The Australian National University

TR-CS-95-05

On Balancing Workload in a Highly Mobile Environment

Jeffrey X. Yu, Kian-Lee Tan and Xun Qu

August 1995

Joint Computer Science Technical Report Series

Department of Computer Science
Faculty of Engineering and Information Technology

Computer Sciences Laboratory
Research School of Information Sciences and Engineering

This technical report series is published jointly by the Department of Computer Science, Faculty of Engineering and Information Technology, and the Computer Sciences Laboratory, Research School of Information Sciences and Engineering, The Australian National University.

Please direct correspondence regarding this series to:

Technical Reports
Department of Computer Science
Faculty of Engineering and Information Technology
The Australian National University
Canberra ACT 0200
Australia

or send email to:

`Technical.Reports@cs.anu.edu.au`

A list of technical reports, including some abstracts and copies of some full reports may be found at:

`http://cs.anu.edu.au/techreports/`

Recent reports in this series:

TR-CS-95-04 Department of Computer Science. *Annual report 1994*. August 1995.

TR-CS-95-03 Douglas R. Sweet and Richard P. Brent. *Error analysis of a partial pivoting method for structured matrices*. June 1995.

TR-CS-95-02 Jeffrey X. Yu and Kian-Lee Tan. *Scheduling issues in partitioned temporal join*. May 1995.

TR-CS-95-01 Craig Eldershaw and Richard P. Brent. *Factorization of large integers on some vector and parallel computers*. January 1995.

TR-CS-94-10 John N. Zigman and Peiyi Tang. *Implementing global address space in distributed local memories*. October 1994.

TR-CS-94-09 Nianshu Gao and Peiyi Tang. *Vectorization using reversible data dependencies*. October 1994.

On Balancing Workload in a Highly Mobile Environment

Jeffrey Xu Yu
Department of Computer Science
Australian National University
Canberra, ACT 0200, Australia
email: yu@cs.anu.edu.au

Kian-Lee Tan
Dept. of Information Systems & Computer Science
National University of Singapore
Lower Kent Ridge, Singapore 0511
email: tankl@iscs.nus.sg

Xun Qu
Computer Sciences Laboratory
Australian National University
Canberra, ACT 0200, Australia
email: quxun@cslab.anu.edu.au

Abstract

Traditionally, in a distributed database system, it has been assumed that most queries can be processed with the data that reside at the site where the query is submitted. However, this concept of localization is challenged by new technologies that facilitate mobility of users. In a highly mobile environment, queries issued by a mobile user are more likely to require remote access. As a result, existing algorithms may no longer provide the desirable performance. In this paper, we propose and study several algorithms for improving the throughput of the system in such a context. Based on a simulation study, our results show that unless the communication is low, strategies that balance the workload based on the machine load perform best.

1. Introduction

Traditionally, a distributed database comprises a collection of multiple, logically interrelated databases that are distributed over a computer network [Ozsu91]. To facilitate distribution transparency and accesses to the data, the DBMS is managed by a software system – the distributed DBMS. Researches on and systems built for distributed databases were mainly in this context since late 1970s. However, recent advances in computer hardware and communications technologies have led to the integration of powerful mobile/portable computers into existing networks via a cellular connection or a wireless LAN[1].

[1] For simplicity, in this paper, we shall restrict to wireless LAN.

As argued in [Imie93, Alon93b], the introduction of mobile computers has led to new applications and opened new avenues for research. In a mobile environment, a *database host*, a server in the static network, has to support a large number of query requests that come from two sources – local terminals and mobile hosts. These two sources differ in the followings:

(1) Local terminals are connected directly to the database hosts on the wired network. Mobile hosts are the portable computers that submit queries to a nearby database host on the wired network via a mobile support station. Since this nearby database host may not be the database host that store/produce the results of the queries, we shall distinguish it by calling it the *query host*.

(2) Users on local terminals access local data more frequently than remote data. On the contrary, users on mobile hosts possibly access data that are remote to its query host.

Traditionally, distributed databases and techniques are designed based on the concept of localization, i.e. data are stored at the site where they are more frequently accessed, and queries posed at a site generally access data local to the site. However, because of the different roles that a database host may play with respect to local terminals and mobile hosts, this assumption may no longer hold in a mobile environment where no localization can be predefined. More precisely, if queries directed to a database host are from local terminals, then it is most likely that the rule-of-thumb that 80% of these queries access local data holds. However, if a large number of queries are from mobile hosts, we can expect the percentage of queries accessing remote data to increase significantly. In fact, if all query hosts need to process queries coming from mobile hosts by accessing remote data, one may even expect a centralized approach to outperform a distributed approach in a mobile environment in terms of the amount of data transferred. In this paper, we investigate the adaptability of the load-balancing techniques in a highly mobile environment. In the following, we summarize some of our major concerns.

(1) As mentioned above, the communication traffic on the wired network in a mobile environment can be much higher than that in a distributed DBMS. The efforts done on distributed DBMSs to minimize amount of data transferred, such as data fragmentation, may lead to negative effects. How effective are traditional techniques needs to be reevaluated.

(2) There are two different groups of users. A group of users works on local terminals, and a group of users works on mobile hosts. On some database hosts, it might be desirable that the priority of queries from local terminals be ranked higher than those from mobile hosts. However, on the other hand, at the expenses of queries from local terminals, it might be requested to process queries from mobile hosts as soon as possible, in order to shorten the response time for queries from mobile hosts. How to prioritize the different types of queries is an issue of concern.

(3) In a mobile environment, the response time is important to users working on mobile hosts. Mobile hosts are movable, and are most likely to request that the results be sent back by distributed DBMSs as soon as possible. A shorter response time will reduce the possibility that a mobile host moves to another region during the transmission of the result. It can therefore reduce the possibility of forwarding the results and/or retransmission of results, and hence reduce the amount of data being transmitted on the wired network. However, in order to shorten the response time, it needs either to process queries in parallel or to find a database host on which a query processing can start quickly. In either case, the communication traffic can be high. A set of good processing strategies for mobile environment should be developed.

(4) The total throughput and the balance of total workload on all database hosts and the wired network are important. Inefficient techniques might make some database hosts and/or links between database hosts to become bottleneck while the other database hosts and links are idle in a mobile environment. Some load-balancing mechanisms should therefore be designed.

All the above-mentioned points indicate the importance of effectively managing the communication traffic on the wired network in a mobile environment. All of the problems are caused by the fact that the concept of localization fades. This is particularly bad for a highly mobile environment. First, patterns of query requests from mobile hosts are hardly predetermined compared with queries from local terminals. Second, the number of queries from mobile hosts on a database is not predictable.

The remainder of the paper is organized as follows. In Section 2, we briefly mention some current works on mobile computing. Section 3 describes our architectural assumptions and our system model. Section 4 presents possible strategies which are adoptable in a mobile environment. In Section 5, our simulation experiments and results will be given. Finally, Section 6 summarizes our results and discusses our plans for future works.

2. Previous Work

While mobile computing has drawn a large number of research activities in recent years, to our current knowledge, there is no reported study on traffic management on the static network and workload management on the database hosts. In this section, we summarize some of the existing work that dealt with the impact of mobility on distributed computations. Four issues were raised in [Badr93]. First, the locality of the mobile hosts must be determined. Several schemes have been developed to search for the current position of a mobile host [Awer91, Badr95, Ioan93].

Second, the low communication bandwidth of wireless medium may limit the performance of the systems. Several studies addressed this issue by minimizing communication in the wireless

network. In [Huan94], the problem is tackled by replicating data across the mobile hosts as well as the database hosts. Lai and et. al. approached the problem by designing an adaptive protocol [Lai95] that batches queries to reduce the number of messages and/or the amount of data being transferred.

The third issue concerns the physical connectivity of the network which changes as mobile hosts move from cell to cell. To resolve this problem, it was proposed that distributed algorithms in mobile environment should be designed such that the main bulk of the communication and computation costs is borne by the static portion of the network [Badr95]. In other words, the mobile host acts like a "terminal" – submits queries to the server and receives results from the server.

Finally, because of its portability, mobile hosts have constraints such as limited battery life. To conserve energy, techniques for increasing the effective battery life are desirable. Alonso and Ganguly have examined the criterion for optimizing queries based on energy efficency in a mobile environment [Alon93a]. In [Imie94], broadcast data are indexed to conserve energy. Another direction is to operate the mobile hosts in "doze" mode.

