-search at a time, soon your interpersonal communications



problems may really be all in your head rather than in your telecommunications system. Other GAs are being developed to optimize placement and routing of cell towers for best coverage and ease of switching, so your cell phone and blackberry will be thankful for GAs too[2].

Encryption and Code Breaking:



On the security front, GAs can be used both to create encryption for sensitive data as well as to break those codes. Encrypting data, protecting copyrights and breaking competitors' codes have been important in the computer world ever since there have been computers, so the competition is intense. Every time someone adds more complexity to their encryption algorithms, someone else comes up with a GA that can break the code. It is hoped that one day soon we will have quantum computers that will be able to generate completely indecipherable codes. Of course, by then the 'other guys' will have quantum computers too, so it's a sure bet the spy vs. spy games will go on indefinitely[3].

2. MATERIALIZED VIEW SELECTION

Materialized view selection consists of three optimization problems, i.e., query optimization, multiple query optimization, and materialized view selection. It should be pointed out that a set of locally optimized queries may not be optimal anymore if multiple queries are considered together. Similarly, an optimal set of multiple queries does not guarantee the optimal selection of materialized views because a different set may lead to better materialized views. It is important to consider all three problems together in materialized view selection.

2.1 Query and Multiple Query Optimization:

A lot of research has been done on this topic query optimization. A DW is a repository of integrated information available for querying analysis. One issue we have to deal with is multiple query processing.

2.2 Materialized View Selection

In DW, selected information is extracted in advance and stored in a repository. A DW can therefore be seen as a set of materialized views defined over the sources. The problem we are dealing with now is how to select the views to be materialized so that the cost of query processing and view maintenance for all the nodes in a global processing plan is minimized. An easy approach would be to use exhaustive search to find the optimal set of materialized views on the set of queries. However, this approach is impractical if the search space is big. Heuristic algorithms have to be used to trim the search space in order to get the results quickly [5], [6]. However, the performance of a heuristic algorithm depends heavily on the quality of heuristics which may be difficult and /or costly to obtain in practice. Heuristic algorithms also get stuck easily in a local optimum.

Compared with heuristic algorithms, Genetic algorithms have many advantages, such as searching from a population of points using probabilistic transition rules. In order to avoid an exhaustive search in the whole solution space and obtain a better solution than that obtained by heuristic methods, we propose a new Genetic approach to materialized view selection.

3. GENETIC ALGORITHMS FOR MATERIALIZED VIEW SELECTION

Genetic algorithms is solve many real world problems with success [19]. They use population based stochastic search strategies and are unlikely to be trapped in a poor local optimum. They make few assumptions about a problem domain yet are capable of incorporating domain knowledge in the design of chromosome representation and variation operators. They are particularly suited for large and complex problems where little prior knowledge is available.

The genetic algorithm apply on the above motivating example taken from[1,4] and selects the best

set of materialized views with the minimal total cost for a particular global processing plan. The results

Details of our implementation are described in the following subsections.

Abstract framework for genetic algorithm

```
BEGIN
Generate the initial population, G(0);
Evalulate all indivisuals in G(0);
t:=0;
REPEAT
    t=t+1;
    Select G(t) from G(t-1);
    After  G(t) using variation operat;
    Evaluate all indivisuals in G(t);
    UNTIL a satisfactory solution is found;
    End;
```

Mapping from a DAG (i.e., genotype) to a binary string (i.e., phenotype).

1. Input a global processing plan represented by DAG.
2. Use a certain graph traversal strategy, such as breadth first, depth first or other problem specific strategies, to traverse through all nodes in the DAG and produce an order list of nodes.
3. Create a binary string according to this order ,where 0 indicates that the corresponding node is not materialized and 1 represent that the corresponding node is materialized. The binary string is also called the mapping array[7].

3.1 Representation of Solutions

Representation is one of the key issues in problem solving. Good representations often lead to a more efficient algorithm for solving a problem. Different problems usually require different representations. The representation of materialized views for optimization is based on DAGs. Each DAG is encoded as a binary string[8].

