ISSN:0975-9646

Query Optimization Strategies in Distributed Databases

Shyam Padia, Sushant Khulge, Akhilesh Gupta, Parth Khadilikar

Computer Science Department, Mumbai University
Mumbai, India

Abstract- *The query optimization problem in large-scale distributed databases is NP-hard in nature and difficult to solve. The complexity of the optimizer increases as the number of relations and number of joins in a query increases. Research is being carried out to find an appropriate algorithm to seek an optimal solution especially when the size of the database increases. Various Optimization Strategies have been reviewed in this paper and the studies show that the performance of distributed query optimization is improved when Ant Colony Optimization Algorithm is integrated with other optimization algorithms.*

Keywords: Query Execution Plan, Distributed Database, Ant Colony Algorithm, Search Strategies

1. INTRODUCTION

With the launch of high speed communication networks, significant research is devoted to developing highly efficient techniques for processing complex queries in a cost effective manner in a Distributed Database Environment. A Distributed Database is a collection of logically interrelated database distributed over a computer network so as to improve the performance, reliability, availability and modularity of the distributed systems. Query processing is much more difficult in Distributed Environments than in Centralized Environments. Since the data is geographically distributed onto multiple sites, the processing of query involves transmission of data among different sites. The retrieval of data from different sites is known as Distributed Query Processing (DQP). The query processor selects data from databases located at multiple sites in a network and performs processing over multi le CPUs to achieve a single query result set. There are three phases involved in Distributed Query Processing [1][9][10][12]:

Local Processing Phase: In this phase, the initial Algebraic Query specified on global relations is transformed in to fragments (Data Decomposition) and made available to the respective sites for processing (Data Localization) like local selections and projections.

Reduction Phase: A sequence of Joins and Semi Joins (reducers) are used to minimize the amount of data i.e. the size of the relations that needs to be transmitted in order to accomplish a join operation in a cost effective manner.

Final Processing or Assembly Phase: In this phase all the processed files are transmitted to the assembly site for the generation of final output.

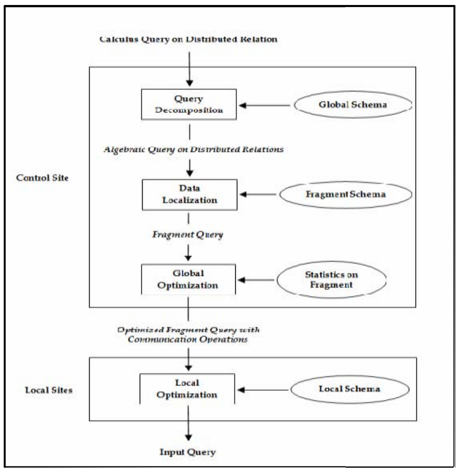

Fig1. Distributed Query Optimization Model [4].

The performance of a distributed query is critically dependent upon the ability of the query optimizer to derive efficient query processing strategies [2]. Query Optimization is one of the most important and expensive stages in executing distributed queries. The complexity of the optimization process is determined by the number of relations referenced, types of initial query access methods and the set of rules involved for generating possible query trees or query graphs.

Once the query, entered by the user, is transformed into a standard relational algebra form, the optimizer searches for an optimal query execution plan [3]. The number of possible alternative query plans increases exponentially with increase in the number of relations required for processing the query. The query optimizer needs to explore the large search space for generating optimal query plans. The query optimization problem in large-scale distributed databases is NP-hard [9] [15] in nature and difficult to solve as exploring all the query plans in this large search space is not feasible. This problem in Distributed Databases is a Combinatorial Optimization problem and has been addressed by various techniques like simulated annealing, iterative improvement, two-phase optimization, Deterministic, Greedy and Heuristic Algorithms to find an optimal solution by taking the time and cost complexity of executing these queries into consideration [4] [5].

In this paper an attempt has been made to study the various Search Strategies that can be implemented to determine Optimal Query Execution Plans in the processing of Distributed Queries. The remainder of this study is as follows. Section 2 discusses the components of Distributed Query Optimization. In Section 3, various Solution Algorithms that have been applied by scientist for query optimization are discussed and finally section 4 concludes the research paper and provides scope for future studies.

2. COMPONENTS OF DISTRIBUTED QUERY OPTIMIZATION

Query optimization is a difficult task in Distributed Environment because of numerous factors like data allocation, speed of communication channel, indexing, availability of memory, size of the database, storage of intermediate result, pipelining and size of data transmission [6]. The role of Query Optimizer is to produce Query Execution Plans (QEP) which represents an execution strategy of the query with minimum cost. An optimizer is a software module that performs optimization of queries on the basis of three important components of a query i.e. Search Space, Search Strategies and Cost Models.

