

# A Genetic Algorithmic Approach for Determining Optimal Secondary Index Set for Querying on Multiple Relations

N Raja Kumar Reddy<sup>1</sup> and M M Naidu<sup>2</sup>

<sup>1</sup> Associate Professor, Dept. of Computer Science and Engineering,  
Sri Kalahasteswara Institute of Technology, Srikalahasti – 517640, A.P., India  
E-mail: [rajakumarreddyn@rediffmail.com](mailto:rajakumarreddyn@rediffmail.com)

<sup>2</sup> Professor of Computer Science and Engineering,  
S V University College of Engineering, S V University, Tirupati, A.P., India  
E-mail: [mmnaidu@yahoo.com](mailto:mmnaidu@yahoo.com)

## Abstract

The primary job of a database administrator, as a part of physical database design and tuning, is to determine an optimal secondary index set for a given workload of a multi-relation database application. As the solution space of candidate secondary index sets is prohibitively large, it boils down to NP-Hard problem. Owing to the nature of the problem, Genetic Algorithmic Approach (GAA) is proposed for determining the optimal secondary index set. The model is validated and sensitivity analysis is performed.

**Keyword** Database Management System, Genetic Algorithm, Fitness Function, Secondary Indexes, Sensitivity Analysis.

## 1 Introduction

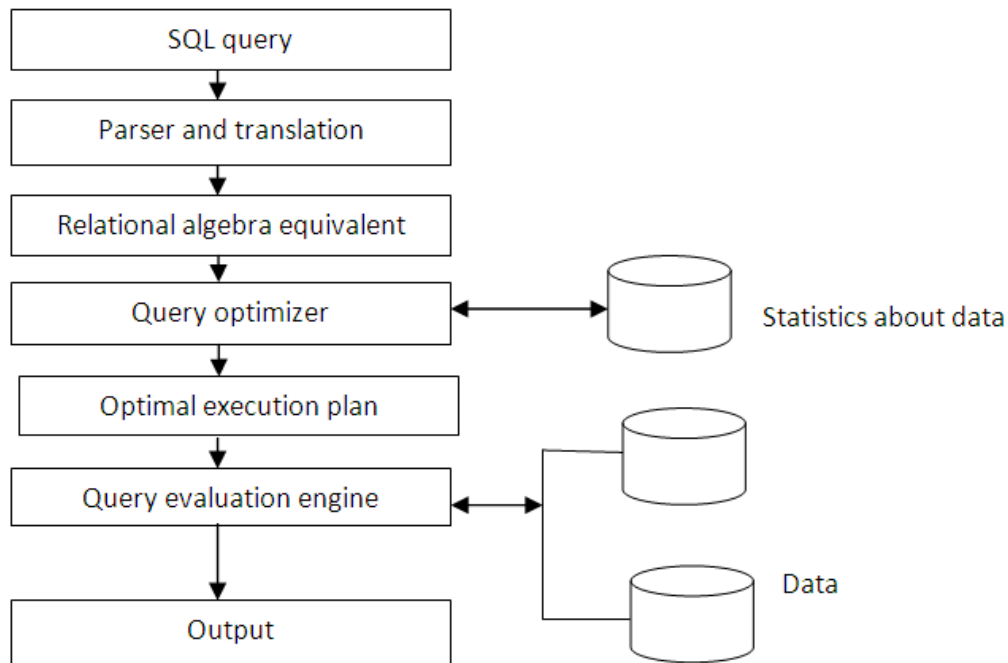
Enterprise-class databases need database administrators for performance tuning. An important area in which tuning is needed is in determining physical database design for the workload [1] and particularly in the selection of indexes to build for the database [2] to minimize the query process time for the workload. Database tuning needs a complete knowledge about the system internals, data characteristics, applications, workload, query processing strategies etc [3].

The basic steps [4], [5], [6] involved in answering the query submitted to the SQL Server are described as shown in Figure 1. The processes involved in answering the query can be broken down into the processes occurred in the relational engine and

the processes occurred in the storage engine [6].

In the relational engine, the query is parsed to check the syntax and outputs the parse tree or query tree or sequence tree which denotes the logical steps needed to execute the query. The parse tree is then passed to a process called the algebraizer to resolve the names of objects, tables, and columns, referred to within the query. The algebraizer outputs the query processor tree by including a hash, a coded value to represent the query and then it passed on to the query optimizer to generate an execution plan. Using the query processor tree and the statistics about the data, the optimizer estimates the cost involved in answering the query and generates an execution plan which consumes minimal resources. The storage engine or query evaluation engine uses that execution plan as a basis to answer the query [6].

While estimating the cost involved in answering the query, the query optimizer uses the existing index set to consume minimal resources. Hence there is a need to update the existing index set with the useful index set for answering the given planned queries in the workload to consume minimal resources.



**Figure 1: Basic steps involved in the query process**

Generally, there are two kinds of indexes called primary and secondary indexes [7]. The tuples in the relation are ordered based on the primary index but not on the secondary indexes. There are at most one primary index and multiple secondary indexes for a relation.

The selection of optimal secondary index set for multiple relations to answer the planned queries in the workload needs to evaluate  $2^D$  candidate secondary index

sets where  $D$  is the sum of non-key attributes of all the relations. The solution space is discrete and size increases steeply as  $D$  exceeds 10. Hence the problem of determining an optimal secondary index set for multiple relations to answer the planned queries in the workload is considered.

Genetic Algorithms are inspired by Darwin's theory and designed to search large spaces for optimal function values [8], [9]. Due to steep increase in size of the candidate secondary index sets when  $D$  exceeds 10, Genetic Algorithmic approach is proposed to determine an optimal secondary index set for multiple relations to answer the planned queries in the given workload.