3.2 Fitness Functions in our Genetic Algorithms

Since the objective in our cost model is to minimize the sum of query and maintenance cost while the fitness function of Genetic algorithm is usually defined as maximization, we have applied the following simple transformation to define the fitness function from the cost

$$f(x)= \begin{cases} C_{max} - c(x), & \text{when } c(x) < C_{max} \\ 0, & \text{otherwise} \end{cases}$$

Where c(x) denote the cost function and f(x) denote the fittest function . There are a lot of ways of choosing the coefficient C_{max}. It can be set to the largest value C(x) in the current population or the largest in the last k generations. Each individual in a population represents a set of materialized views. Its fitness depends on the total query and maintenance cost .

3.3 Crossover :

Crossover encourages information exchange among different individuals. It helps the propagation of useful genes in the population and assembling better individuals. One-point crossover is used in our Genetic algorithms for its simplicity and effectiveness in our case. In a Genetic algorithm, the crossover is implemented as a kind of cut-and-swap operator [09]. For example,given two individuals

L_1 = 1100100 | 0100100001111

and

L_2 = 0100110 | 1011000100111

Where L_1 indicates that nodes {Q5, Q4, Q1, result4, tmp3, tmp1, tmp2, tmp5, and tmp6} are materialized and L_2 means that nodes {Q4, Q1, result5, result2, result3, tmp9, tmp7, tmp2, tmp5, and tmp6} are materialized. Assume the crossover point (indicated by symbol |) is chosen at random as seven, between one and 20. Then the two offspring after crossover are

L'_1 = 1100100 | 1011000100111 and

L'_2 = 0100110 | 0100100001111

where L'_1 indicates that nodes {Q5, Q4, Q1, result2, result3, tmp9, tmp7, tmp2, tmp5, and tmp6} are materialized and L'_2 shows that nodes {Q4, Q1, result5, result4, tmp3, tmp1, tmp2, tmp5, and tmp6 }are materialized. Two new sets of materialized views are generated which have inherited genes from both parents.

3.4 Mutation :

Although crossover can put good genes together to generate better offspring. It cannot generate new genes. Mutation is needed to create new genes that may not be

present in any member of a population and enables the algorithm to reach all possible solutions (in theory) in the search space. Mutation in Genetic algorithm is implemented as a bit-flipping operator. Given an individual[10]

L = 11001000100100001111

A random position between onr and 20 will be generated first. Say it is 16. Then the 16th bit will be flipped from 0 to 1 with a probability to produce the offspring

L' = 11001000100100011111

4. CONCLUSION

In this paper the study has mainly focused on how to select of views to be materialized so that the sum of processing a set of queries and maintaining the materialized views is minimized. A genetic algorithm is proposed to solve view selection problem.

5. REFRENCES

[01] B. Kristin, M. C. Ferris, and Y. Ioannidis, "A genetic algorithm for database query optimization," Univ. Wisconsin, Madison, Tech. Rep.TR1004, 1991.

[02] M. Gregory, "Genetic algorithm optimization of distributed database queries," in Proc. ICEC, 1998, pp. 271–276.

[03] Y. E. Ioannidis, "Query optimization," ACM Comput. Surv., vol. 28, no. 1, pp. 121–123, Mar. 1996.

[04] S. Chaudhuri, "An overview of query optimization in relational systems,"in Proc. 17th ACMSIGACT-SIGMOD-SIGART Symp. Principles Database Syst. (PODS), June 1998, pp. 34–43.

[05] C. Wang and M.-S. Chen, "On the complexity of distributed query optimization," IEEE Trans. Knowl. Data Eng., vol. 8, pp. 650–662, Aug.1996.

[06] A. Ho and G. Lumpkin, "The genetic query optimizer," in Genetic Algorithmsat Stanford 1994, J. R. Koza, Ed. Stanford, CA: Stanford Univ.,1994, pp. 67–76.

[07] C. Zhang and J. Yang, "Genetic algorithm for materialized view selection in data warehouse environments," in Proc. First Int. Conf. Data Warehousing Knowledge Discovery, Lecture Notes in Computer Science, Florence, Italy, 1999.

[08] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA: Addison-Wesley, 1989.

[09] T. Back, D. B. Fogel, and Z. Michalewicz, Handbook of Evolutionary Computation. Amsterdam, The Netherlands: IOP/Oxford Univ. Press,1997.

[10] R. E. Smith, D. E. Goldberg, and J. A. Earickson, "SGA-C: A C-language implementation of simple genetic algorithm," TCGA, Clearing House for Genetic Algorithms, Univ. Alabama, Dept. Eng. Mech.,Tuscaloosa, Rep. 91 002, Mar. 1994.