2.1 Search Space

It refers to the generation of sets of alternative and equivalent QEPs of an input query by applying Transformational Rules such that they differ in the execution order of the operators [7]. The QEPs are commonly referred to as Operator Trees or Join Trees whose operators are various types of Joins or Cartesian Products. This can be represented as a query graph (annotated tree) denoted as G = (N,A) where N is the set of nodes(vertices) in the Query Graph and A is the set of arcs (edges) [2][8]. Each node represents a set of Base File (BF) in the join specification of the query. Two nodes are connected by an arc if the query joins the two corresponding files. Each node in the query graph has an associated site set. These leaf nodes represent file materializations resulting from local processing and it is a reduced file. The root node represents the final step where the query result is generated. Each parent node uses the result files of its children as its inputs.

For example, file BF_0 is stored at sites S_1 and S_2; file BF_1 at site S_3; file BF_2 at site S_4; and file BF_3 at sites S_3 and S_4. Consider a query that joins a sequence of four files, referenced BF_0 through BF_3. Join attributes are designated A_0 through A_2 (i.e. four files are joined using three join operations). In SQL this query is specified as:
select A_1, <non-join projection attribute list>
from BF_0, BF_1, BF_2, BF_3
where $BF0.A0 = BF1.A0$ and $BF1.A1 = BF2.A1$ and $BF2.A2 = BF3.A2$

The query graph for the distributed query, augmented with the site set for each file, is shown below.

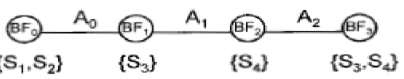

Fig 2. Query Graph for Distributed Query [11]

Since a query graph is a complicated query tree modeling various join operations, the QEP for the given query can be represented as follows:

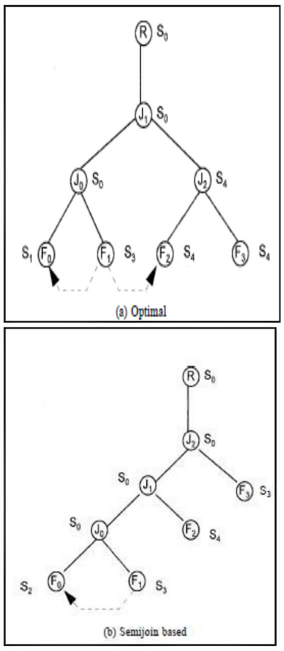

Fig 3. Query Execution Plan for above Query [8]

In a Distributed Database Query involving many relations and many joins, the number of QEP increases exponentially because of the Associative and Commutative property of the Join Operation. (O (N!) where N is the number of relations). The query optimizer needs to explore the large search space for generating optimal query plans.

The task of Query Optimizer is to [9][15]:

a) Determine the Order of Execution of Join Operation.
b) Determine the Join Operation Access Methods.
c) Determine the shape of the Join Query Graph in the given search space by implementing the appropriate search strategy such that the performance measure of the resulting Query Execution Plan is optimized. It can be Linear Trees (Left Deep Trees), Right Deep trees, Bushy Join Trees, Binary Trees or Zig-Zag Trees.
d) Determine the order of data movements between the sites (Join Sites) so as to reduce the amount of data and cost on the communication network.

To explore all the query plans in this large search space various Search Strategies have been researched upon to find optimal plans.

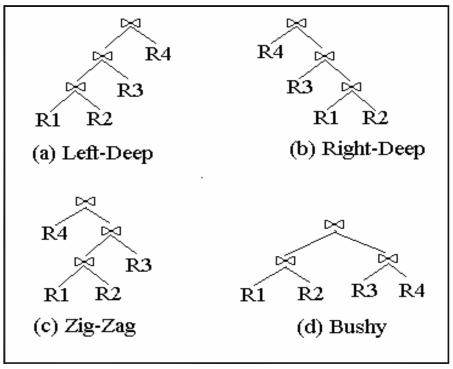

Fig 4. Shapes of Query Join Graphs [15]

2.2 Search Strategy

It refers to the algorithms applied to explore the search space and determine the best Query Execution Plan (QEP) based on Join Selectivity and Join Sites so as to reduce the cost of query optimization. There are basically two classes of strategies that solve the problem of Join Scheduling [5] for Query Optimization.