The rest of the paper is described as follows. In Section 2, brief literature review is presented. Section 3 describes the problem. In Section 4, Mathematical model is formulated. Section 5 outlines the proposed work. Section 6 presents the evaluation and validation of the work. Section 7 presents the sensitivity analysis and Section 8 concludes the work.

## **2 Literature Review**

A brief literature is presented on the contributions of many authors on the selection of secondary indexes to improve the overall system performance.

In the physical database design, selection of secondary indexes is crucial and is a considerable interesting problem [10], [11]. Secondary indexes are used to facilitate faster data access with bypassing exhaustive searches for the needed data [10]. The problem of selecting an optimal set of secondary indexes has been studied by many contributors and authors since 1970 and it is a well recognized problem [2], [12], [13].

Despite a long history of the work, the selection of optimal index set has not been addressed due to large number of candidate index sets and lack of a good search technique to select the best index set [2].

For selecting an optimal secondary index set, the internal structure of the SQL Server, workload particulars, usage of data, uses of the indexes, statistics about the data used by query optimizer are essential factors. It is difficult to obtain the same due to frequent changes in the requirements, size of the data and changes in the workload etc [3].

The index selection problem is defined as selecting an optimal set of indexes for a relation having  $d$  non-key attributes. In order to select an optimal set of indexes, there is a need to examine  $2^d$  candidate index sets [14], [15]. Being a search problem, the number of candidate index sets is large and hence it is difficult search problem [1], [2].

Schkolnick [16] proposed a solution to select the best set of indexes in an exponential time in the number of attributes. Barucci E et al [17] presented a near optimal solution to the index selection problem in a polynomial time. Comer D [15] illustrated the difficulty in selecting an optimal set of indexes and demonstrated as an NP-Complete problem. Due to difficulty in determining the optimal set of indexes, many contributors and authors provided an approximate solution at a reasonable amount of time and cost. It is motivated to study the approximate solution to the index

selection problem and analyse the same.

The Index selection problem is further defined as selecting an optimal or near optimal set of indexes consisting of at most one primary index and multiple secondary indexes and hence index selection problem is classified into two sub-problems – selection of primary index and selection of secondary indexes [11].

Heuristic techniques are commonly used to select primary index due to various reasons [11]. The primary index needs the physical ordering of the data on it and hence at most only one is possible for it. More attention has been paid to the selection of secondary indexes [10], [11], [17]. The selection of secondary indexes is illustrated as difficult search problem and formulated as an optimization problem [18]. The selection of secondary index set is illustrated as NP-Complete problem [19].

Heuristic and exact solutions to determine the set of indexes are identified in the literature [20]. The Heuristic solutions [17] to the problem provided an approximate solution but no guaranteed to process the given workload with minimal transaction cost. The exact solutions [21] to the problem of selection of indexes need to examine the exhaustive enumerated candidate solutions and hence they are time consuming and are not possible in reasonable time and cost. Genetic Algorithms are used to solve NP-Complete problems [22].

Genetic Algorithms are search techniques based on the mechanics of natural selection and natural genetics [8]. Genetic Algorithms perform multidimensional search from a population of random solutions and applied elitism, crossover and mutation operations on the earlier population of solutions to identify and optimal solution for optimization problems. Unlike conventional techniques, the probability to arrive a false peak solutions can be reduced by creating a well set of population of candidate solutions.

In Genetic Algorithm, there is no specific method to terminate the Genetic Algorithm either on completion of fixed number of generations or on obtaining the solution with no improvement further. Bhandari et. al [23] proposed a stopping criterion based on the variance of the fitness function values of the best chromosomes obtained over the generations is considered in the proposed Genetic Algorithm to get better results.

An index tuning wizard in SQL Server [24] begins with single attribute indexes and then working on large number attributes indexes as the time permits. The objective of the index tuning wizard is to reduce the number of invocations while getting solution. It may not possible to select set of indexes when there is no enough data in the relations being sampled [25].

As database gets larger and larger, an optimal solution to determine the best secondary index set is yet to be determined and agreed upon. The researchers are still have been attempting to find the optimal or near optimal solution to the selection of secondary index set so as improve the overall performance of the system.

Raja Kumar Reddy and Naidu [26] proposed a Genetic Algorithm to determine an optimal secondary index set for a given workload on a single relation. This paper proposes a genetic algorithmic approach for determining an optimal secondary index sets for a given workload on multiple relations.

### 3 Problem Description

A database application uses the services of a database server for storing, processing and retrieving of data employing multiple relations referred to as database. It facilitates answering ad hoc and planned queries to meet the functional requirements of the users. The database administrator undertakes physical database design and tuning to meet the non-functional requirements of the application. The primary job of a database administrator is to select indexes for multiple relations that minimize effort for answering queries. In this study, a set of planned queries associated with their probability of occurrences is referred to as workload. To meet the performance requirements of an application, a secondary index set for multiple relations is to be determined that minimizes the query answering effort. The secondary index set that minimizes the query answering effort for a given workload of an application is called as optimal secondary index set. The problem of determining optimal secondary index set is considered in this paper. Since the solution space for the problem is prohibitively large for searching an optimal solution, it boils down to NP-Hard problem. Hence, a Genetic Algorithmic Approach which employs randomized search is proposed.

### 4 Mathematical Model

A mathematical model is formulated with the following assumptions that represent the present problem:

- 1) The workload means a set of planned queries associated with their probability of occurrences.
- 2) It is required to access multiple relations for answering each query of the workload.
- 3) Each relation has a primary key and multiple non-key attributes.
- 4) A secondary index is created on a single non-key attribute of a relation.