The work reported in [Acha93] exploits multicasting as follows: copies of a message is sent to many mobile host stations which will broadcast the message to mobile hosts. This technique, while effective for small messages, is not practical in our context since data and/or results of a query are usually large and will lead to bottleneck in the static network.

Load-balancing has been a very active area of research since early 1980s [Brya81, Care86, Eage85, Hac86, Livn82, Lu90, Ni81, Ni82, Yu86]. While early work tries to balance the number of tasks at the sites [Livn82, Ni81, Ni82], more recent work have employed more complex metrics that are based on resource demands, and the estimated response time [Care86, Lu90]. However, most of these work are studied under the assumption of localization. The impact of mobility was never an issue at that time.

3. The Mobile Environment

We adopt the system model specified in [Badr95]. A "mobile host" implies that it is able to move while retaining its network connections. The infrastructure machines that communicate directly with the mobile hosts are called "mobile support stations". A "cell" is a logical or geographical coverage area under a mobile support station. All mobile hosts that have identified themselves with a particular mobile support station are considered to be "local" to that mobile support station. A mobile host can directly communicate with a mobile support station if the mobile host is physically located within the cell serviced by the mobile support station. At any given instant of time, a mobile host may logically belong to only one cell; its current cell defines a mobile host's location.

The system model consists of two distinct sets of entities: a large number of mobile hosts and relatively fewer, but more powerful, "fixed hosts". The overall network architecture thus consists of a "wired network" of database hosts that connect the otherwise isolated and low-bandwidth "wireless networks". Each wireless network comprises a mobile support station and the mobile hosts local to its cell.

Unlike the system model in [Badr95], in our system model, a set of fixed hosts is restricted to be a set of database hosts along with a set of "local terminals" connected to the database hosts on the wired network. A "database host" manages a large amount of data in the form of either relations or subsets of relations called fragments. If a centralized approach is adopted, one database host serves as both a data server and a query server, while the remaining database hosts are query servers. In this case, all data are managed in the form of relations on the data sever. If a distributed approach is adopted, all database hosts work as data servers and query servers. In this case, data are distributed across all database hosts and managed in the form of relations or fragments. A database host is responsible for processing query requests translated from multiple mobile support stations as well as from local terminals. Also, the wired network consists of a wide area network (WAN) and local area networks (LAN).

Queries from local terminals are translated to the database host to which the local terminals are connected. Queries from mobile hosts are translated to a nearby database host, called the *query host*, through a mobile support station. The query host will then act as a result site for such query requests and process such query requests. The query host is also responsible to arrange the transmission of results to the local terminals or the mobile hosts through the mobile support station. From now on, a "local query" means a query from a local terminal, and a "mobile query" means a query from a mobile host.

In this paper, we attempt to investigate the impact of mobile computing on the wired network, in particular the wide area network that connects the database hosts, in a mobile environment. For simplicity of the performance study, we have made the the following assumptions. First, all processing will be done on database hosts. This is similar to the approach proposed in [Badr95]. Under this assumption, we can ignore the wireless network in our study. Second, the architecture of the network among database hosts is a point-to-point wide area network. Third, all mobile support stations and local terminals are assumed to connect to a database host in a very fast local area network. The cost for data transmission both between a local terminal and a database host, and between a mobile support station and a database host is ignored. Fourth, there is no caching. It is reasonable because that the large amount of queries accessing remote data can turn off caching function. Last, queries are read-only queries including single join queries and selection queries.

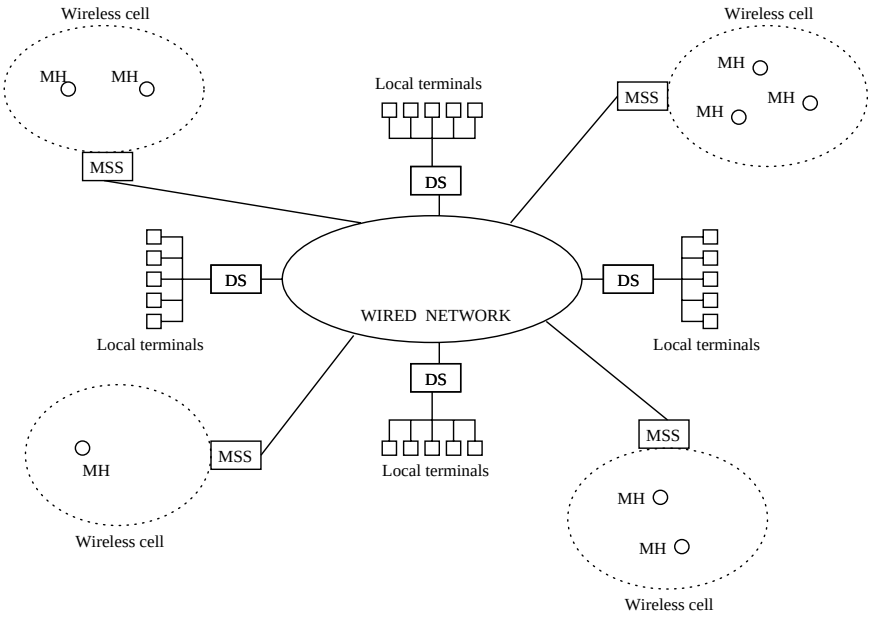

Figure 1: The mobile environment.

4. Query Processing Strategies

In this section, we present the processing strategies that are the focus of this paper. As mentioned earlier, the focus is on the wide area network part of the wired network in a mobile environment.

4.1. Centralized Approach

If we adopt a centralized approach in a mobile environment, the system comprises a database host which serves as both a data server and a query server. The rest of database hosts are query servers. The database host which serves as a data server has much more CPU power than the other database hosts which work as query servers. All queries arriving on the database hosts except the data server can be treated as mobile queries, and so they all require remote access. All queries arriving on the database host which works as a data server are local queries. In this architecture, for each database host, there is no difference between mobile queries and local queries. For a same query, the total costs on a database host for processing the query if it comes from a mobile source is the same as that which comes from a local terminal. Such an architecture is not favourable to those who work on local terminals and need to keep data locally. However, it might be beneficial for processing queries from mobile hosts because there are no such data transmission needed to transfer data from multiple sites to a query processing site.

4.2. Distributed Approach

If we adopt a distributed approach in a mobile environment, we assume that all database hosts are both a data server and a query server. Each database host has less CPU power than that of the data server used in a centralized system. The rule remains unchanged that 80% of local queries access local data and 20% of local queries need to access remote data. The number and the patterns of mobile queries can hardly be predetermined. We assume that most of them need to access remote data. Under a distributed approach, there are two issues that have to be addressed. Since there are two different groups of users working in a mobile environment, the order in which queries are executed may affect performance. We studied three prioritization strategies:

- **Mobile queries are assigned higher priorities than local queries.** In this strategy, mobile queries are assigned higher priorities than local queries. This is to reduce response time for mobile queries. However, it might lower the total throughput because mobile queries need remote accesses. In addition, local users who work on local terminals may be penalized and suffer from long response time, even though they only need to access local data.

- **Local queries are prefered over mobile queries.** In order to protect local users, who work on local terminals and mostly access local data, from a possible large number of mobile queries arriving at database hosts, it may be desirable that local queries be executed first. This strategy assigned higher priorities to local queries. As such, it might improve the total throughput but at the expense of long response time of mobile queries.

- **Local queries and mobile queries are treated equally.** In some situation, it may not be possible to specify a priority between mobile queries and local queries. In this case, this policy of "equality" will be adopted, i.e. all queries enjoyed the same priority and is served in a first-come-first-serve basis.

In general, a distributed DBMS system attempts to execute queries to minimize the total amount of data transmission. For a selection query, the selection will be executed on the sites where a part of the required relation resides. The results will be sent to the issued site. For a simple join query, the query optimizer will attempt to find an execution site on which the total amount of data transmission is minimum. Then all of the data involved in the join query will be sent to the execution site. Finally, the result of the join query will be sent back to the result site from the execution site if it is different from the result site. However, in a mobile environment, some database hosts may be overloaded and the response time for both local queries and mobile queries will be high. This will be critical especially for situations with a large number of mobile queries that need remote access. We study the following three load balancing strategies that can be exploited:

- **Minimize the amount of data being transferred:** This strategy minimizes the amount of data to be transferred, and is a commonly used strategy in many distributed DBMSs.

- **Minimize the workload of database host:** This strategy attempts to process mobile queries on a database host whose workload is minimum. It essentially attempts to balance the processing load at the database hosts.