The first approach is Deterministic Strategy that proceeds by building plans, starting from base relations, joining one or more relations at each step till complete plans are obtained. To reduce the optimization cost, the plans that do not lead to optimal solutions are pruned [2]. The Dynamic Programming builds all such plans using Breadth-first search while Greedy Algorithms uses depth-first search.

The other approach is Randomized Strategies [11][17] that search the optimal solution around some particular points. These strategies do not guarantee optimal plan but they avoid high cost of optimization in terms of memory and time consumption. Iterative Improvement and Simulated Annealing are common solution algorithms under these strategies.

One of the difficulties faced by the researchers in this area is tractability. The solution space grows exponentially when the number of relations and distributed data storage sites increases. The ability of the optimizer to address the issues of Join Order, Join Method, query data size reduction and reduction in the cost of the query is an extremely difficult task.

However, Evolutionary Algorithms like Ant Colony Optimization, Genetic Algorithms and Particle Swarm Optimization are now being studied to find optimal and suboptimal solution for the large join queries in the given search space that are processed by Relational and Distributed Databases. These algorithms are particularly successful because of their global searching capability, robust nature and their ability to handle different combinatorial optimization problems. The various types of search strategies have been discussed in detail in section 3.

2.3 Cost Model

The Objective of Query Optimization in Distributed Database Environment is to minimize the total cost of computer resources. An optimizer cost model includes cost functions to predict the cost of operators and formulas to evaluate the sizes of the results [16]. The cost function can be expressed with respect to either total time or response time. The total time is inclusive of Local Processing Cost (CPU Time + I/O Cost), Communication Cost (Fixed time to initiate a message
+ Time to transmit a data). Minimizing the total time implies that the utilization of resources increases thus increasing the system throughput. The Response Time is evaluated as the time elapsed between initiation and completion of a query including parallelism. In parallel transferring, the response time is minimized by increasing the degree of parallel execution [11].

Primarily, the cost of a query depends on the size of intermediate relation that are produced during execution and which must be transmitted over a network for a query operation at a different site. The emphasis is on the estimation of the size of the intermediate relations based on Join Orders and Join Methods so as to reduce the amount of data transfers and hence decrease the total cost and total time of the distributed query execution.

3. Solution Algorithms

The central component of a query optimizer is its Search Strategy or Enumeration Algorithm. In this section, the research on Query Optimization Techniques based on a number of Optimization Algorithms used in Distributed Database Queries is explored.

One of the early Distributed Database System was SDD-1 which was designed for slow wide area network and made use of semi joins to reduce the communication cost by generating static unchangeable query plans without considering the horizontal or vertical data fragments of distributed database [13]. The R* system used by DDB query optimizer also generated static unchangeable query in faster networks but they neither employed semi joins nor handled horizontal or vertical fragmentation [21]. Distributed–INGRES [2] was able to generate dynamic QEP at run-time on faster communication networks. Semi Join for the reduction of query size was not used but the system was able to handle horizontal fragmentation without replication.

Significant amount of work has been carried out on generating optimal solutions for Join Order of the query. The Randomized strategies [18] definitely reduce the cost of the query optimization but they have a constant space overhead issue and are slower than heuristics. Deterministic Strategies [19] generate runtime Dynamic solutions but they have exponential time and space complexity associated with them when the numbers of relations increases in the distributed query. The Two-Phase Optimization Algorithm [20] is a combination of Iterative Improvement and Simulated Annealing. It performs a random walk along different solutions of search space, generates an optimal solution but increases the space overhead of query optimization.
A dynamic programming based solution procedure to minimize the sum of communication cost and local processing cost by optimally determining Join Order and Join Methods (nested-loop or sort-merge) and Join Sites was also proposed by scientist. However they assumed data to be stored non-redundantly [21][22]. Chen and Yu proposed a heuristic algorithm that determined the Join Order and Join Sites with an assumption that file copies are pre-selected when multiple copies exists [14].

A lot of research focused on the reduction phase of the distributed query processing where the objective is to find a minimum cost semi join sequence that fully reduces the files referenced by the query. This is achieved by applying join predicates in a query plan in order to reduce the size of the intermediate query results thus reducing the cost of the operation. An algorithm based on Dynamic Programming that identified beneficial semi joins and determined Join Order and Join Sites was worked upon by scientist [23]. Two new concepts in the reduction phase of Distributed Database: Gainful Semi joins and pure join attributes [24] was also proposed. Mi Xifieng and Fan designed a new algorithm based on the commonly used optimization algorithms to significantly reduce the amount of intermediate data and network communication cost and to improve the optimization efficiency [25].

