PSO-BASED RESOURCE SCHEDULING ALGORITHM FOR PARALLEL QUERY PROCESSING ON GRIDS

Arturo Pérez-Cebreros, Gilberto Martínez-Luna and Nareli Cruz-Cortés
Database and Information Systems Laboratory, Center for Computer Research
National Polytechnic Institute, Av. Juan Dios Batiz s/n, Zacatenco 07738, Mexico City, Mexico

Keywords: Particle Swarm Optimization, Genetic Algorithms, Grid, Database, Query Scheduler.

Abstract: The accelerated development in Grid computing has positioned it as promising next generation computing platforms. Grid computing contains resource management, task scheduling, security problems, information management and so on. In the context of database query processing, existing parallelisation techniques can not operate well in Grid environments, because the way they select machines and allocate queries. This is due to the geographic distribution of resources that are owned by different organizations. The resource owners have different usage or access policies, cost models, varying loads and availability. It is a big challenge for efficient scheduling algorithm design and implementation. In this paper, a heuristic approach based on particle swarm optimization algorithm is adopted to solving parallel query scheduling problem in grid environment.

1 INTRODUCTION

The emerging paradigm of grid computing and the construction of computational grids are making the development of large scale applications possible from optimization and other fields (Foster, 1998). However, this have increased the necessity of novel applications that require close and potentially sophisticated interaction and data sharing between resources that may belong to different organizations. Examples include the bio-informatics labs across the world sharing their simulation tools, experimental results; as well as the use of the donated spare computer time of thousands of PCs connected to the Internet in order to solve computation problems. Hence, the question is how database management systems and technologies can best be deployed or adapted for be used in such environments. As a consequence of this, the databases technologies have led to many proposals that try to integrate databases with Grid applications (Alpdemer, 2003; Liu, 2003; Narayanan, 2003). In particular, query processors for Grid-enabled databases, such as (Alpdemer, 2003; Smith, 2003) can provide effective declarative support for combining data access with analysis to perform non-trivial tasks, and are well suited for intensive applications as they naturally care for parallelism. This is due to the fact that many complicated tasks can be effectively encapsulated and specified by databases queries. However, one of the more difficult tasks for the efficient exploitation of parallelism in such query processors; it is scheduling a large number of queries, in order to achieve better performance. For the reason that many computing resources are geographically distributed under different ownerships, each having their own accesses policy, cost, and various constraints. Unlike scheduling problem in distributed systems, this problem is much more complex as new features of Grid systems such as its dynamic nature and the high degree of heterogeneity of tasks and resources must be tackled.

Systems like OGSA-DQP are capable of performing query processing over heterogeneous local database management systems, like mySQL and Oracle. Such systems may employ a number of machines to run the query processing tasks. Clearly, these machines can be heterogeneous as well, in terms of their computational capacity and characteristics. For instance, some may have high-speed interconnections, interfaces, operating systems, a larger amount of memory or a low power CPU. This kind of heterogeneity environment (HE) has motivated our project.

In this paper, the Particle Swarm Optimization (PSO) is employ to solve the scheduling problem in Grid enabled databases environment. The practicality of the approach lies in the fact that resource scheduling remains NP-complete even when an unlimited number of processors is available (Chrétienne, 1992), as a logical consequence of the NP-completeness of scheduling, the scientific community has been eager to investigate efficient scheduling algorithms based on heuristics or approximation techniques that produce near optimal solutions. In practice, scheduling must rely on these algorithms due to the intractability of the problem.

This paper is organized as follows. We present the problem in Section 2. A Particle Swarm Optimization (PSO) is introduced in Section 3. We compare our work with others in Section 4. Experiment settings and results are discussed in Section 5 and some conclusions are given in Section 6.

2 PROBLEM DESCRIPTION

It is well know that, even in homogeneous systems, choosing the maximum degree of parallelism not only harms the efficiency of resource utilization, but can also degrade the system's performance (Wilschut, 1992). Therefore, on Grid enabled, it holds as well.

Furthermore, the problem of query partition parallelism is discussed under the viewpoint of sending query fragments to many databases and executing the query in parallel. So the main focus is on executing a query fragment at any database as long as the cost for query execution is minimal. Therefore, it is clear that some optimizations and restrictions for data access and query execution are required. Unfortunately, finding a schedule of minimal length is in general a difficult problem. This becomes intuitively clear as one realizes that an optimal schedule is a trade-off between high parallelism and low inter-machine communication. On the one hand, query fragments should be distributed among the machines in order to balance the workload. On the other hand, the more the query fragments are distributed, the more inter-machine communications.