The notation is defined in Table 1 given below:

|           |  |
|-----------|--|
| L         | Workload, a set of planned queries associated with their probability of occurrences      |
| n         | Number of planned queries in L   |
| $q_i$     | $i^{\text{th}}$ Query in L   |
| $p_i$     | Probability of occurrence of $q_i$ where $\sum_{i=1}^n p_i = 1$                          |
| r         | Number of relations  |
| $T_i$     | Relation i   |
| $d_i$     | Number of non-key attributes in $T_i$  |
| D         | $\sum_{i=1}^r d_i$ , Sum of the non-key attributes of all the relations                  |
| $m_{i,j}$ | Number of attributes not indexed in $T_j$ which should have been indexed for $q_i \in L$ |

|               |  |
|---------------|--|
| $M_i$         | $\sum_{j=1}^r m_{i,j}$   |
| $e_{i,j}$     | Number of attributes indexed in $T_j$ which should have not been indexed for $q_i \in L$ |
| $E_i$         | $\sum_{j=1}^r e_{i,j}$   |
| $\delta$      | Imputed weight for querying on an attribute not indexed which should have been indexed   |
| $1 - \delta$  | Imputed weight for querying on an attribute indexed which should have not been indexed   |
| $P$           | Population, set of chromosomes with length $D$   |
| $P_s$         | Population size, number of chromosomes in the population                                 |
| $E_r$         | Percentage of chromosomes in the current $P$ used for Elitism                            |
| $P_c$         | Crossover probability  |
| $P_m$         | Mutation probability   |
| $v_i$         | Fitness function value for $q_i \in L$   |
| $Z$           | Expected fitness value   |
| $C_g$         | Generation number counter  |
| $C_s$         | Sample number counter  |
| $Z_i$         | Minimum expected fitness value of the $i^{\text{th}}$ generation                         |
| $\bar{Z}_n$   | Average of $Z_i$ 's up to the $n^{\text{th}}$ generation                                 |
| $\bar{Z}_n^2$ | Average of $Z_i^2$ 's up to the $n^{\text{th}}$ generation                               |
| $\sigma_n^2$  | Variance of $Z_i$ 's up to the $n^{\text{th}}$ generation                                |

Table 1: Notation used in the Model

$$\text{Minimize } Z = \sum_{i=1}^n v_i p_i \quad (1)$$

Where

$$v_i = \delta M_i + (1 - \delta) E_i \quad (2)$$

## 5 Proposed Genetic Algorithmic Approach(GAA)

The Genetic Algorithms are evolutionary algorithms (EA) that are used to solve the optimization problems using the techniques inspired by natural evolution such as elitism, selection of parents for crossover and mutation [8],[9]. The steps in the Genetic Algorithm are:

- 1) Generate the initial population randomly
- 2) Compute fitness value, using fitness function, for each chromosome in the initial population
- 3) Select the best chromosome from the initial population based on their fitness value

- 4) Repeat until stopping criterion is met
  - 4.1) Generate next population by applying elitism, crossover and mutation
  - 4.2) Compute fitness value, using fitness function, for each chromosome in the present population
  - 4.3) Select the best chromosome from the present population

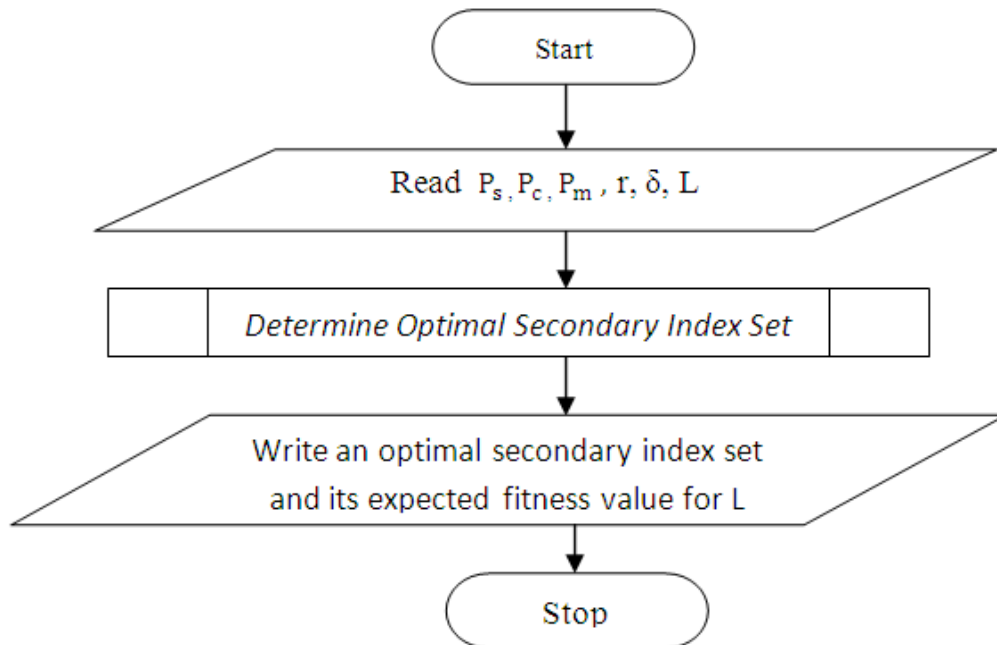
The members in the population, called chromosomes, are represented as bit strings. While generating the next population, elitism, crossover and mutation operators are applied on the chromosomes in the current population. The best fraction of the current population, called elitism rate, is copied to the next population to ensure the survival of the fittest. Crossover is used to combine the genes of the two parents using crossover probability to yield two offspring. Mutation is applied to change a fraction of genes of offspring referred to as mutation probability [26].