Since Dynamic Programming is not a feasible solution optimization of Distributed Queries because of its large space complexity and complicated objective functions, new set of techniques like Genetic Algorithm, Ant Colony Optimization Algorithm and Hybrids of various Evolutionary Algorithms are now being explored for solutions to Distributed Queries.

Genetic Algorithms are a family of Computational models inspired by nature or Biological Evolutions. The concept of GA was proposed by John Holland where randomly generated solutions to a problem are evaluated as chromosomes and these chromosomes are allowed to produce new set of individuals or children with better characteristics via crossover and mutations operators based on fitness function [28]. The algorithm was also able to reduce the cost of the distributed query tree.

Ant Colony Optimization Algorithm (ACO) is a novel meta-heuristic algorithm which is suitable for problems related to Combinatorial Optimization. Like query optimization in distributed database because of its characteristics like intelligent search techniques, global optimization, robust, distributed computing and ability to combine with other heuristics [31]. ACO was first proposed by three Italian scholars, Dorigo M, Colorni A and Maniezzo V in 1992.It is a Bionic Optimization Algorithm inspired by Ants that uses probabilistic technique for solving computational problems. It is built on the mechanism of positive feedback, so it is very robust, provides intelligent search and can be used for Global Optimization Solutions [32].
Inspite of Ant Colony Optimization Algorithm having special characteristics like distributed computing, robust nature and positive feedback mechanism, ACO has some deficiencies:

a) The initial formation needed by ACO has no systematic way of startup.
b) The convergence speed of ACO is lower at the beginning for there is only a little pheromone difference on the path at that time but the convergence speed increases towards optimum answer because of positive feedback mechanism.

In [11], the scientist proposed a new GA based query optimizer called NGA that improves a given QEP by considering Join Order, Join Site and Semi join reducers. The algorithm was able to reduce the Local Processing Cost and Network Communication Cost by using new values for mutation and crossovers and outperformed Exhaustive Search. The potential of Genetic Algorithm to optimize distributed queries on the problem of fully reducing all the tables in a tree structured data model query was also worked upon [26]. A combination algorithm of Genetic Algorithm and Learning Automata [27] for producing optimal Query Execution Plans on the basis of the Join Order Execution and Join Site Selection in distributed database was also proposed by the scientist. In [28], author proposed a combination of GA and Heuristics for solving Join Ordering Problems as Travelling Salesman Problem in Large Scale Database and the computational experiments proved it to be a viable solution for Distributed Systems also. Hybrid of GA and Best-Worst Ant Colony Optimization (BWACO) [29] was implemented to find out the optimal Query Execution Plan and Join Order by reducing Query Execution Time for Multi-Join Query Optimization in Database. The algorithm implemented positive feedback mechanism of ACO with global search capability of GA. Another Hybrid of GA and ACO [30] was implemented on Join Ordering problem in Databases (only nested loop joins considered) by overcoming the shortcomings of both the algorithms. The algorithm adopts Genetic Algorithm to give pheromone to distribute. And then it makes use of ant colony algorithm to give the precision of the solution.

The capability of Hybrid GA-ACO to search extensive amplitude to answers for join queries in relational Database can be extended to optimize the join queries in distributed database where the most important challenge is to generate the best QEP for optimal results. Rho et.al [8] proposed a Genetic Algorithm based solution procedure to quickly determine optimal QEP. This model includes Copy Identification (redundancy of data), Beneficial Semi joins identification, Join Site Selection, Join Order Execution, and Local Processing Cost and Communication Cost.

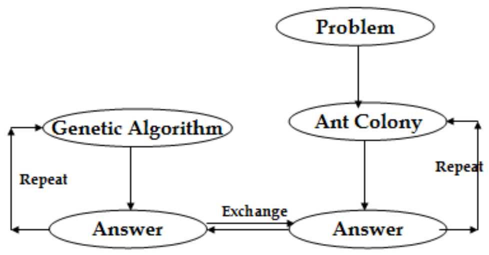

Figure 5. Diagram for Hybrid GA-ACO [30]

Tansel et.al. [33] proposed DP-ACO (Dynamic Programming- Ant Colony Optimization) algorithm for the optimization of multi way chain equijoin queries in Distributed Database Environment. Dynamic Programming suffers from long execution times and very large memory requirements as the size of the relations and number of joins increases in the query. DP-ACO have proved to be viable solution by producing good execution plans with 15 way join queries within limited time and very limited memory space. Another advantage of DP-ACO is that is that they can be easily adapted to existing query optimizers that commonly use DP-based algorithms.