- **Minimize the workload of network:** In a point-to-point network, some links are overloaded but some links are idle. This strategy attempts to execute mobile queries on a database host to which the links connected are lowly loaded compared with others. This strategy is likely to increase the total amount of data transmission, but it is expected to improve the response time and the total throughput.

5. Experiments and Results

In order to study the performance of the proposed query processing algorithms in a mobile environment, we constructed a simulation model. This section describes the simulation model and report the results of the study.

5.1. The Simulation Model

The structure of our simulation model is based on the system model that was shown in Section 3. Here, we will describe how the model captures the database, workload and various physical resources for processing queries in a mobile environment. The simulation model has been implemented using the C++SIM simulation package [Proj94].

Figure 2(a) shows our model of a mobile environment. The system consists of a set of database hosts interconnected by a network. The network in this paper is assumed to be a point-to-point architecture. In the figure, we only show three database hosts. The number of database hosts in our simulation is fixed to 8. Queries are issued on either a mobile host or a local terminal and arrive at the query queue of the respective database host. The network is a point-to-point network. Hence, all data transmission between two database hosts is managed by a network queue. Queries that require remote access are sent out via the network queue.

In Figure 2(b), we show the "internals" of a database host. A database host will execute queries kept in the query queue based on the prioritization strategy employed. When a query only needs to access local data, this query will be executed at the database host. When a query needs remote data access, the database host will send the corresponding requests to other database hosts, and keep this query request in a separate queue until all the result is available. When a request from another database host is picked up from the query queue, the

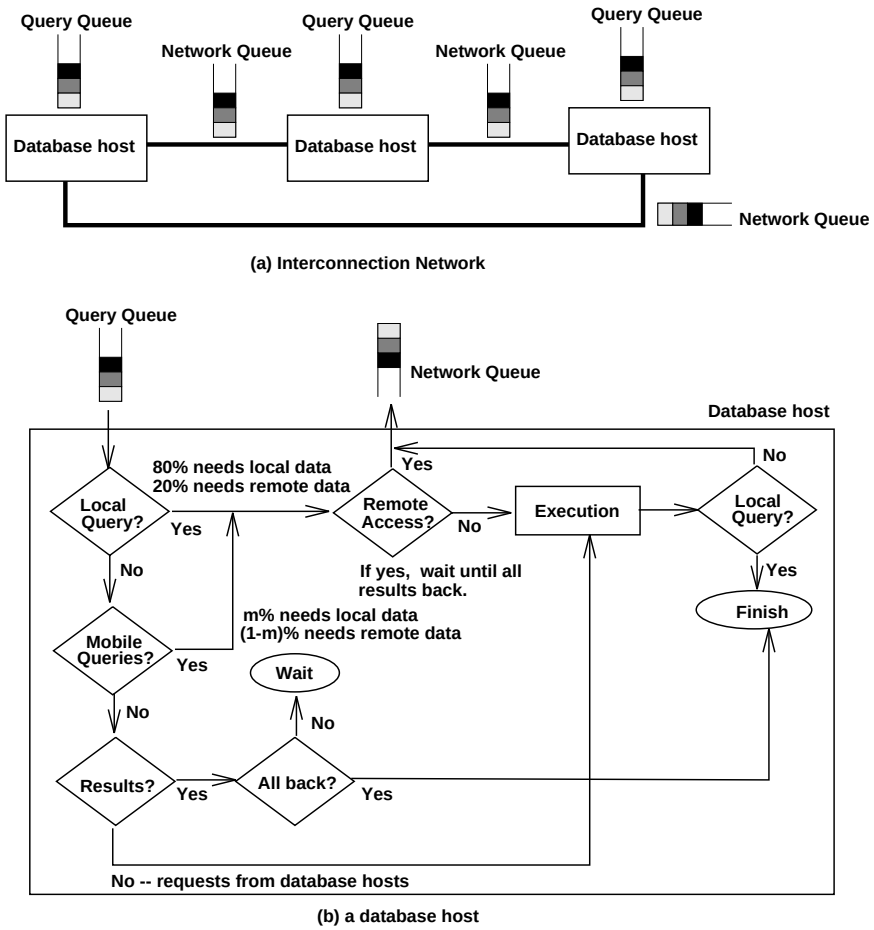

Figure 2: The simulation model.

database host will execute the request (according to the prioritization strategy), and then sent the results back to the requesting database host. No buffer management on database hosts is used in our simulation. A query is executed in a fashion that all CPU and disk resources are available to the query. As a mater of fact, this may end up inaccurate results, but we believe that the current assumptions we have will not change the result dramatically.

When a selection query needs to access remote data, the following operations are performed: the database host where the query is submitted sends the requests to the respective database hosts that contains the data fragments; these database hosts process the selection at their own site and send the result back.

For join queries that involve remote data access, all remote data will be transmitted to a preselected site where the join is to be processed. If the preselected site is different from the issuing database host, the results of such join has to be sent back to the issuing database host. In our study, we employ the merge-join algorithm for join processing.

Note that the three load balancing strategies proposed in this paper don't have any impact on selection queries when remote accesses are needed. More precisely, in our simulation, we generate a random number for each query that needs remote accesses, for example, m. Then we randomly choose m database hosts. The selected database hosts will be affected. For a selection query that needs remote accesses, the m database hosts will send the data back to the issuing site. The load balancing strategies, however, will have significant impact on join queries because they are used to choose a site on which the join will be processed. For a join query that needs remote access, we choose a database host to be the site on which the join is processed by using one of the load balancing strategies. The $m-1$ database hosts send data to the selected database host, which will process the join, and send the result back to the issuing site.

We ran the simulator for 4000 queries in total, and collected the total amount of data transmitted over the network (in terms of number of pages) and the average of response time. The total throughput is a reciprocal of the average of response time. The main parameters used are shown in Table 1.

Parameter	**Description**	**Default Value**
DBarch	Centralized or distributed DBMS	distributed
HostNum	Number of database hosts	8
MachineCPU	Instruction rate of CPU	15 MIPS
PageSize	Page size	4K Bytes
PerPageInst	Average number of instructions per page	10000
DiskTime	Disk access time per page	20 milliseconds
NetworkBandwidth	Network bandwidth	1000K bits per second
MobileDistributed	% of mobile queries access remote data	75%
LocalDistributed	% of local queries access remote data	20%
DataSize	The size of pages requested per site	100K pages
JoinSelectivity	Join selectivity	50%
SelectionSelectivity	Selection selectivity	50%
JoinOpFactor	% of simple join queries	50%
ProceecedJobs	Number of queries being processed	4000
LocalUser	Number of local users	50
MobileUser	Number of mobile users	0 - 150

Table 1: Parameters and their Meaning

For a distributed DBMS, we use a 15 MIPS CPU and a disk driver which needs 20 milliseconds to access a page. For a centralized DBMS, we use a 10 times faster CPU and a 10 times faster disk driver for the data server. The number of local users who are working at local terminals is fixed to be 50. We vary the number of mobile users from 0 to 150. A local or mobile user can send another query when his/her previous queries have been processed. The "MobileDistributed" and "LocalDistributed" are the percentage of mobile and local queries

which need remote data access, respectively. In all the following simulations, we fix the "LocalDistributed" and "MobileDistributed" to be 20% and 75%, respectively. The "DataSize" specifies the maximum number of pages a query can request at a database host. As default, it is set to be 100 of 4K pages. A query can request data from multiple database hosts. The data size at each host is randomly chosen between one fourth of "DataSize", 25 of 4K pages, and "DataSize", 100 of 4K pages. The simulation will stop when the total number of queries exceeds the "ProcessedJobs". No replication are considered in our current simulations. In our study, half of the queries are join queries, and the other half of the queries are selection queries.

We shall also use the following abbreviations for subsequent discussions:

- **LD** indicates those local queries that need remote accesses.
- **MD** indicates those mobile queries that need remote accesses.

Besides the centralized approach, we also studied 9 strategies for the distributed approach. They are the combinations of the three priorization strategies and the three data load balancing strategies as mentioned in Section 4.2. For the priorization strategies, we have the following notations:

- **M** represents that mobile queries are given precedence over local queries. In our study, we will process 10 mobile queries before 1 local query is processed (unless there is no mobile queries in the queue).
- **L** indicates that local queries have higher priority over mobile queries. Similarly, we process 10 local queries before 1 mobile query if any local queries are in the queue.
- **E** means there are no differences between local and mobile queries. In this case, the strategy of First-Come-First-Server is applied.