A stopping criterion for the proposed Genetic Algorithm is defined, by applying Bhandari et. al proposed a stopping criterion [23] for Genetic Algorithm, based on the variance of the expected fitness values for workload with best chromosomes obtained over the generations and is calculated using equation (3).

$$\sigma_n^2 = \frac{1}{n} \sum_{i=1}^n (Z_i - \bar{Z}_n)^2 = \bar{Z}_n^2 - \bar{Z}_n^2 \quad (3)$$

The proposed GAA, to determine an optimal set of secondary indexes for multiple relations for the given workload, L, and  $\delta$ , is described as shown in the Figure 2. The predefined method called *Determine optimal secondary index set* returns an optimal set of secondary indexes for multiple relations along with its associated expected fitness value, Z, for L shown in Figure 3. The predefined methods used in Figure 3 are described below:

- 1) *Generate initial set of chromosomes, initial population, randomly*: This predefined method creates  $P_s$  chromosomes, bit strings, of length D, called initial population. Each bit in the bit string is discrete uniform variant [0, 1].
- 2) *Find the best chromosome with minimum Z for L*: This predefined method computes the expected fitness value, Z, for L, using equation (1), for each chromosome in the population and returns the best chromosome with minimum expected fitness value for the workload, L.
- 3) *Calculate  $\sigma_n^2$* : This predefined method calculates the variance of the expected fitness values for L with best chromosomes obtained up to  $n^{\text{th}}$  generation by applying equation (3).
- 4) *Generate the set of chromosomes for next population*: This method creates next population with  $P_s$  chromosomes by applying elitism, crossover and mutation.



**Figure 2: The proposed GAA to determine an optimal set of secondary indexes for the given  $L$ , and  $\delta$**

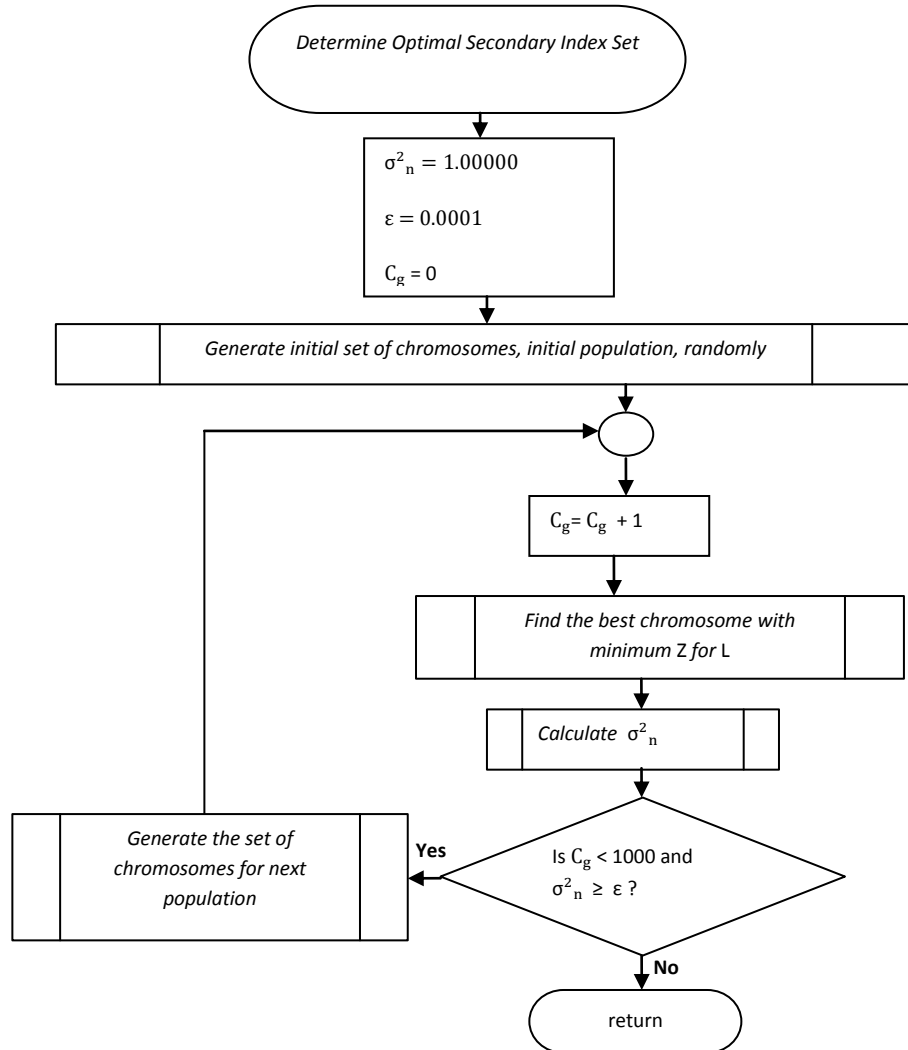
### 5.1 Input and Output of the Proposed Model

