Using Remote Joins for the Processing of Distributed Mobile Queries

Chang-Hung Lee and Ming-Syan Chen
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, ROC
E-mail:{ chlee@arbor.ee.ntu.edu.tw, mschen@cc.ee.ntu.edu.tw }

Abstract

The query processing in a mobile computing environment involves join processing among different sites which include static servers and mobile computers. In this paper, we first present some unique features of a mobile environment, and then, in light of these features, devise query processing methods for both join and query processing. Remote mobile joins are said to be effectual if they are, when interleaved into a join sequence, able to reduce the amount of data transmission cost required for distributed mobile query processing. It can be verified that the total data transmission cost of the processing in a distributed mobile query can be reduced by the algorithms designed by using effectual remote joins. A simulator is developed to evaluate the performance of algorithms devised. Our results show that the approach of interleaving the processing of distributed mobile queries with effectual remote mobile joins is not only efficient but also effective in reducing the total amount of data transmission cost required to process distributed mobile queries.

1 Introduction

Recently, as the number of mobile applications increases rapidly, there has been a growing demand for the use of distributed database architectures for various applications. Applications such as stock activities, traffic reports and weather forecast have become increasingly popular. Various wireless data networking technologies, including IS-136 [16], CDMA2000 [12], Wireless Application Protocol (WAP) and third generation mobile phone, have been developed recently. Among others, with the rapid advances in the palm computer technologies, a mobile computer is envisioned to be equipped with more powerful capabilities, including the storage of a small database and the capacity of data processing [17]. Consequently, the query processing in a mobile computing system which involves fixed hosts and several mobile computers has emerged as an issue of growing importance.

Consider an inventory application for example where a salesperson uses, for his/her work, a mobile computer devise in which a fragment of database contains the information of his/her customer records. In Figure 1 a portable computer, such as M_2, is hand-carried by this salesperson and is located at $Cell_1$, and F_1 and M_3 are also located at $Cell_1$. On the other hand, F_4, M_5, and M_6 with different data sets are allocated at $Cell_2$. F_1 and F_4 represent fixed hosts and M_2, M_3, M_5, and M_6 are mobile hosts. Note that depending on the corresponding coherency control mechanism employed, the data copy in the fixed host server could be obsolete. Since the most up-to-date data may be stored in the mobile computers, a query generated by a salesperson could be a sequence of joins to be performed across the relations residing in the server and several mobile computers, resulting in a very different execution scenario from the one for query processing in a traditional distributed system [5].

Consequently, we shall consider in this paper three important asymmetric features of a mobile computing system as pointed out in [14], and in light of these features, develop corresponding query processing schemes for mobile computing systems. The first asymmetric feature is on the computing capability between fixed hosts and mobile hosts. Usually mobile computers has limited resources for their computing operations, and the server is certainly much more powerful than a portable computing device. Note that in traditional distributed query processing, the sites involved in a query processing are usually assumed to have the same level of processing capability, which is, however, not valid in a mobile environment. The second asymmetric feature is on the transmission bandwidth between fixed hosts and mobile hosts. Clearly, the transmitting capability among mobile hosts is smaller than that among fixed hosts, since the transmission bandwidth of fixed hosts is usually much larger than that of mobile hosts. The third asymmetric feature is on the transmission cost coefficients among local hosts and remote hosts. The transmission cost required

0-7695-0996-7/01 $10.00 © 2001 IEEE

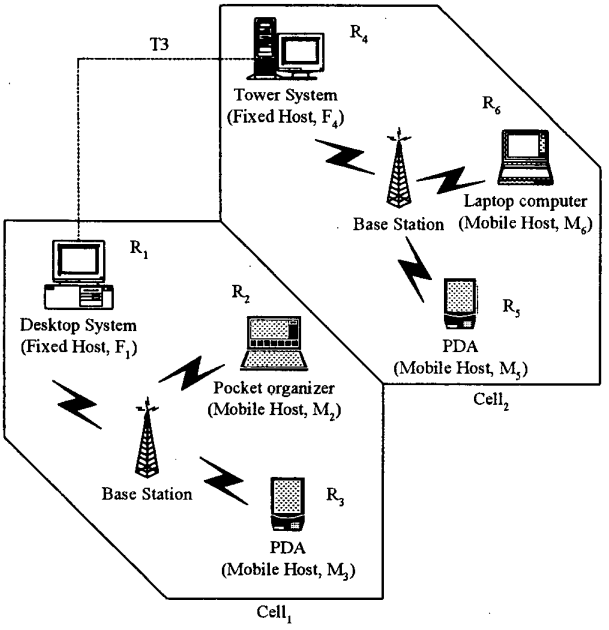


Figure 1. An illustrative mobile computing environment with mobile hosts and fixed hosts

for transmitting one unit of data among local hosts is much smaller than the corresponding cost required among remote hosts. These features distinguish the query processing in a mobile environment from the one in a traditional distributed system and also have to be considered when the costs of the corresponding operations are modelled [11, 13].