For the three load balancing strategies, we denote them as:

- **Data** indicates the strategy of minimizing the amount of data being transferred.
- **Mac** indicates the strategy of minimizing the workload of database hosts
- **Net** indicates the strategy of minimizing the workload of network.

In our study, we also model two different scenarios on query distributions: uniform and non-uniform query distribution. For uniform query distribution, each query is equally likely to access data from any of the database hosts. Hence, all database hosts will receive the similar number of requests from other database hosts. There is no such a bias towards any database host. For the non-uniform query distribution, a large number of queries will access data from a particular site. Hence, a particular database host will always receive a request whenever any

database host requests a remote access from other database hosts. It simulates the situation where a database host in a distributed DBMS behaves like the data server in a centralized DBMS system. The following notations are used for the various algorithms:

- **CDB.** Centralized DBMS approach.
- **DDB-X-Y.** Distributed DBMS approach with uniform query distribution. Here, the prioritization strategy **X** is used, and the load balancing strategy **Y** is employed. For example, **DDB-E-Net** denotes the distributed strategy that gives no preference to all queries, mobile or local, and attempts to improve the system performance by minimizing the workload of network.
- **FDDB-X-Y.** Distributed DBMS approach with non-uniform query distribution. As before, **X** refers to the prioritization strategy used, and **Y** refers to the load balancing strategy employed. For example,

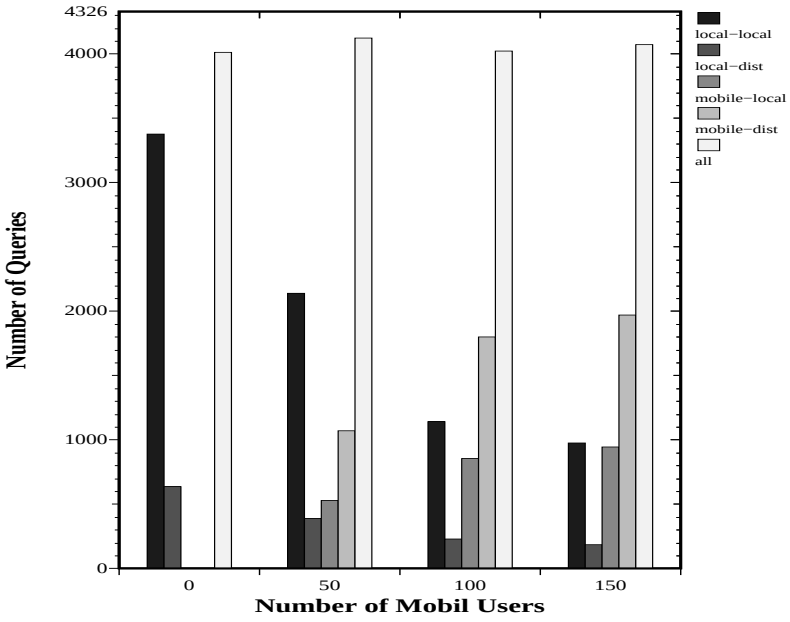

Figure 3: The queries being processed when network-bandwidth = 1000K bits/sec.

5.2. Experiment 1: Uniform Query Distribution

In this experiment, we studied the algorithms when the query distribution is uniform. Figure 3 shows the number of queries being executed for the case of "E-Data". For the other cases,

they show the similar patterns. When the number of mobile users is 0, about 700 local queries out of 4000 need remote access. However, by the time the number of mobile users increases to 150, a half of the total queries needed remote accesses. As can be seen, the total number of distributed queries is moderate in our simulation. Only a small portion of local queries needs remote accesses. The number drops from 700 to 200. The number of mobile queries that need remote accesses increases from 1000, when the number of mobile users is 50, to 2000 when the number of mobile users is 150.

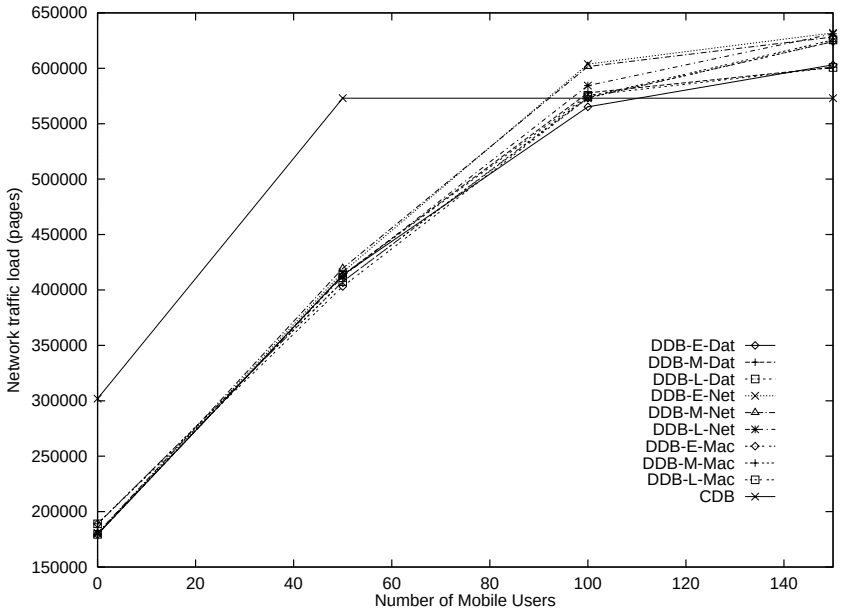

Figure 4: Amount of data transmitted when network-bandwidth = 1000K bits/sec.

Figure 4 shows the total amount of data transmitted and Figure 5 shows the average response time when the network bandwidth is 1000K bits per second. From the graphs, we made several interesting observations. First, we compare the performance of the centralized approach with that of the distributed approach. In Figure 4, the amount of data transmitted under a distributed approach exceeds that needed in a centralized DBMS when the number of mobile users is large (> 100). Nevertheless, the average response time in a distributed DBMS is still faster than that in a centralized DBMS. In a centralized DBMS system, the response time for queries that need remote access jumps to 50 seconds when the number of mobile users becomes 50 and remains unchanged since then. As can be seen from Figure 5, the average response time for all queries are less than 20 seconds. It is because that the links connecting the data server in a centralized DBMS is overloaded while the other links are idle in a point-to-point network architecture.

Second, among all the distributed strategies, those that are based on "Mac" consistently outperforms those that are based on "Data" and 'Net" (see Figures 4 and 5). It implies that, when network bandwidth is big enough compared with the machine power, it is more

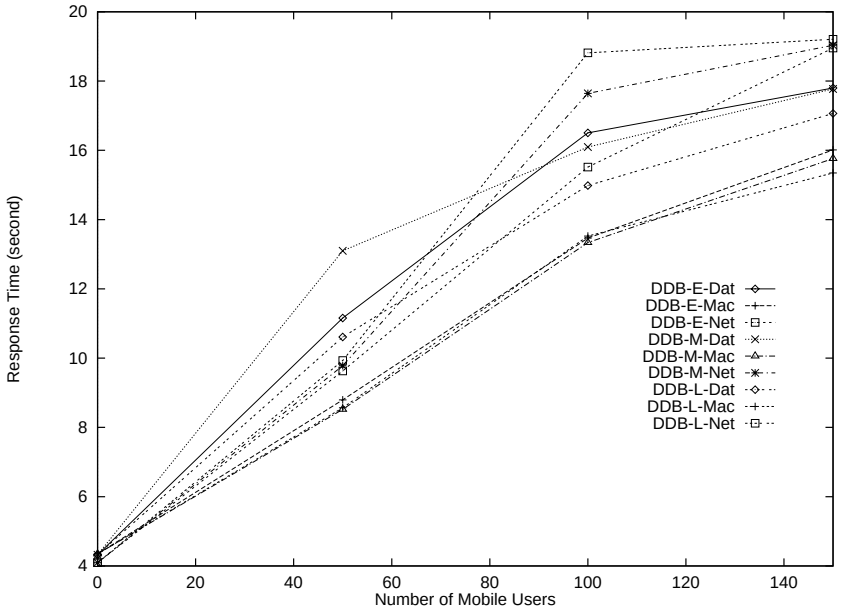

Figure 5: The average response time when network-bandwidth = 1000K bits/sec.

important to balance machine workload rather than data transmission or workload on data link, even though the network may be overloaded. It also indicates, in a point-to-point network architecture, if a machine has less workload, it implies that the links connecting to this machine is also likely to be less loaded.