Another problem of resource selection for a query scheduler on emergent Grids is the budget; the integration of computational economy as part of scheduling system greatly influences the way computational resources are selected to meet the user requirements. The users should be able to submit their queries along with their requirements to a scheduling system.

Moreover, many of the resource management systems (e.g. (Stonebraker, 1994; Heiser, 1998; Amir, 2000; Buyya, 2000)) support a single model for resource trading, provide their own programming model and are implemented as monolithic systems. To overcome these limitations, the modern Grid computing systems use a layered architecture. Users and owners's resources have their own expectations and strategies for being part of the Grid (Buyya, 2001). In particular, the resource consumers adopt the strategy of solving their problems at low cost within a required time frame. The resource providers adopt the strategy of obtaining best possible return on their investment while trying to maximize their resource utilization by offering a competitive service access cost in order to attract consumers.

Indeed, the grid enabled database environment is dynamic and, also the number of resources to manage and the number of queries to be scheduled. These queries are usually very large making thus the problem a complex large scale optimization problem (as several optimization criteria such as response time and/or economic cost must to be matched).

3 PSO QUERY SCHEDULER ON GRIDS

3.1 Solution Approach

The complexity of the process of resource selection and creating a schedule S for a query graph G on a set of databases D is a key problem in emergent computational Grids. It should be obvious that generally there is more than one possible schedule for a given graph and a set of databases (which of course would have consequences for the scheduling of the other queries). Due that the usual purpose in employing a parallel system is the fast execution of a program; the usual aim is to produce a schedule of minimal length.

The algorithm proposed here receives a query plan which can be partitioned into sub-plans that can be evaluated on different machines. In our simulation, we assume the existence of a query optimizer which first constructs a single-node plan, and then transforms the single-node plan into a multi-node one, in order to reduce the search space (Kossman, 2000).

For example, exchange operators encapsulate the parallelism and involve communication (Graefe, 1990), and they naturally define the boundaries of different sub plans. Let us assume there are two databases, each containing one table, namely Orders and LineItem. The nodes of the plan are query operators (scans, projects, joins, and exchanges or data communication). These operators may be executed on different machines in parallel. But we assume that scans are not parallelisable, because data from any single table is accessed in an existing database. So suppose that the optimizer decides, using PSO Query Scheduler proposed here, that the join of this example, implemented as a hash join algorithm, should be cloned at any site, in order to deal with the second stage, where the single node plan is parallelized and how to improve it.

In particular, in grid-enabled databases considered here, a query is decomposed into a set of partitions S. Define |S| to be the number of partitions in the set S and si to be the ith partition, then S = {s_i, $0 \le i < |S|$}. A grid-enabled database consists of a set of heterogeneous databases D. Define |D| to be the number of databases in the set D an m_j to be the jth database, then M = {m_j, $0 \le j < |D|$}. The estimated expected execution time of partition si on database m_j is T_{ij}, where $0 \le i < |S|$ and $0 \le j < |D|$.

To formulate this problem under our simulation model, an estimation of the computational load of each partition, the computing capacity of each resource, and an estimation of the prior load (CPU, Network) of each one of the resources are required. Thus, we make the usual assumption that we know the computing capacity of each resource, the estimation or prediction of the computational necessities (workload) of each partition, and the load of prior work of each resource. This assumption is typically made for the current state-of-the-art in Heterogeneous Computing systems when studying the matching and scheduling problem (Freund, 1994; Singh, 1996; Shroff, 1996). Finding the estimated expected execution times for subtasks is another research problem, which is outside the scope of this paper.

3.2 Particle Swarm Optimization

PSO is a population-based search algorithm and is initialized with a population of random solutions, called particles (Hu, 2004). Unlike in other evolutionary computation techniques, each particle in PSO is also associated with a velocity. Particles fly through the search space with velocities which are dynamically adjusted according to their historical behaviour.