Due to the presence of asymmetric features in a mobile computing environment, the conventional query processing for a distributed database cannot be directly applied to a mobile computing system. In view of this, we shall explicitly devise query processing methods for both joins and query processing. Remote mobile joins are said to be *effectual* if they are, when interleaved into a join sequence, able to reduce the amount of data transmission cost required for distributed mobile query processing. Instead of processing queries by performing the minimum-cost joins sequentially as with conventional methodologies [3, 5, 6, 8], judiciously interleaving effectual remote mobile joins into a query scheduling can significantly reduce the total amount of data communication among different cells. It can be verified that the total data transmission cost of the processing in a distributed mobile query can be reduced by the algorithms designed by using effectual remote joins. In essence, the notion of remote mobile joins is extended from those of beneficial semijoins in [5] and join reducers in [6]. However, such an extension is not straightforward due to the very difference between a mobile computing environment and a distributed database system mentioned above. Performance studies on the sensitivity of various important parameters, including the number of mobile relations in a cell architecture, the number of relation tuples, and the amount of an attribute cardinality are also conducted. It is shown by our simulation results that by exploiting three asymmetric features with effectual remote mobile joins, these characteristic functions are very powerful in reducing the amounts of data transmission cost incurred, and can lead to the design of an efficient and effective query processing procedure for a mobile computing environment.

We mention in passing that without dealing with query processing, the authors in [1] studied the issues of optimization between energy consumption and server workload in a mobile environment. Several research efforts have been elaborated upon developing a location dependent query mechanism [7, 15, 18]. The authors in [10] presented the concept of queries with location constraints, i.e., constraints which involve location of mobile users. In [15], the authors proposed a spatial-temporal data model for querying of moving data in mobile environments. The position update policy and the impression of moving data are addressed in [18]. Without fully exploiting the asymmetric features of computing capability, the attention of prior studies was mainly paid to the query mechanisms with location constraints and query processing in traditional distributed databases [4, 8, 9, 13], but not to exploring the cost model and the query processing schemes for a mobile computing system.

This rest of this paper is organized as follows. Preliminaries are given in Section 2. Two join schemes and query processing schemes for multi-join queries are proposed in Section 3. Performance studies are conducted in Section 4. This paper concludes with Section 5.

2 Preliminaries

As in most previous works in distributed databases [3], we assume a query is in the form of conjunctions of *equi-join* predicates and all attributes are renamed in such a way that two join attributes have the same attribute name if and only if they have a join predicate between them. $|K|$ is used to denote the cardinality of a set K. For notational simplicity, the width of an attribute A and that of a tuple in R_i are assumed to be one unit. The size of the total amount of data in R_i can then be denoted by $|R_i|$. $|A|$ is used to denote the cardinality of the domain of an attribute A. Define the selectivity $\rho_{i,a}$ of attribute A in R_i as $\frac{|R_i(A)|}{|A|}$, where R_i(A) is the set of distinct values for the attribute A in R_i. $R_i - A \rightarrow R_j$ means a semijoin from R_i to R_j on attribute A. After the semijoin $R_i - A \rightarrow R_j$, the cardinality of R_j can be estimated as $|R_j|\rho_{i,a}$. To simplify the notation, $R_i \rightarrow R_j$ is

used to mean a semijoin from R_i to R_j in the case that the semijoin attribute does not have to be specified. Also, the notation $R_i \Rightarrow R_j$ is used to mean that R_i is sent to the site of R_j and a join operation is performed with R_j there. We use R_i' to denote the resulting relation after joins/semijoins are applied to an original relation R_i.

Conventionally, a function of the form $C(X) = c_0 + c_1 \cdot X$ is used to characterize communication cost, where X is the amount of data shipped from one site to another, c_1 is the communication cost per data unit, and the start-up connection cost c_0 is usually less significant. However, if the network topology is taken into consideration, the notion of identifying a *profitable* semijoin that prior work relied upon is incomplete, and in fact, might be misleading in some cases. Explicitly, c_1 is not a constant when network characteristics is considered, and its value is dependent upon the network topology.