Third, as seen in Figure 4, as the number of mobile user grows, the total amount of data transmitted for strategies that employ "Net" is always larger than the strategies that employ "Data" or "Mac". In particular, when number of mobile users is 100, the total amount of data transmitted for the "Net" strategies is about 600000 4K pages which is 30000 more than the strategies that used "Data" and "Mac". As a result, the "Net" strategy doesn't achieve the best average response time as can be seen in Figure 5. In Figure 5, it shows that, when number of users is moderate, the "Net" strategies are better than the "Data" strategies. However, when the number of mobile users is big enough, the "Data" strategies can outperform the "Net" strategies. This is because the network itself is overloaded, and there is no room for a system to minimize the workload of the network. Furthermore, for a large number of mobile users, the strategies that employ "Net" might lead to bottleneck at the network.

Figure 6 shows the response time for those queries that need remote access when propritization strategy is "E". Even though a large number of mobile queries is executed in comparison with the number of local queries, as number of mobile users increases, the average response time for "LD" sharply jumps from 14 seconds for the case when there is no mobile user to a maximum of 35 seconds when there are 150 mobile users. It implies that a large number of mobile queries that need remote accesses can squeeze out a small number of local queries that need remote accesses. For queries that only need local accesses, the average time is between

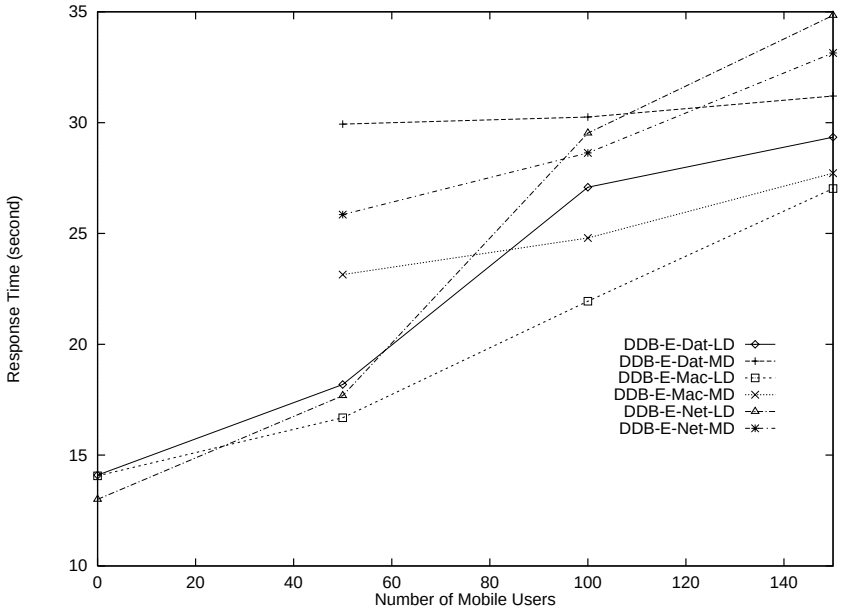

Figure 6: The response time for remote data accesses for "E" prioritization strategies (network-bandwidth = 1000K bits/sec).

2.3 seconds and 3 seconds.

Figure 7 and Figure 8 show the response time for distributed queries when propritization strategies: "L" and "M" are chosen, respectively. In both strategies, the response time for "MD" doesn't change significantly in comparison with the strategy "E". However, these two strategies have impact on local queries. When the number of mobile users is large (150), for the strategies that employed "M", the response time for the fewer local queries is between 35 and 65 seconds. For the strategies that used "L", the response time for local queries are around 15 seconds. It indicates, for a large number of mobile users, the strategies that used "L" can keep the response time of local queries more or less as a constant without having significant impact on mobile queries.

In summary, the results in this section are obtained under the condition that a) there is no such a bias toward any database hosts and b) the network bandwidth is 1000K bits per second. The strategy "Mac" always outperforms the others, namely "Data" and "Net". The strategy "L-Mac" can keep the response time of local queries more or less as a constant while not having impact on mobile queries.

5.3. Experiment 2: Non-uniform Query Distribution

In this experiment, we study the FDDB-based algorithms for the non-uniform query distribution scenario. Figure 10 shows the total number queries being executed for "E-Data" strategy. For the other cases, they show the similar patterns.

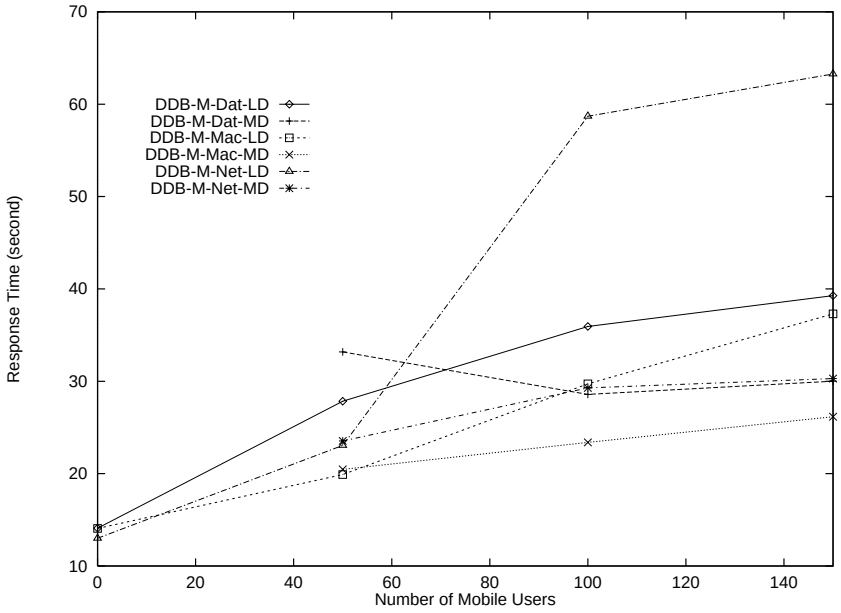

Figure 7: The response time for remote data accesses for "M" prioritization strategies (network-bandwidth = 1000K bits/sec).

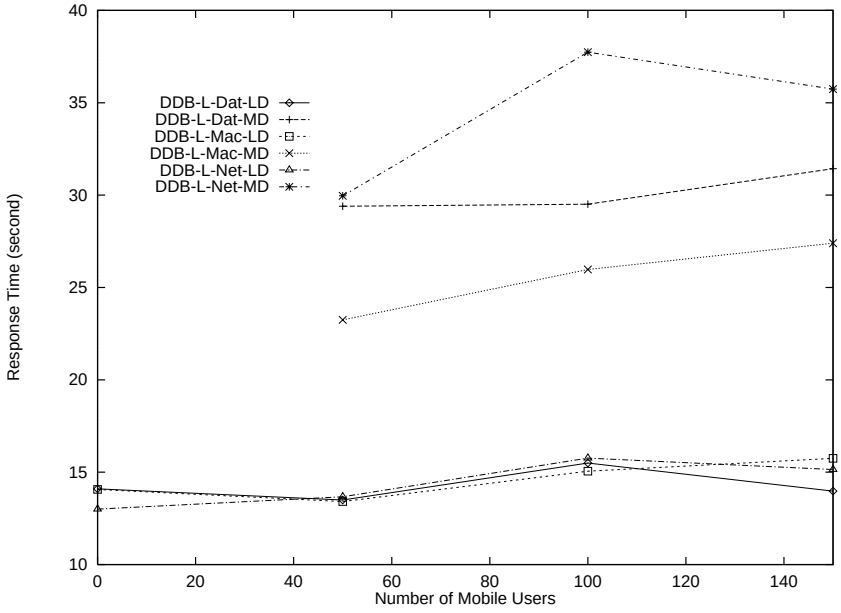

Figure 8: The response time for remote data accesses for "L" prioritization strategies (network-bandwidth = 1000K bits/sec).