By use of the properties of Ant Colony Algorithm and Particle Swarm Optimization, a hybrid algorithm is proposed to solve the traveling salesman problems [35]. The algorithm first adopts statistics method to get several initial better solutions and in accordance with them, gives information pheromone to distribute. Then it makes use of the ant colony algorithm to get several solutions through information pheromone accumulation and renewal. Finally, by using across and mutation operation of particle swarm optimization, the effective solutions are obtained. The Hybrid Algorithm of ACO-PSO has proved to be effective.

With the increasing number of relations in a query, much use of memory and processing is needed. DDBMS is now being used as a standard DBMS in all commercial applications which involve data from various sites. The path marking the behavior of ants is applied to direct the ants towards the unexplored areas of search space and visit all the nodes without knowing the graphic topology for generation of optimal solutions of distributed database queries. These ants calculate the running times of the execution plans of the given query and provide quick, high performance and optimal results in a cost effective manner.

The Search strategy adopted by the Query Optimizer in Distributed Database Management System can help to reduce the query execution time and the cost incurred on the query and hence increases performance of a query by selecting the best Query Execution Plan. The implementations of these probabilistic algorithms have proved to generate viable solutions when the size of the query and the number of joins in the query grows.

CONCLUSION

The realization of hybrids of Ant Colony Optimization Algorithm towards the optimization of distributed database queries is still a novice field. Research in the creation and implementation of hybrids of ACO to solve various types of optimization problems are in progress. The results proved that hybrids of ACO are effective and viable in optimization problems. Research has shown that the implementations of these probabilistic algorithms have proved to generate viable solutions in distributed as well as relational database management system when the size of the query and the number of joins in the query grows. There is still a lot of opportunity to generate optimized solutions and to refine search strategies using hybrids of ACO for the Queries in Distributed Database especially when the size and complexity of the relations increases with a number of parameters influencing the query.

REFERENCES

[1] C.Yu, Z M Ozoyoglu, K. Kam," Optimization of Distributed Tree Queries", J.Comput. Sys. Sci, Vol 29, No 3, pp 409-445, 1984.

[2] S.Ceri, G. Pologatti, "Distributed Database Principles and Systems", Mc GrawHill.

[3] A. Hameurlain, F. Morvan, "Evolution of Query Optimization Methods", Trans. on Large Scale Data and Knowledge Cent. Syst.I, LNCS 5740, pp211-242, 2009.

[4] R. Ghaemi, AM Fard, Md. NB Sulaiman, " Towards Optimal Query Execution in Data Grids", Advanced Technologies, pp 57-72, 2005.

[5] A. Aljanaby, E. Abuelrub, M.Odeh, "A Survey of Distributed Query Optimization", The International Arab Journal of Information Technology, Vo2, No.1, 2005.

[6] Craig S. Mullins, "Distributed Query Optimization", Technical Support 2006.

[7] D.Abdullah, "Query Optimization in Distributed Databases",

[8] S. Rho, T. March, " Optimizing Distributed Joins Queries: A Genetic Approach", Annels, of Operations Research, Science Publishers, pp 199-288, 1997.

[9] M.Chen, P.Yu, "Using Join Operations as Reducers in Distributed Query Processing", Proceedings of 2nd Intl. Symp. on Databases in Parallel and Distributed System, July 1990.

[10] PMG Apers, AR Henver, SB Yao, "Optimization Algorithms for Distributed Queries", IEE Transactions on Software Engineering, Vol. 9, no.1, pp 57-68, January 1983.

[11] E. Sevinc, A. Cosar, "An Evolutionary Genetic Algorithm for Optimization of Distributed Database Queries",The Computer Journal, Vol. 54, No.8, 2011.

[12] C.Wang, M.Chen, "On Complexity of Distributed Query Optimization", IEEE Transactions on Knowledge and Data Engineering, Vol. 8, No. 4, August 1996.

[13] PA Bernstein, N Goodman, E.Wong, C. Reeve, "Query Processing in a System for Distributed Database (SDD-1)", ACM Trans. Database Sys., Vol. 6, No. 4, pp 602-625, December 1981.