**Input:**  $P_s, P_c, P_m, r, \delta, L$

**Output:** Attribute(s) of the relations in the database to create secondary indexes.

## 6 Model Validation

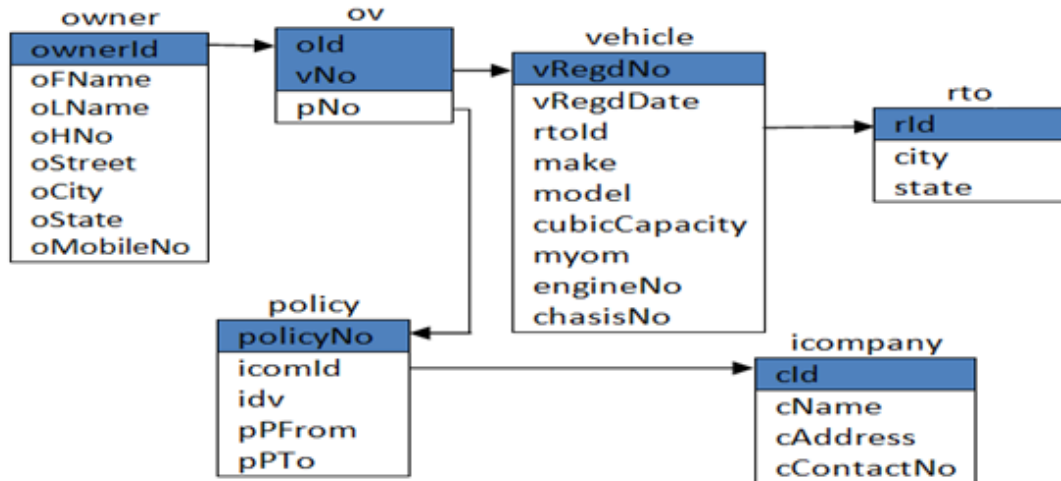
The proposed GAA is implemented on the platform.NET in C# for SQL Server 2000 as shown in Figure 2. The solution space of candidate secondary index sets is discrete and increases steeply for  $D > 10$ . For model validation, a schema diagram, shown in the Figure 4, is designed for Road Transport Department Application which maintains the data related to registration authorities, vehicle registration and insurance policies. The entity sets, vehicle owner, vehicle, registration authority, insurance company and insurance policy, yield five relations and a ternary relationship among vehicle owner, vehicle and insurance policy yield another relation.



**Figure 3: Predefined method - Determine Optimal Secondary Index Set**

The workload used to validate the proposed model is given in Table 2. The binary representation of an instance of planned queries in L is described in Table 3. The GAA for determining the optimal secondary index set on six relations is implemented employing the input parameters,  $P_s = 90$ ,  $P_c = 0.80$ ,  $P_m = 0.05$ ,  $r = 6$  and  $\delta = 0.95$ . The cost of the optimal solution of GAA for L is estimated using Query Analyzer Tool of the SQL Server 2000 and it is 0.085440.

For validating the model, 10 secondary index sets are chosen randomly using appropriate random variate generation techniques. The cost of each index set is estimated for the same workload using Query Analyzer of the SQL Server 2000 and shown in Table 4 and Figure 5. It is clearly evident that these estimated costs are higher than that of the optimal secondary index set determined employing GAA.



**Figure 4: Schema diagram for Road Transport Department Application**

**Table 2: Workload**

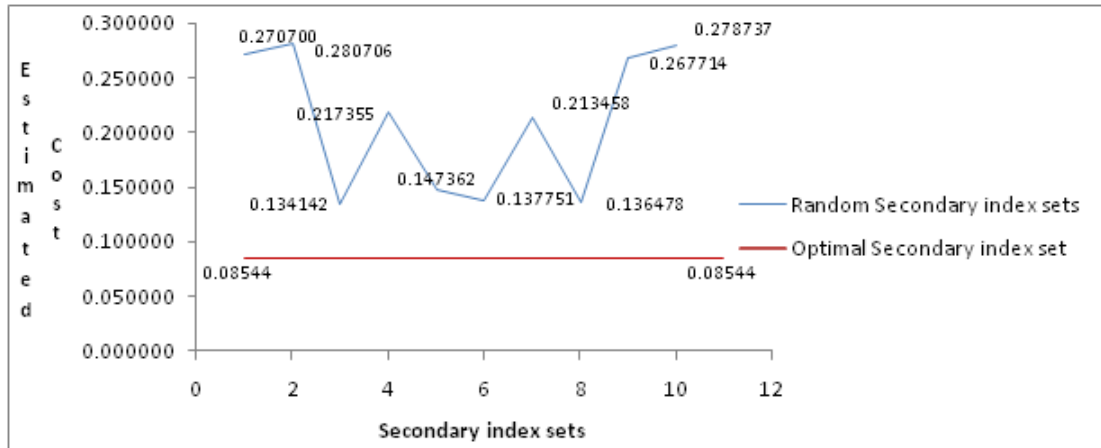
| S.No. | Planned queries in L   | $p_i$ |
|-------|--|-------|
| 1     | List the details of all the vehicles owned by an owner given his name.   | 0.50  |
| 2     | List the details of all the vehicles owned by an owner given his unique identity.  | 0.05  |
| 3     | List the details of all the vehicles given the city.   | 0.25  |
| 4     | List the details of all the vehicles and their owner details given a manufacturing company, model and cylinder capacity.   | 0.10  |
| 5     | List the details of all the vehicles owner details given a manufacturing company, model, cylinder capacity and city.   | 0.05  |
| 6     | List the details of all the vehicles and owners insured their vehicles at the given insurance company.   | 0.01  |
| 7     | List the details of all vehicles the name of a given owner and insurance company.  | 0.01  |
| 8     | List the details of insurance policy and owner given the unique identity of a vehicle.   | 0.01  |
| 9     | List the details of all the vehicles and their owners given the unique identity of Regional Transport office.  | 0.01  |
| 10    | List the details of all the vehicles and their owners given the unique identity of Regional Transport office, make, model, Cubic capacity and month-year of manufacture. | 0.01  |