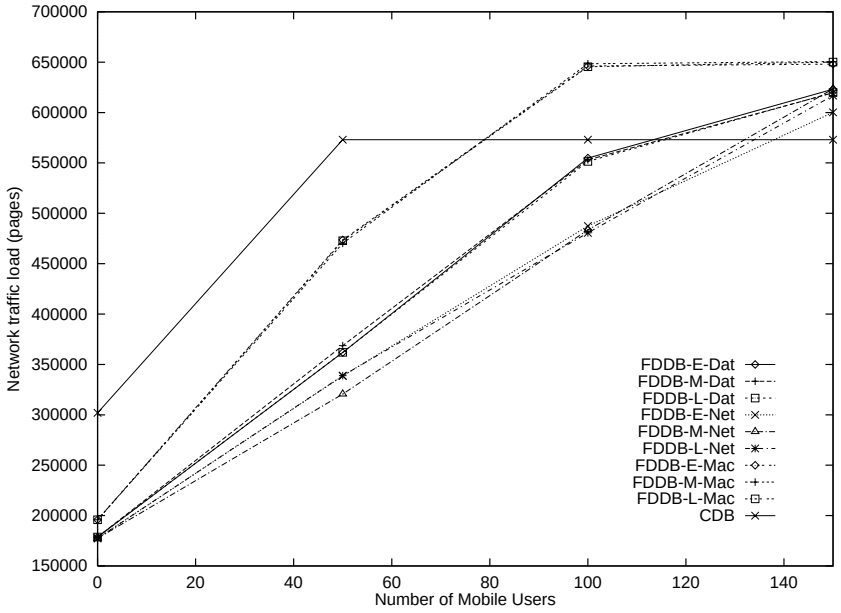

Figure 9: The data transferred when network-bandwidth = 1000K bits/sec.

Figure 9 shows the total data transmission when the network bandwidth is 1000K bits per second. Under non-uniform query distribution scenario, we see a totally different picture from the uniform query distribution scenario. The strategies that are based on "Mac" always send more data over the network than the strategies that are based on "Data". The strategies that are based on "Net" always transmit less data than the other two strategies. When the strategies "Mac" is chosen, the database host, which works as the data server in a centralized DBMS system, is overloaded. The links to the data server is also overloaded. The strategy "Mac" attempts to choose a database host which has less workload for a join query that needs remote accesses. The strategies of "Net" and "Data" might still send data to the overloaded database host but they are on average superior to the "Mac" strategies.

Strategy	Total Data Transferred	Temporary Data	Final Results
"E-Mac"	646179	279848	366331
"E-Data"	554873	242995	311878
"E-Net"	487377	245620	241757

Table 2: Data transmitted when the number of mobile users is 100.

Table 2 shows the amount of data transmission when the number of mobile users is 100 for the "E"-based strategies. In the table, we show the total amount of data being transferred, the temporary data being transferred to a site on which the join is processed and the total data amount for the final results. For the strategy E-Net, 50% of the data transmitted are used for temporary relations. On the contrary, in the other two strategies, 43% are used for sending

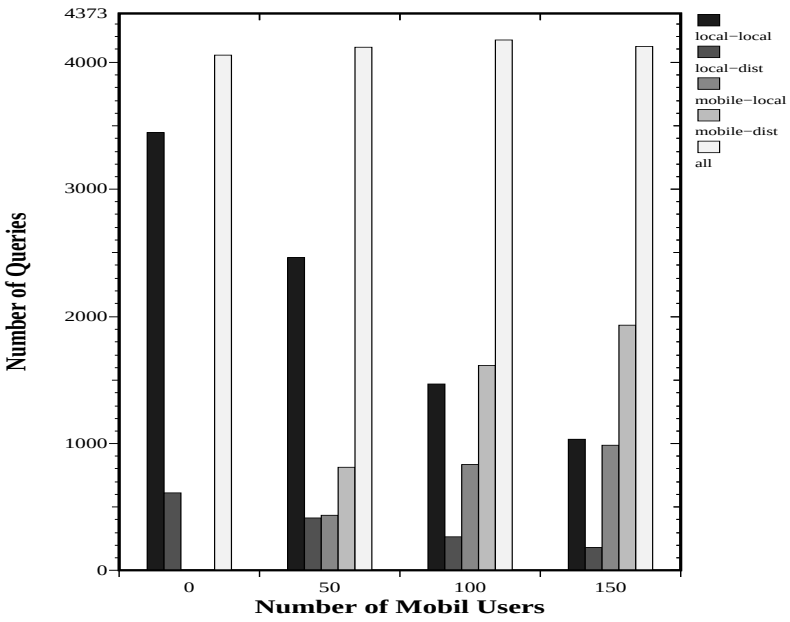

Figure 10: The queries being processed when network-bandwidth = 1000K bits/sec.

temporary relations. From Figure 11, which shows the average response time for all queries, it is clear that the "Net"-based strategies are better than the other two categories of strategies.

Strategy	DistributedJoin	DistributedSelect	LocalJoin	LocalSelect
"E-Mac"	684	1483	916	984
"E-Data"	453	1419	1113	1180
"E-Net"	218	1284	1309	1404

Table 3: Number of queries when the number of mobile users is 100.

This is the case because for strategies based on "Net", the number of local queries being executed are larger (as shown in Table 3). It implies that the network is overloaded and database hosts are working on local queries. However, the strategies based on "Mac" process more distributed queries than the other two strategies.

Figure 12, Figure 13 and Figure 14 show the response time for distributed queries for propritization strategies "E", "M" and "L", respectively.

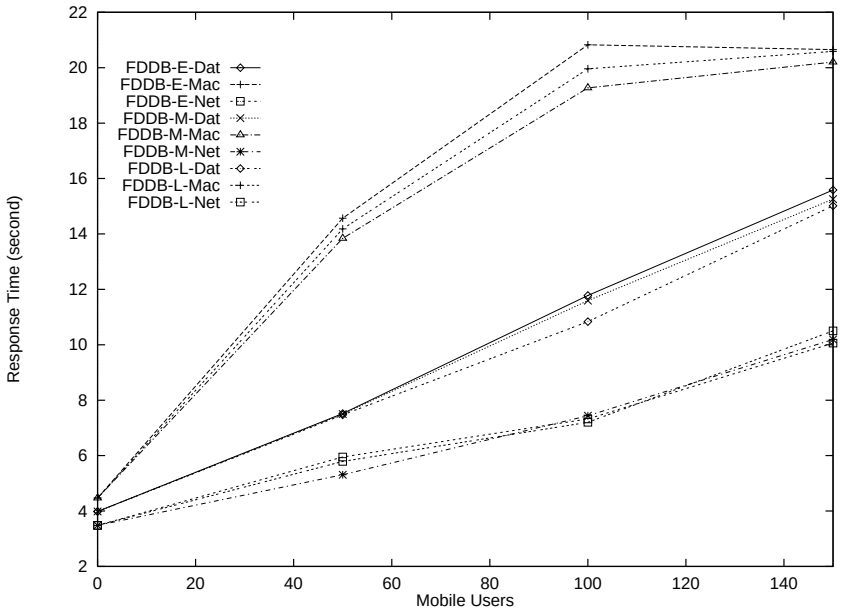

Figure 11: The average response time when network-bandwidth = 1000K bits/sec.

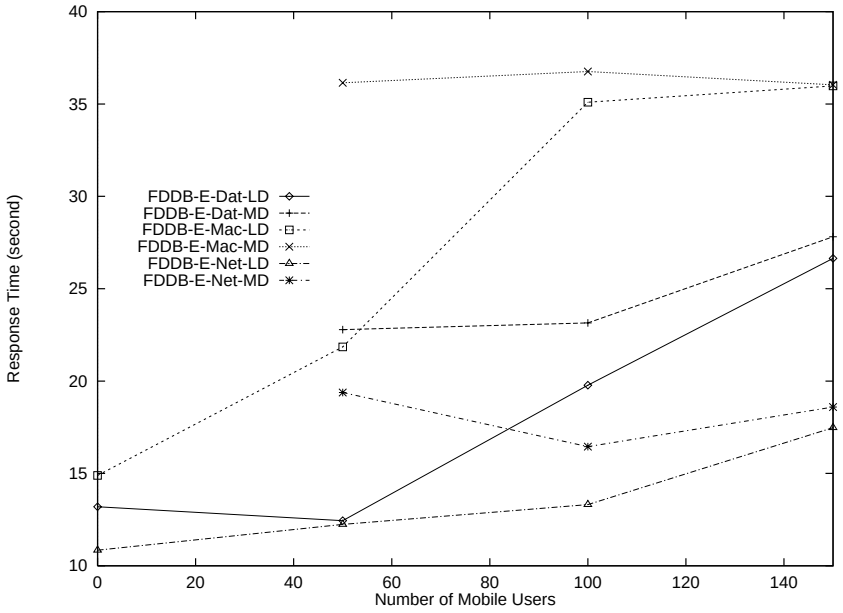

Figure 12: The response time for remote data accesses for "E" prioritization strategies (network-bandwidth = 1000K bits/sec).