In particular, PSO learns the scenario and uses it to solve optimization problems. Thus, each single solution is like a 'bird' in the search space, which is called 'particle'. All particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. So the particles fly through the search space by following the particles (solutions) and then searches for optima by updating each generation, following equations will be used to compute the new elements of velocity and position vectors:

$$v_{id}(k+1)=X[w_k(k)v_{id}(k)+c_1r_1[p_{id}(k)-x_{id}(k)]+c_2r_2[p_{gd}(k)-x_{id}(k)]] \quad (1)$$

$$x_{id}(k+1)=x_{id}(k)+v_{id}(k+1) \quad (2)$$

where wk is the inertia weight, which represents the particle's preference to continue moving in the same direction it was going on the previous iteration, introduced in (Shi, 1998), X is the constriction coefficient, which serves as a balancing factor for the local and global search, introduced in (Clerc, 2002), c1 and c2 are cognitive and social factors respectively, often set equal to 2, k represents the iteration number, r1 and r2 are random numbers between [0,1], i(i=1,2,...,N) is the index representing the particles in the swarm and d (d=1,2,...n) is the index for dimensions of searching space. We define one particle as a possible solution in the population, and the fitness function is the minimum time/cost value. The main PSO algorithm is given below:

```
begin
   for i=1 to number of particles do
         Initialize position and
         velocity  randomly;
         Initialize the neighborhood;
   end
   repeat
         compute the fitness value
         G($x_i$);
         for i=1 to number of
         particles do
               if G($x_i$)>G($xpb_i$) then
                     for d=1 to number
                     of dimensions do
                           $xpb_{id}$=$x_{id}$;
                     end
               end
         select the local best
         position in    the
         neighborhood $lb_i$;
```

```
        for d=1 to number of
        dimensions do
            w=rand();
            $v_{id}$=w * $v_{id}$ +
            cir1(pb$x_{id}$-$x_{id}$) +
            c2r2($lb_{id}$-$x_{id}$);
            $x_{id}$=$x_{id}$+$v_{id}$;
        end
    end
    perform mutation;
    until    Maximum    number    of
    generations;
    scheduling resources;
  end
```

From a higher level point of view our approach transform an existing plan to a more efficient, by a PSO technique, modifying the set of resources allocated to a part of the query plan. Transformational approaches to query optimization have already been employed for constructing query plans (Ioannidis, 1996), however not for scheduling resources.

4 RELATED WORK

There is one previous work (Gounaris, 2006) that deals with the scheduling of resources in heterogeneous environments to support arbitrary degrees of partition parallelism; however, it mitigates the problem of devising a near optimal workload distribution of tuples among the selected databases. In other words, they assume that all the available machines are similar in terms of workload, so they simplified the problem of resource scheduling by neglecting the problems of a near optimal distribution due to its NP-hard complexity.

In the literature, different approaches can be found that relate to ours in certain ways. Due to its theoretical and practical relevance, the evolutionary computing research community has started to examine the Grid resource scheduling (Di, 2004; Abraham, 2000; Zomaya, 2001; Meijer, 2004; Braun, 2001). However, the existing approaches in the literature show several limitations: in some works just the uni-objective case is considered and usually either concrete grid environments. Moreover, the schedulers's performance has been studied only on small size instances. Dynamic aspects of this problem have not been addressed so far to Grid enabled databases environments.

Particularly in the database field, distributed query processing has mostly been influenced by some pioneering systems. The most influential, System R* (Mackert, 1986), but they simplified the problem of resource scheduling by neglecting the benefits of partitioned parallelism. In other words, the data are retrieved from a single site only, and are joined on a single site, which is either the site of one of the inputs or the site that asked for the data. Distributed Ingres (Epstein, 1978) took a step forward; they used a fragment and replicate query processing strategy. A query is partitioned into equal-sized fragments, each fragment is sent to a computer processor, and all other relations are replicated in all computer processors, however, different machines may have different processing speeds and/or different access methods for accessing required data, thus equal-distribution of fragments may lead to load imbalance. In (Rahm, 1995) discusses an approach for load balancing employing partitioned parallelism, although it refers to completely homogeneous environments, it does not force the system to employ all the available nodes when there are not needed. Other existing techniques for parallel and distributed databases (Garofalakis, 1997; Mayr, 2003, DeWitt, 1986) do not consider partitioned parallelism or completely ignore the resource selection phase by assuming a fixed set of resources then they try to schedule tasks over these resources. In other words, they also assume homogeneous and stable environments in terms of capabilities, connection speed and ownership.

5 EXPERIMENTAL SETTINGS AND RESULTS

For the evaluation of our proposal; we built a simulator by extending the cost model in (Sampaio, 2002), which is a detailed and validated simulator developed for parallel object database systems, it has been adapted to operate in a heterogeneous and autonomous environment and has been incorporated in the query engine.