|    |   |                                  |
|----|---|----------------------------------|
| 7  | select vNo from icompany, policy, ov, owner<br>where cName='Maruti Insurance Co.' and icomId=cId and pNo=policyNo and oFName='Karthik' and oLName='Amber' and oId=ownerId;                                | 01100000000000000101010000100000 |
| 8  | select policyNo, idv, pPFrom, pPTo, oFName, oLName from vehicle, ov, policy, owner where engineNo='dxap123456' and chasisNo='cxdx567890' and vNo=vRegdNo and policyNo=pNo and ownerId=oId;                | 10000000000000001101010000000000 |
| 9  | select vRegdNo, vRegdDate, oFName, oLName from rto, vehicle, ov, owner<br>where city='Tirupati' and rtoId=rId and vNo=vRegdNo and ownerId=oId;  | 10000000001000000010000000000010 |
| 10 | select * from rto, vehicle, ov, owner<br>where city = 'Tirupati' and rtoId = rId and cubicCapacity=1000 and make = 'Maruti Co.' and model = 'Dzire' and myom = '03/1999' and vNo=vRegdNo and ownerId=oId; | 10000000001111100010000000000010 |

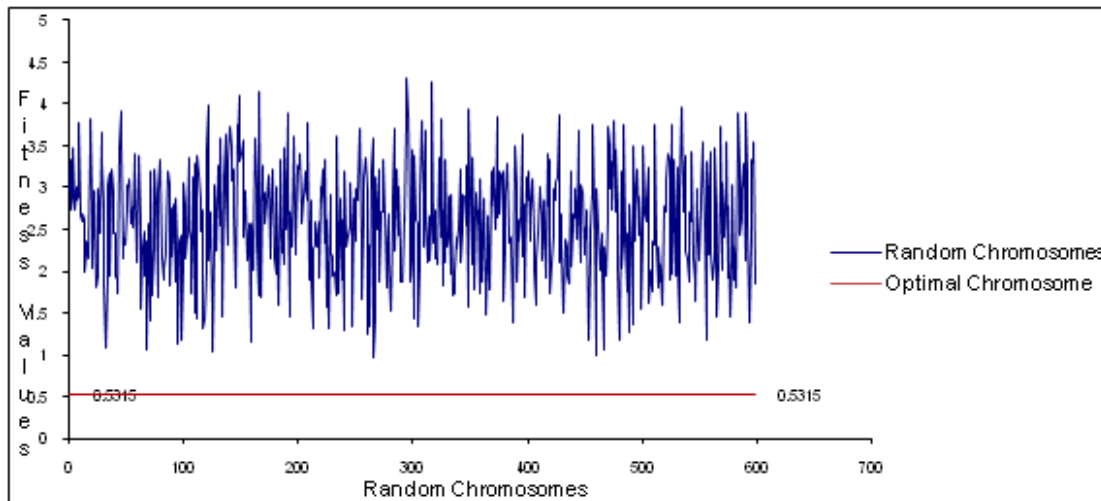
**Table 4: Random secondary index sets, in bit strings, and their estimated costs, using Query Analyzer of SQL Server 2000, for the workload L**

| Secondary index set               | Estimated Cost |
|-----------------------------------|----------------|
| 100000110001011100101100010101010 | 0.270700       |
| 100000110001011100101100010101010 | 0.280706       |
| 00110001111010001010110010100101  | 0.134142       |
| 10000100011001001111101111100111  | 0.217355       |
| 00100010000101010001001000111110  | 0.147362       |
| 01010011010001110110100101000100  | 0.137751       |
| 00010100011010100100011001100010  | 0.213458       |
| 10110011110010111000000110110011  | 0.136478       |
| 10001001101011010011101000010011  | 0.267714       |
| 10011010001010100010011010010000  | 0.278737       |



**Figure 5: Estimated Costs, using Query Analyzer of SQL Server 2000, for the workload L with random secondary index sets and optimal secondary index set**

Further, the expected fitness values of randomly generated secondary index sets are computed and they are found to be higher than that of the optimal secondary index set determined employing GAA as shown in Figure 6. Hence, it is concluded that the model is valid.



**Figure 6: Expected fitness values, using GAA, for the workload L with random secondary index sets and optimal secondary index set**

### 7 Sensitivity Analysis

Figure 7 describes the simulation procedure designed to perform sensitivity analysis. The predefined methods used in the simulation procedure are described hereunder:

- 1) *Generate workload, L*: This procedure generates n random bit strings of length

D each. In the bit strings, each bit is discrete uniform variant  $[0, 1]$ . Each bit string represents a query in  $L$ . It also generates  $n$  random associated probabilities of occurrences of queries in  $L$ .

- 2) *Determine optimal secondary index set*: This procedure returns an optimal set of secondary indexes for multiple relations for the given  $L$  as described in Figure 3.

Sensitivity analysis is performed varying  $\delta$  from 0.55 to 0.95 and  $n$  from 10 to 100. For each combination of  $\delta$  and  $n$ , 10 samples of workload are considered. The observed fitness values for workload size, 100, varying  $\delta$  from 0.55 to 0.95 for 10 samples are given in Table 5. It is clearly observed, from Table 5, that the expected fitness values for workload are sensitive to the changes in the values of the input parameters called  $\delta$ , and queries in  $L$  of the proposed model.