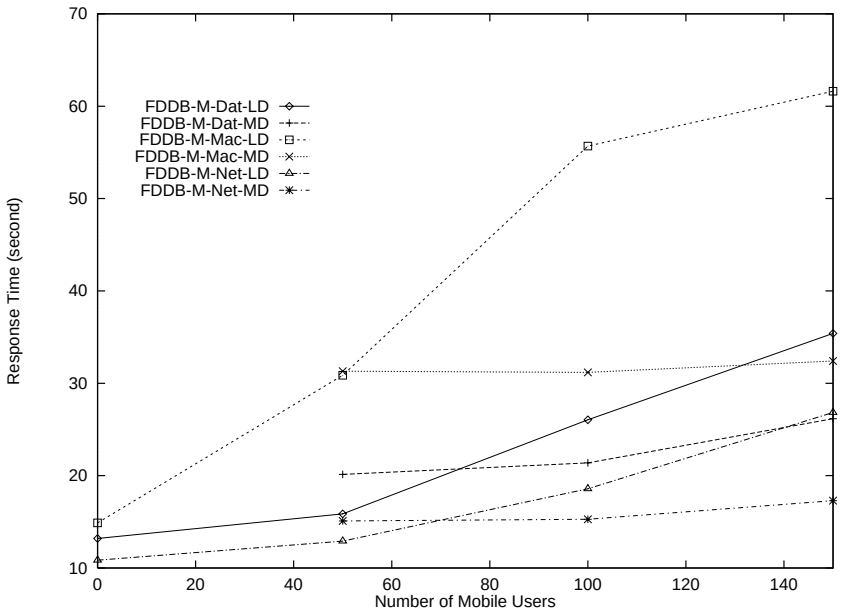

Figure 13: The response time for remote data accesses for "M" prioritization strategies (network-bandwidth = 1000K bits/sec).

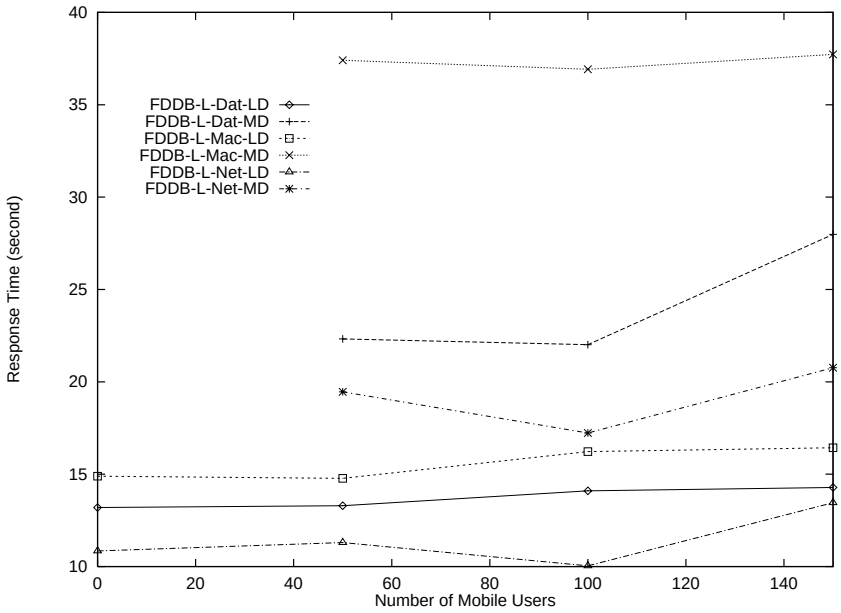

Figure 14: The response time for remote data accesses for "L" prioritization strategies (network-bandwidth = 1000K bits/sec).

5.4. Experiment 3: Effect of Communication Bandwidth

In this experiment, we study the impact of communication bandwidth on the algorithms. Due to space constraints, we limit our discussion to only the case of uniform query distribution. Moreover, we shall present only the data transmission and average response time graphs.

In Figures 15 and 16, we show the amount of data transmitted, and the average response time when network bandwidth is 100K bits per second. In Figures 17 and 18, we show the amount of data transmitted, and the average response time when network bandwidth is 10K bits per second. The interesting observation is that as the bandwidth gets smaller, the performance of "Mac"-based strategies become more inferior. In fact, when the "Net"-based strategies become superior for low or moderate number of mobile users. while the "Data"-based strategies outperform the rest for large number of mobile users.

6. Conclusion

In this paper, we have revisited the problem of processing queries in a highly mobile distributed environment. Unlike previous work on distributed databases, a highly mobile violates the concept of localization which most distributed database systems assumed.

We proposed a set of prioritization and load balancing strategies from which we derived a large set of algorithms. Based on a simulation study, our results showed that unless the communication is low, strategies that balance the workload based on the machine load performed best.

We plan to extend this work in the following areas. First, we plan to develop a more comprehensive model that incorporates buffering techniques. We also plan to study the impact of complex mobile queries. Next, we have already begun work on optimization of complex mobile queries. The main objective here is to minimize energy consumption at the mobile equipments without affecting the throughput on the database hosts at the wired network.

References

[Acha93] A. Acharya and B.R. Badrinath, "Delivering Multicast Messages in Networks with Mobile Hosts," in *Proceedings of the 13th International Conference on Distributed Computing Systems*, pp. 388–392, June 1993.

[Alon93a] R. Alonso and S. Ganguly, "Query Optimization for Energy Efficiency in Mobile Environment," in *Proceedings of the 1993 Workshop on Optimization in Databases*, Aigen, Austria, September 1993.

[Alon93b] R. Alonso and H. Korth, "Database Systems in Nomadic Computing," in *Proceedings of the 1993 ACM-SIGMOD International Conference on Management of Data*, pp. 388–392, June 1993.

[Awer91] B. Awerbuch and D. Peleg, "Concurrent On-line Tracking of Mobile Users," in *Proceedings of the ACM-SIGCOMM Symposium on Communication, Architectures and Protocols*, September 1991.

[Badr93] B.R. Badrinath, A. Acharya, and T. Imielinski, "Impact of Mobility on Distributed Computations," *Operating System Review*, Vol. 27, No. 2, April 1993.

[Badr95] B.R. Badrinath, A. Acharya, and T. Imielinski, "Designing Distributed Algorithms for Mobile Computing Networks," in *Technical Report, submitted for publication*, DCS, Rutgers University, 1995.

[Brya81] R. Bryant and R. Finkel, "A Stable Distributed Scheduling Algorithm," in *Proceedings of the 2nd International Conference on Distributed Computing Systems*, pp. 314–323, April 1981.

[Care86] M. J. Carey and H. Lu, "Load Balancing in a Locally Distributed Database System," in *Proceedings of the 1986 ACM-SIGMOD International Conference on Management of Data*, Washington DC, May 1986.

[Eage85] D.L. Eager, E.D. Lazowska, and J. Zahorjan, "A Comparison of Receiver-Initiated and Sender-Initiated Dynamic Load Sharing," in *Proceedings of the ACM-SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, August 1985.

[Hac86] A. Hac and T.J. Johnson, "A Study of Dynamic Load Balancing in a Distributed System," in *Proceedings of the ACM-SIGCOMM Symposium on Communications, Architecture and Protocol*, pp. 348–356, August 1986.

[Huan94] Y. Huang, P. Sistla, and O. Wolfson, "Data Replication for Mobile Computers," in *Proceedings of the 1994 ACM-SIGMOD International Conference on Management of Data*, pp. 13–24, June 1994.

[Imie93] T. Imielinski and B.R. Badrinath, "Data Management for Mobile Computing," *SIGMOD RECORD*, Vol. 22, No. 1, pp.34–39, March 1993.

[Imie94] T. Imielinski, S. Viswanathan, and B.R. Badrinath, "Energy Efficient Indexing on Air," in *Proceedings of the 1994 ACM-SIGMOD International Conference on Management of Data*, pp. 25–36, June 1994.

[Ioan93] J. Ioannidis and G. Q. Maguire, "The Design and Implementation of a Mobile Internetworking Architecture," in *1992 Winter Usenix*, January 1993.

[Lai95] S.J. Lai, A. Zaslavsky, G.P. Martin, and L.H. Yeo, "Cost Efficient Adaptive Protocol with Buffering for Advanced Mobile Database Applications," in *Proceedings of the 4th International Conference on Database Systems for Advanced Applications*, pp. 87–94, April 1995.