In our simulation, we define 156 cost models, each one is a database; each one estimates the response time by estimating the cost of each operator instance separately in time units. The communication costs are also considered; these costs are composed of fixed costs per message, per-byte costs to transfer data, and CPU costs to pack and unpack tuples. Each database in our simulation is heterogeneous because we change several system parameters, which are described in Table 1.

We assumed that the processing speeds of each machine/resource, budget(G$) and network of each query are known. Hence, Expected Time to Compute matrix (ETC) is updated by the time units that are generated by our cost models; where the position ETC[i][j] indicates the expected execution time of partition i in machine m. This ETC matrix is needed in order to compute our fitness function.

Table 1: System parameters.

Description	Unit
Seek time of disk	s
Time to probe hash table	s
Size of the network packet	bytes
Network bandwidth	Mb/s
Size of the exchange producer cache	tuples
Size of the exchange consumer cache	tuples
Time to insert into hash table	tuples

Table 2: Tables from TPC-H.

Name	Short Name	Cardinality	Tuple Size
Part	P	200,000	159
PartSupp	PS	800,000	144
Orders	O	1,500,000	104
LineItem	L	6,000,000	112
Supplier	S	10,000	159

Table 3: Parameter settings of PSO and GA algorithm.

Name	Parameter description	Parameter value
PSO	Size of swarm	100
	Self-recognition coefficient (c1)	2
	Social coefficient (c2)	2
	Weight (w)	0.9 0.4
	Max velocity	100
GA	Size of population	100
	Probability of crossover	0.8
	Probability of mutation	0.03
	Scale for mutations	0.1

For the experiment, we used a variety of queries, which makes relations from Table 2, and all queries make use of the TPC-H database. Finally, we totally submitted 4000 queries consecutively to the Grid in two stages; and the computational complexity of all these queries was randomly chosen (about three to five joins). In first stage, we submitted 2000 queries to the Grid and the queries were scheduled by our PSO Query Scheduler algorithm. After the first 2000 queries were all finished, we submitted the second 2000 queries that were allocated with the Genetic Query Scheduler (Di, 2004).

Thus, we compared the performance of PSO algorithm with Genetic algorithm (GA) that has many similarities. The experimental parameter settings of PSO and GA algorithms are described in Table 3.

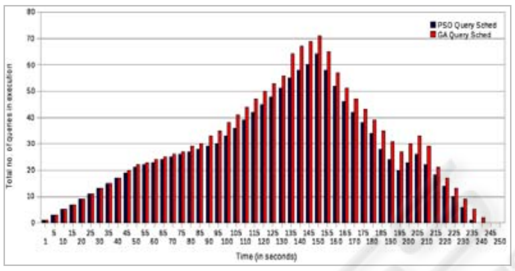

Figure 1: Total no. of queries in execution during time optimization strategy.

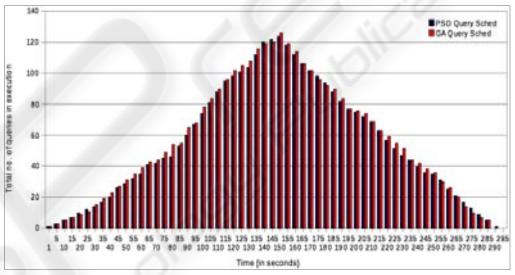

Figure 2: Total no. of queries in execution during cost optimization strategy.

In our experiments, this randomly generated scenario was used for two reasons: (1) it is desirable to obtain data that demonstrate the effectiveness of the approach over a broad range of conditions, (2) it is not clear what characteristics a “typical” heterogeneous computing queries would exhibit. So we conducted experiments for two different optimization strategies:

1. Optimize for time (produce results as early as possible).
2. Optimize for cost (produce results by deadline, but reduce cost)

The number of queries in execution on resources (Y-axis) at different times (X-axis) during the experimentation is shown in Figure 1 and 2 for Time and Cost Optimization strategies respectively.

For example, in bioinformatics databases, this provides consumers the ability to trade-off between time frame and cost that they would like to invest for solving the problem in hand. When the deadline is too tight and results are needed at the earliest possible time, then consumers should be prepared to spend more money. This can be shown in Figure 3, the amount of budget consumed by Cost Optimization strategies.