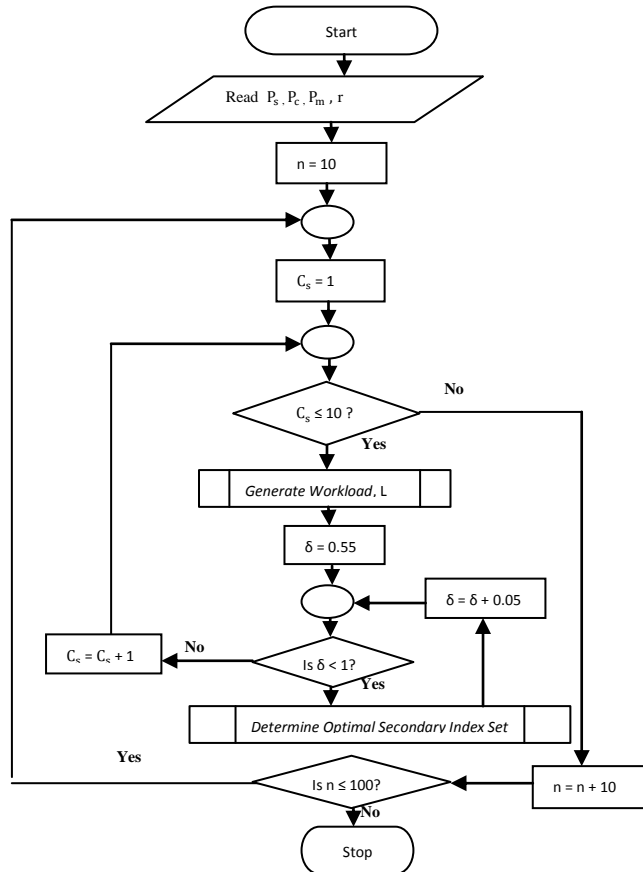
### 7.1 Observations from the sensitivity analysis

The following observations have been made from the analysis on the results obtained from the simulation procedure:

- 1) It is observed that as  $\delta$  increases the number of attributes of relation(s) not indexed which have to be indexed decreases in the optimal secondary index set for multiple relations for the given workload  $L$ . So the fitness value of the optimal secondary index set decreases as depicted in Table 5. Hence, the DBA may select appropriate  $\delta$  by considering the tradeoff between the access time to answer the queries in  $L$  and the required storage space for secondary indexes of relation(s).
- 2) Obviously, workloads differ in sizes, differ in the number of attributes and / or selected attributes of relation(s) in their optimal secondary index set and so the expected fitness values.
- 3) It is also observed that different workloads of the same sizes called samples differ in number of attributes and / or selected attributes of relation(s) in their optimal secondary index set and so the expected fitness value.

**Table 5: Expected fitness values of an optimal secondary index set of samples for workload of size  $n = 100$  for  $\delta$  from 0.55 to 0.95**

| workload size $n = 100$ |                 |                |                 |                |                 |                |                 |                |                 |
|-------------------------|-----------------|----------------|-----------------|----------------|-----------------|----------------|-----------------|----------------|-----------------|
| Samples                 | $\delta = 0.55$ | $\delta = 0.6$ | $\delta = 0.65$ | $\delta = 0.7$ | $\delta = 0.75$ | $\delta = 0.8$ | $\delta = 0.85$ | $\delta = 0.9$ | $\delta = 0.95$ |
| s1                      | 0.21528         | 0.19506        | 0.17340         | 0.16560        | 0.14480         | 0.13349        | 0.12033         | 0.06684        | 0.05013         |
| s2                      | 0.22393         | 0.20393        | 0.17951         | 0.17451        | 0.14430         | 0.12754        | 0.11768         | 0.05129        | 0.02564         |
| s3                      | 0.22037         | 0.20956        | 0.18488         | 0.16438        | 0.12709         | 0.11308        | 0.09300         | 0.09139        | 0.07328         |
| s4                      | 0.21484         | 0.19684        | 0.17922         | 0.16928        | 0.13488         | 0.11064        | 0.10955         | 0.07686        | 0.07419         |
| s5                      | 0.22045         | 0.20238        | 0.18907         | 0.16234        | 0.13738         | 0.11900        | 0.11287         | 0.07351        | 0.04956         |
| s6                      | 0.21261         | 0.19345        | 0.17688         | 0.16687        | 0.13709         | 0.09659        | 0.09480         | 0.08447        | 0.07934         |
| s7                      | 0.22077         | 0.19957        | 0.18168         | 0.15783        | 0.13310         | 0.11267        | 0.10535         | 0.09089        | 0.06575         |
| s8                      | 0.22703         | 0.20946        | 0.18483         | 0.16417        | 0.13930         | 0.11977        | 0.09348         | 0.09115        | 0.04943         |
| s9                      | 0.22517         | 0.20809        | 0.17870         | 0.17717        | 0.14655         | 0.10762        | 0.10124         | 0.08993        | 0.04983         |
| s10                     | 0.22230         | 0.19814        | 0.18193         | 0.15605        | 0.14827         | 0.12257        | 0.09576         | 0.08981        | 0.05240         |



**Figure 7: Procedure to perform Sensitivity Analysis**

## 8 Conclusions

A Genetic Algorithmic Approach is proposed to determine an optimal secondary index set for multiple relations for a given workload to minimize the effort so as to improve the system performance. For validating the model and applicability of GAA, a database, usually applicable to any Road Transport Department, is designed. The cost of the optimal secondary index set determined using GAA and 10 randomly generated secondary index sets for a given workload is estimated employing Query Analyzer of SQL Server 2000. It is observed that the estimated cost of the secondary index set determined using GAA is less than that of randomly generated secondary index sets. Further, the fitness values for 600 randomly generated chromosomes are higher than that of the optimal chromosome determined using GAA. Hence, it is concluded that model is valid. Further, sensitivity analysis is performed varying the values of input parameters of the model  $\delta$ ,  $n$  and  $C_s$ . It is observed that the fitness values of the optimal chromosome are sensitive to the changes in input parameters of the model. It facilitates the database administrator to determine the optimal chromosome for the given  $\delta$  and  $L$ .