[Livn82] M. Livny and M. Melman, "Load Balancing in Homogeneous Broadcast Distributed Systems," in *Proceedings of the ACM Computer Network Performance Symposium*, April 1982.

[Lu90] H. Lu and K.L. Tan, "Buffer and Load Balancing in Locally Distributed Database Systems," in *Proceedings of the 6th International Conference on Data Engineering*, February 1990.

[Ni81] L. Ni and K. Abani, "Nonpreemtive Load Balancing in a Class of Local Area Networks," in *Proceedings of the Computer Networking Symposium*, December 1981.

[Ni82] L. Ni, "A Distributed Load Balancing Algorithm for Point-to-Point Local Computer Networks," in *Proceedings of the COMPCON*, Fall 1982.

[Ozsu91] M.T. Ozsu and P. Valduriez, *Principles of Distributed Database Systems,* Prentice-Hall, New Jersey, 1991.

[Proj94] Arjuna Project, "C++SIM User's Guide," 1994.

[Yu86] P.S. Yu, S. Balsamo, and Y.H. Lee, "Dynamic Load Sharing in Distributed Database Systems," in *Proceedings of the Fall Joint Computing Conference*, pp. 675–683, November 1986.

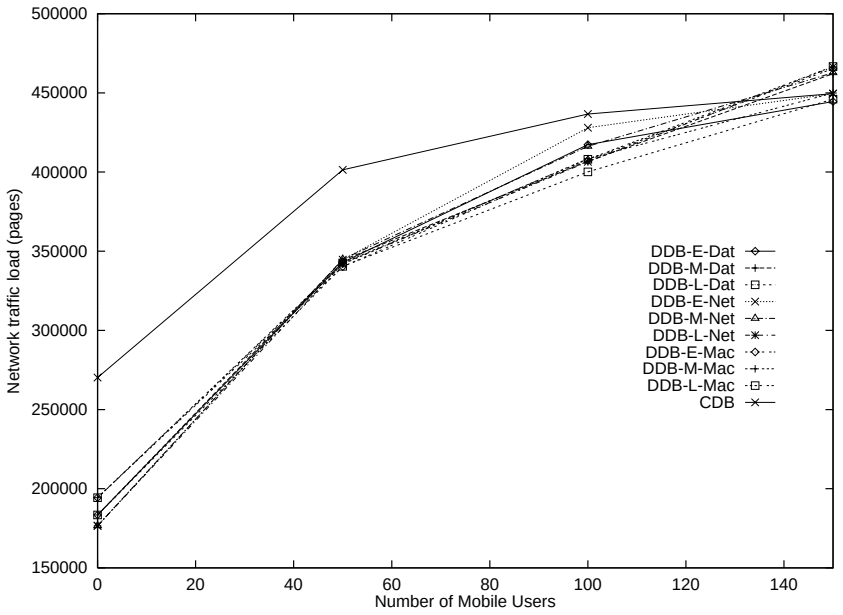

Figure 15: The data transferred when network-bandwidth = 100K bits/sec.

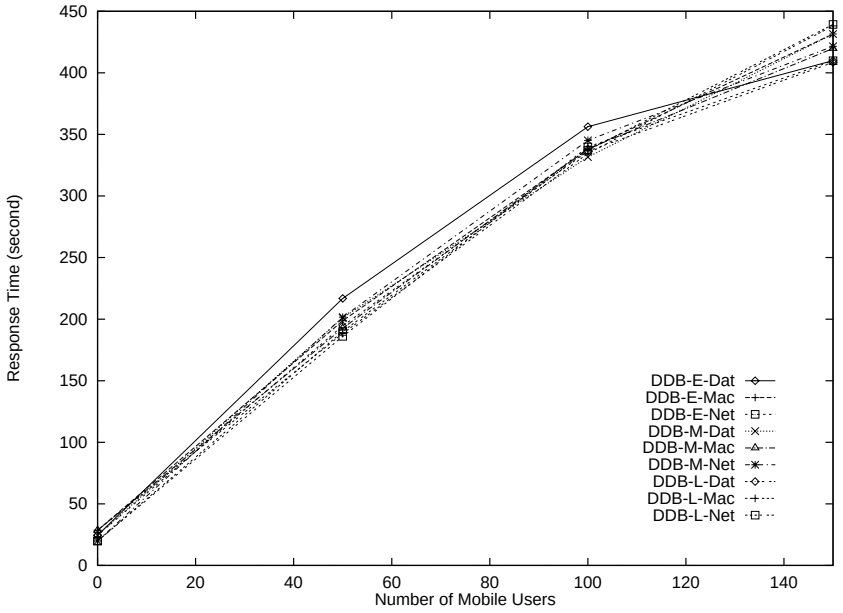

Figure 16: The average response time when network-bandwidth = 100K bits/sec.

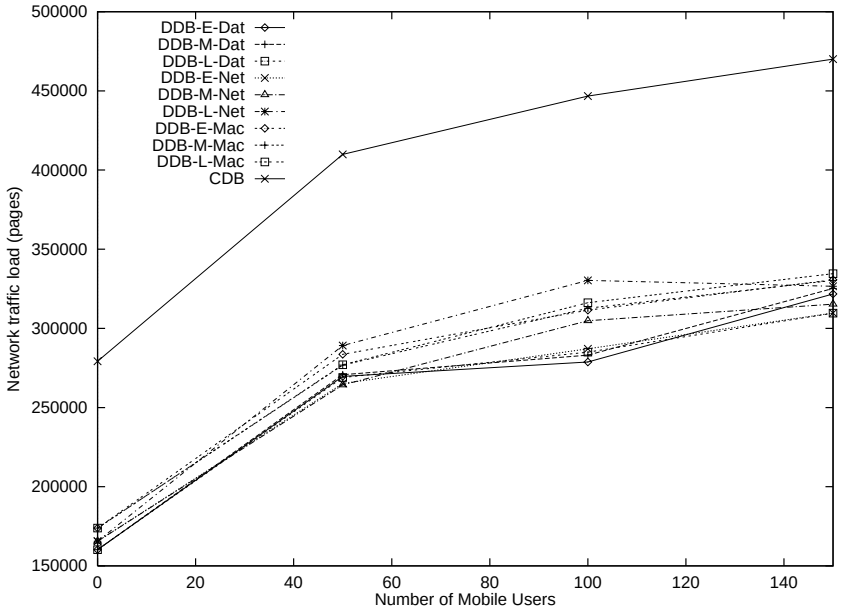

Figure 17: The data transferred when network-bandwidth = 10K bits/sec.

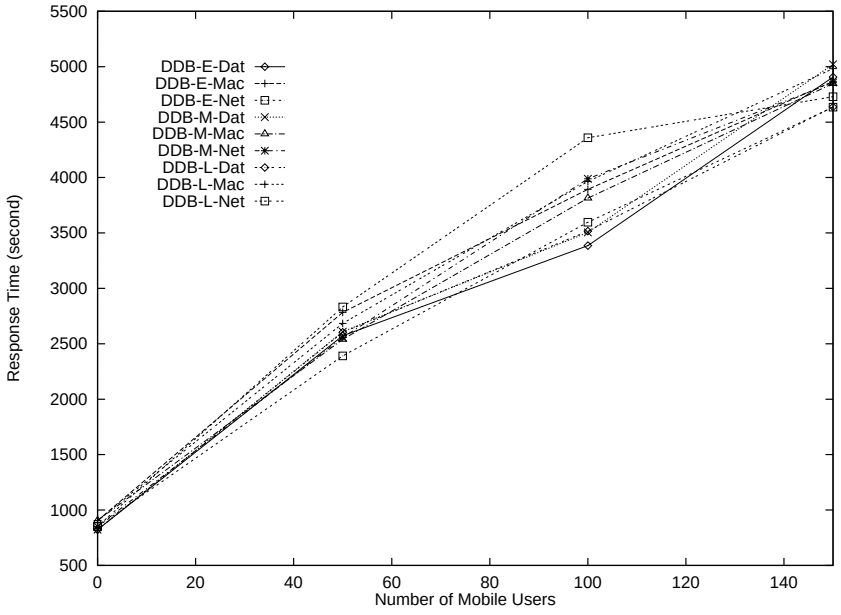

Figure 18: The average response time when network-bandwidth = 10K bits/sec.