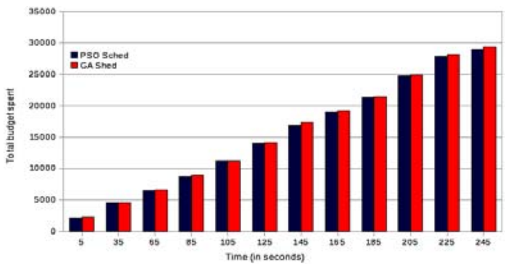

Figure 3: The total amount spent during cost optimization strategy.

6 CONCLUSIONS

Grid computing technologies are enabling the creation of virtual enterprises for sharing distributed resources and changing the way we compute, communicate, and interact with database systems and people. Current distributed database applications operating in heterogeneous settings, like computational Grids, tend to run queries with a sub optimal degree of partitioned parallelism, with negative consequences for performance when the queries are computation and data intensive. On the fly creation of Internet-scale virtual computing environments is becoming more of a reality than dream. Hence, the system managing resources in this complex environment need to be smart, adaptable to changes in the environment and user requirements. At the same time, they need to provide a scalable, controllable, measurable, and understandable policy for management resources. The main contribution of this work is the proposal of PSO resource scheduler that allows a near optimal degree of partitioned parallelism so as to complete the queries in a minimum time as well as utilizing the resources in a computational economy approach and compare it with genetic algorithm under the same condition. To the best of our knowledge, this is the first such proposal.

From the simulated experiment, the results demonstrate that PSO algorithm can get better effect for a large scale optimization problem. Nowadays we are working with more complex scenarios, such as scenarios with changing levels of contention. Finally, another research direction is to create different heuristic based algorithms for problems arising in grid computing.

ACKNOWLEDGEMENTS

The first author gratefully acknowledges support from CONACyT through a scholarship to pursue studies at CIC-IPN’s Computer Science Department. The second author acknowledges support from SIP-IPN through project 20091245 and 20091317. The third author acknowledges support from SIP-IPN through project 20091712.

REFERENCES

Abraham, A., Buyya, R., and Nath, B., 2000. Nature's heuristics for scheduling jobs on computational grids. *In the 8th IEEE Int. Conference on Advance Computing and Communications*. India.

Alpdemir, N., Mukherjee, A., Paton, N.W., Watson, P., Fernandes, A.A.A., Gounaris, A., and Smith, J., 2003. Service-based distributed querying on the grid. *In Proc. Of ICSOC, pp. 467-482*.

Amir, Y., Awebuch, B., Barak, A., Borgstrom, S., Keren, A., 2000. An opportunity cost approach for job assignment in a scalable computing cluster. *IEEE Transactions on Parallel and Distributed Systems 11(7):760-768*.

Braun, T.D., Siegel, H.J., Beck, N., Bölöni, L.L., Maheswaran, M., Reuther, A.I., Robertson, J.P., Theys, M.D., and Yao, B., 2001. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *Journal of Parallel and Distributed Computating, Vol 61, No 6*, p. 810-837.

Buyya, R., Abramson, D, Giddy, J., 2001. A case for economy Grid architecture for service-oriented Grid computing. In proceedings of the International Parallel and Distributed Processing Symposium. *10th IEEE International heterogeneous Computing Workshop, CA. IEEE Computer Society Press*: Los Alamitos, CA.

Buyya, R., Abramson, D, Giddy, J., 2000. An economy driven resource management architecture for global computational power Grids. *PDPTA '00, In proceedings of the 2000 International Conference on Parallel and Distributed Processing Techniques and Applications.* CSREA Press.

Chrétienne, P., 1992. Task scheduling with interprocessor communication delays. *European Journal of Operational Research,* 57:348-354.

Clerc, m., Kennedy, J., 2002. The particle swarm-explosion, stability and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation* 6(1) 58-73 .

DeWitt, D.J., Gerber, R., Graefe, G., Heytens, M., Kumar, K., and Muralikrishna, M., 1986. GAMMA- A high performance dataflow database machine. *In Proc. Of the 12th VLDB Conf.*, pp. 228-237.

Di Martino, V., and Mililotti, M., 2004. Sub optimal scheduling in a grid using genetic algorithms. *In Parallel Computing, Vol. 30*, p. 553-565 .

Epstein, R., Stonebraker, M., and Wong, E., 1978. Distributed query processing in a relational database system. *In Proc. Of the ACM SIGMOD Conf.*, pp. 169-180.