In general, it is very difficult to determine a network cost model since the practical transmission bandwidth for a network traffic is in fact time-dependent. Hence, statistical average values of transmission bandwidth of the network are employed to provide a proper solution. As a consequence, the transmission coefficient $c_{m \to n}$ is used to serve as the statical average value in each network edge. We define an *effectual semijoin* as follows.

Definition 1, Effectual Semijoin: A semijoin, $R_1(S_1) - B \rightarrow R_2(S_2)$, is called *effectual,* if its cost of sending $R_1(B)$, i.e., $c_{1 \to 2}(|R_1(B)| = |B|\rho_{1,b})$, is smaller than its benefit, i.e., $c_{2 \to 1}(|R_2| - |R_2|\rho_{1,b} = |R_2|(1 - \rho_{1,b}))$, where R_1 and R_2 are located at sites S_1 and S_2 respectively, and $|R_2|$ and $|R_2|\rho_{1,b}$ represent, respectively, the sizes of R_2 before and after the semijoin. Thus, $|R_1(B)|c_{1 \to 2}$ is used to denote the cost of a semijoin $R_1 - B \rightarrow R_2$.

Symb.	Description
c^L_{FF}	Local trans. cost coeffi. among fixed hosts
c^L_{MF}	Local trans. cost coeffi. between mobile and fixed hosts
c^L_{MM}	Local trans. cost coeffi. among mobile hosts
c^R_{FF}	Remote trans. cost coeffi. among fixed hosts
c^R_{MF}	Remote trans. cost coeffi. between mobile and fixed hosts
c^R_{MM}	Remote trans. cost coeffi. among mobile hosts
r^{RL}_{FF}	Trans. cost ratio between remote and local fixed hosts
r^{RL}_{MM}	Trans. cost ratio between remote and local mobile hosts
r^L_{MF}	Trans. cost ratio between local mobile and fixed hosts
r^R_{MF}	Trans. cost ratio between remote mobile and fixed hosts

Table 2. Descriptions of symbols for the cost model in a mobile computing system

Consequently, we derive a cost model which considers these three asymmetric features of a mobile computing system. Our model consists of two distinct sets of entities: mobile hosts and fixed hosts [2]. Furthermore, we use *local* and *remote* to indicate two different communication modes. Local communication means that the transmission is among hosts in the same cell, whereas remote communication means that the transmission is among different cells. For ease of our discussion, symbols used are shown in Table 2. c^L_{FF} denotes local transmission cost coefficient among fixed hosts and we assume c^L_{FF} is a basic coefficient and its value is given as one unit for transmitting one unit of data among local fixed hosts. The local transmission cost coefficient among mobile hosts is denoted by c^L_{MM}. Analogly, we use c^L_{MF} to indicate the local transmission cost coefficient between mobile hosts and fixed hosts. For remote communication, we have three parameters to model the transmission costs among mobile and fixed hosts, i.e., c^R_{FF}, c^R_{MM}, and c^R_{MF}. In addition, several transmission cost ratios are used to represent the relationship among these transmission coefficients, i.e., $r^{RL}_{FF} = \frac{c^R_{FF}}{c^L_{FF}}$, $r^{RL}_{MM} = \frac{c^R_{MM}}{c^L_{MM}}$, $r^L_{MF} = \frac{c^L_{MM}}{c^L_{FF}}$, and $r^R_{MF} = \frac{c^R_{MM}}{c^R_{FF}}$.

3 Query Processing in a Mobile Computing System

Join processing in a mobile computing system is discussed in Section 3.1. The query processing scheme with a divide-and-conquer technique based on the cell architecture (to be referred to as scheme QP_C) is discussed in Section 3.2. The scheme that is devised with effectual remote mobile joins (to be referred to as scheme QP_R) is described in Section 3.3.

3.1 Join Processing in a Mobile Computing System

We now derive the solution procedure for minimizing the cost of join methods in a mobile computing system. Consider the scenario of join processing in Figure 2, where the fixed host F_1 has relation R_1 and the fixed hosts F_2 has relation R_2. R_3 is located at the mobile host M_3. Suppose that the mobile user M_3 submits a query that performs a join operation of R_1, R_2 and R_3 on their common attribute A and B, $R_1.A = R_3.A$ and $R_2.B = R_3.B$, with the corresponding selectivity factors ρ_A and ρ_B, respectively. We will select F_1 as the location for storing the join result. With this given model, we shall examine two join methods. To simplify our presentation, $TC(J)$ is used to represent the data transmission cost of the join method J.