## References

- [1] Chaudhuri, S., Christensen, E., Graefe, G., Narasayya, V., and Zwilling, M., 1999, "Self-Tuning Technology in Microsoft SQL Server," Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, pp 20-26.
- [2] Chaudhuri, S., and Narasayya, V., 1998, "AutoAdmin "What-if" Index Analysis Utility," Proc. of ACM SIGMOD ICMD, pp 367-378.
- [3] Sattler, K., Schallehn, E., and Geist, I., 2004, "Autonomous Query-driven Index Tuning," Proc. of IDEAS, pp 439-448.
- [4] Silberschatz, A., Korth, H.F., and Sudarshan, S., 2002, Database System Concepts, 4<sup>th</sup> ed., McGraw-Hill International Edition, India.
- [5] Elmasri, R., and Navathe, S.B., 2004, Fundamentals of Database Systems, 4<sup>th</sup> ed., Pearson Education, India.
- [6] Grant, F., 2012, SQL Server Execution Plans. [Online]. Available: <http://www.red-gate.com/community/books/>
- [7] Delaney, K., Inside Microsoft SQL Server 2000, 2001, WP Publishers & Distributors (P) Limited, India.
- [8] Goldberg, D.E., 2004, Genetic Algorithms in Search, Optimization & Machine Learning, 7<sup>th</sup> reprint, Pearson Education (Singapore) Pte. Ltd., India
- [9] (2014) WIKIPEDIA The Free Encyclopedia web site. [Online]. Available: [http://en.wikipedia.org/wiki/Genetic\\_algorithm](http://en.wikipedia.org/wiki/Genetic_algorithm)
- [10] Ip, M. Y. L., Saxton, L. V., and Raghavan, V.V., 1983, "On the Selection of an Optimal Set of Indexes," IEEE Transactions on Software Engineering, vol. SE-9, No. 2, pp 135-143.
- [11] Choenni, S., Blanken, H., and Chang, T., 1993, "Index Selection in Relational Databases," Proc. of ICCI, pp 491-496.

- [12] Anderson, H.D., and Berra, P.B., 1977, "Minimum Cost Selection of Secondary Indexes for Formatted Files," *ACM Transactions on Database Systems*, vol. 2, No.1, pp 68-90.
- [13] Chaudhuri, S., and Narasayya, V., 1997, "An Efficient, Cost-Driven Index Selection Tool for Microsoft SQL Server," *Proc. of ICVLDB*, pp 146-155.
- [14] Choenni, S., Blanken, H., and Chang, T., 1993, "On the Selection of Secondary Indices in Relational Databases," *Data & Knowledge Engineering*, vol. 11, pp 207-233.
- [15] Comer, D., 1978, "The Difficulty of Optimum Index Selection," *ACM Transactions on Database Systems*, vol. 3, No.4, pp 440-445.
- [16] Schkolnick, M., 1975, "Secondary Index Optimization," *Proc. of the 1975 ACM SIGMOD ICMD*, pp 186-192.
- [17] Barcucci, E., Pinzani, R., and Sprugnoli, R., 1990, "Optimal Selection of Secondary Indexes" *IEEE Transactions on Software Engineering*, vol. 16, No. 1, pp 32-38.
- [18] Lum, V.Y., and Ling, H., 1971, "An Optimization Problem on the Selection of Secondary Keys," *Proc. of the 1971 26<sup>th</sup> Annual Conference ACM*, pp 349-356.
- [19] Piatetsky-Shapiro, G., 1983, "The Optimal Selection of Secondary Indices is NP-Complete," *ACM SIGMOD Record*, vol. 13, No.2, pp 72-75.
- [20] Caprara, A., Fischetti, M., and Maio, D., 1995, "Exact and Approximate Algorithms for the Index Selection Problem in Physical Database Design," *IEEE Transactions on Knowledge and Data Engineering*, vol. 7, No.6, pp 955-967.
- [21] Schkolnick, M., 1975, "The Optimal Selection of Secondary Indices for Files," *Information Systems*, vol. 1, pp 141-146.
- [22] Jong, K.A.D., and Spears, W.M., 1989, "Using Genetic Algorithms to Solve NP-Complete Problems," *Proc. of ICGA*, pp 124-132.
- [23] Bhandari, D., Murthy, C.A., and Pal, S.K., 2012, "Variance as a Stopping Criterion for Genetic Algorithms with Elitist Model," *Fundamenta Informaticae*, vol 120, pp 145-164.
- [24] Chaudhuri, S., and Narasayya, V., 1998, "Microsoft Index Tuning Wizard for SQL Server 7.0," *Proc. of ACM SIGMOD ICMD*, pp 553-554.
- [25] (2014) Microsoft website. [Online] Available: [http://technet.microsoft.com/en-us/library/aa933123\(d=printer,v=sql.80\).aspx](http://technet.microsoft.com/en-us/library/aa933123(d=printer,v=sql.80).aspx)
- [26] Raja Kumar Reddy, N., and Naidu, M.M., 2014, "Selection of Optimal Secondary Index Set for a Single Relation: A Genetic Algorithmic Method," *IJAER*, vol 9, No 21, pp 10073-10086.