[14] MS Chen, PS Yu, "Interleaving Join Sequence with Semijoins in Distributed Query Processing", IEEE Transactions, Issue 5, Vol. 3, August 2005.

[15] S. Pramanik, D. Vineyard, "Optimizing Join Queries in Distributed Database" IEEE Trans, Software Engg. Vol. 14, pp 1391-1426, September 1988.

[16] D. Sukheja, Umesh kr. Singh, " A Novel Approach of Query Optimization for Distributed Database System", IJCSI, Vol. 8, Issue 4, No. 1, July 2011.

[17] AN Swami, A. Gupta, "Optimization of Large Join Queries in Distributed Database", Proc. of ACM-SIGMOD Conference on Management of Data, pp 8-17, 1988.

[18] YC Kang, "Randomized Algorithms for Query Optimization", PhD Thesis, University of Wisconsin-Madison, 1991, Technical Report # 1053.

[19] M. Hussein, F. Morvan, A. Hameurlain, "Dynamic Query Optimization", 19th Intl. Conf. on Parallel and Distributed Computing Systems, ISCA, 2009.

[20] D. Kossman, "The State of Art in Distributed Query Processing", ACM Computing Survey, pp 422-469, 2000.

[21] LF Lohman, GM Mackert, "R* Optimizer Validation and Performance Evaluation for Distributed Queries", Proc. of 12th Intl. Conf. on VLDB, pp 149-159, 1986.

[22] D. Kossman, K. Stocker, "Iterative Dynamic Programming: A New Class of Query Optimization Algorithm", ACM Transactions on Database Systems, 2000.

[23] La Fortune, Wong, " Query Optimization>>>>

[24] M. Chen, P.Yu, "Combining Join and Semi Join Operations for Distributed Query Processing", IEEE Transactions on Knowledge and Data Engineering., Vol. 5, No. 3, 1993.

[25] M. Xifeng, F.Yuanyuan, "Distributed Database System Query Optimization Algorithm Research", IEEE Intl. Conf. on Computer Science and Information Technology", Vol.8, pp 657-660, 2010.

[26] M. Gregory, "Genetic Algorithm Optimization of Distributed Database Queries", IEEE World Congress on Computational Intelligence, pp 271-276, 1998.

[27] M. Naohoghdem, " Query Optimization in Distributed Database using Hybrid Evolutionary Algorithm, Intl. Conf. on Information retrieval and Knowledge Management, pp 125-130, 2010.

[28] Z. Zhou, "Using Heuristics and Genetic Algorithms for Large Scale Database Query Optimization", Journal of Information and Computing Science, Vol.2, No. 4, 2007.

[29] Y. Zhou, W.Wan, J. Liu, "Multi-Joint Query Optimization of Database Based on the Integration of Best-Worst Ant Algorithm and genetic Algorithm", Wireless Mobile Computing (CCWMC 2009), IET Intl. Communication Conference, pp 543, 2009.

[30] H. Kadhkhodaei, F. Mahmoudi, "A Combination Method for Joining Ordering Problem in Relational Database using Genetic Algorithm and Ant Colony", IEEE Trans. On Granular Computing, pp 312-317, 2011.

[31] Enxiu Chen1 and Xiyu Liu, "Ant Colony Optimization - Methods and Applications-Multi-Colony Ant Algorithm", Google Scholar, 2011, http://cdn.intechweb.org/pdfs/13584.pdf

[32] Marco Dorigo, Mauro Birattari, and Thomas Suiitzle, "Ant Colony Optimization- Artificial Ants as a Computational Intelligence Technique", IEEE Computational Intelligence Magazine, November 2006.

[33] Zar Chi Su Su Hlaing, May Aye Khine , "An Ant Colony Optimization Algorithm for Solving Traveling Salesman Problem", International Conference on Information Communication and Management, Singapore, IPCSIT vol.16 (2011), IACSIT Press.

[34] Tansel Dokeroglu, Ahmet Cosar, "Dynamic Programming with Ant Colony Optimization Metaheuristic for optimization of Distributed Database Queries", ISCIS:26th International Symposium on Computer and Information Sciences, IEEE, Vol 2 , pp.107-113, 2011

[35] Gao Shang' 2, Jiang Xin-zil, Tang Kezong', Yang Jingyu, "Hybrid Algorithm Combining Ant Colony Optimization Algorithm with Particle Swarm Optimization", Proceedings of the 25th Chinese Control Conference, Harbin, 7-11 August, 2006.