Epstein, R., Stonebraker, M., and Wong, E., Distributed query processing in a relational data base system, 1978. *In proc. Of the ACM SIGMOD Conf.*, pp. 169-180.

Foster, I., and Kesselman, C., 1998. *The Grid – Blueprint for a New Computing Infrastructure.* Morgan Kaufmann Publishers.

Freund, R.F., 1994. The challenges of heterogeneous computing. *Parallel Systems Fair at the 8th International Parallel Processing Symp. IEEE Computer Society*, Cancun, Mexico, pp. 84-91.

Garofalakis, M., and Ioannidis, Y., 1997. Parallel query scheduling and optimization with time- and space-share resources. *In Proc. Of VLDB*, pp. 296-305.

Gounaris, A., Sakellariou, R., Paton, N.W., Fernandes, A.A.A., 2006. A novel approach to resource scheduling for parallel query processing on computational grids. *Distrib. Parallel Databases* 19(2-3),87-106.

Heiser, G., Lam, F., Russell, S., 1998. Resource managment in the Mungi single-address-space operating system. *In proceedings of Australasian Computer Science Conference*, Perth, Australia, 4-6. Springer.

Ioannidis Y., 1996. Query Optimization. *ACM Computing Surveys, vol. 28, no. 1*.

Kossmann, D., 2000. The State of the art in distributed query processing. *ACM Computing Surveys, vol. 32, no. 4*, pp. 422-469,

Liu, D.T., Franklin, M., Parekh, D., 2003. GridDB: A relational interface for grid. *In ACM SIGMOD, ACM Press*, pp. 660-660.

Mackert, L.F., Lohman, G.M., 1986. R* optimizer validation and performance evaluation for distributed queries. *In Proc. Of the 12th VLDB Conf.* pp. 149-159.

Mayr, T., Bonnet, P., Gehrke, J., Seshadri, P., 2003. Leveraging non-uniform resources for parallel query processing. *In 3rd IEEE CCGrid*.

Meijer, M., 2004. *Scheduling parallel processes using Genetic Algorithms.* Master thesis. Universitat van Amsterdam, Faculteit der Natuurwetenschappen, Wiskunde en Informatica.

Narayanan, S., Catalyurek, U., Kurc, T., Zhang, X., Saltz, J., 2003. Applying database support for large scale data driven science in distributed environments. *In Proc. Of GRID.*

Rahm, E., Marek, R., 1995. Dynamic multi-resource load balancing in parallel database systems. *In 21th VLDB, Conf.* pp. 395-406.

Sampaio, S., Paton, N.W., Smith, J., Watson, P., 2002. Validated cost models for parallel OQL query processing. *In Proc. Of OOIS,* pp. 60-75.

Shi, Y., Eberhart, R.C., 1998. A modified particle swarm optimizer. *In Procedings of the IEEE Congress on Evolutionary Computation*, CEC, Piscataway, NJ. 69-73.

Shroff, P., Watson, D. W., Flann, N.S., and Freund, R.F., 1996. Genetic simulated anneling for scheduling data-dependent tasks in heterogeneous environments. *In Proc. Heterogeneous Computing Workshop. IEEE Computer Society*, Honolulu, HI, pp. 98-104.

Singh, H., Youssef, A., 1996. Mapping and scheduling heterogeneous task graphs using genetic algorithms. *In Proc. Heterogeneous Computing Workshop. IEEE Computer Society*, Honolulu, HI, pp. 86-97.

Smith, J., Gounaris, A., Watson P., Paton N.W. , Fernandes, and Sakellariou, 2003. Distributed query processing on the Grid. *International Journal of Hight Performance Computing Applications, vol. 17, no. 4*, pp. 353.367.

Stonebraker, M., Devine, R., Kornacker, M., Litwin, W., Pfeffer, A., Sah A., Staelin, C., 1994. An economic paradigm for query processing and data migration in Mariposa. *Proceedings 3rd International Conference on Parallel and Distributed Information Systems*, Austin, TX, 28-30. IEEE Computer Society Press: Los Alamitos, CA.

Wilschut, A.N., Flokstra, J., Apers, P., 1992. Parallelism in a main-memory DBMS: The performance of PRISMA/DB. *In Proceedings of the 18th VLDB Conf.*

Zomaya, A.Y., Teh, Y.H., 2001. Observations on Using Genetic Algorithms for Dynamic Load-Balancing. *IEEE Transactions On Parallel and Distributed Systems, Vol 12*, No 9.