In what follows, we examine a join sequence which preforms the joins based on cell architecture with a divide-and-conquer technique in Section 3.1.1. Section 3.1.2. describes the effectual remote mobile join method. Analysis of these join methods is given in Section 3.1.3.

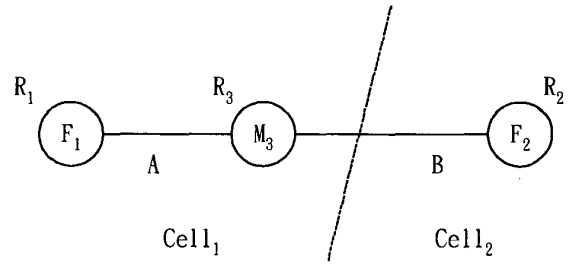


Figure 2. An example scenario for join processing

3.1.1 Processing Joins with Divide-and-Conquer (denoted by J_C)

Consider a query in Figure 2 as an example. Traditionally, the query processing is performed based on the minimum-cost join in a forward-scheduling manner [13]. Since the transmission cost among local communication path is more inexpensive than that among remote communication path, the query will be naturally divided into two separated sub-queries based on the cell architecture and processed independently. This is how the notion of divide and conquer comes out. One is the sub-query belonging to the communication cell $Cell_1$, and the other is belonging to $Cell_2$. After the join results of each sub-query are merged into a fixed host, the residue relations can be processed with the new query. Note that, with forward scheduling method the join processing, merging the partial database R_3 on M_3 to R_1 of F_1 will be the most efficient processing. As a result, a cost of $TC(R_3 \Rightarrow R_1) = c^L_{MF} * |R_3|$ is incurred and a new relation R'_1 is generated in F_1, where $|R'_1| = \frac{|R_1||R_3|}{|A|}$.

After all of local join sequences in each sub-query are finished, two separated sub-queries are merged to be a new query, i.e., $R'_1.B = R_2.B$ between F_1 and F_2. Since the amount of tuples storing in the fixed host database is much larger than the number of an attribute cardinality, i.e., both $|R_1|$ and $|R_2|$ are much larger than $|B|$ in a mobile environment, an effectual semijoin occurs between these two residual relations in fixed hosts. Because of $|R_1| \gg |B|$, $\rho_{1,B}$ is assumed to be unchanged after the join processing. In other words, a semijoin $R'_1 - B \rightarrow R_2$ and a join $R'_2 \Rightarrow R'_1$ will be processed in this merged query, which leads to a cost of $TC(R'_1 - B \rightarrow R_2) + TC(R'_2 \Rightarrow R'_1) = c^R_{FF} * |R'_1(B)| + c^R_{FF} * \rho_{1,B} * |R_2|$. Then, the corresponding costs is summarized as follows.

$$TC(J_C) = c^L_{MF} * |R_3| + c^R_{FF} * \rho_{1,B} * (|B| + |R_2|).$$

3.1.2 Processing Joins with Remote Mobile Join (denoted by J_R)

Next, consider the case of join processing with remote mobile joins. Instead of merging the join operation between F_1 and M_3, R_3 is merged to R_2, followed by the join processing between F_1 and F_2. Even though the remote transmission cost coefficient between mobile hosts and fixed hosts, i.e., c^R_{MF}, is much larger than the local transmission cost between mobile hosts and fixed hosts, i.e., c^L_{MF}, it can be still profitable with a high reduction ratio, leading to the use of an effectual remote mobile join. The total transmission cost will be $TC(R_3 \Rightarrow R_2) + TC(R_1 - B \rightarrow R'_2) + TC(R''_2 \Rightarrow R_1)$, where $TC(R_3 \Rightarrow R_2) = c^R_{MF} * |R_3|$, $TC(R_1 - B \rightarrow R'_2) + TC(R''_2 \Rightarrow R_1) = c^R_{FF} * (\rho_{1,B} * |B| + |R''_2|)$ and $|R''_2| = \rho_{1,B} * \frac{|R_1||R_3|}{|A|}$. Consequently, we have corresponding costs below.

$$TC(J_R) = c^R_{MF} * |R_3| + c^R_{FF} * \rho_{1,B} * (|B| + \frac{|R_1||R_3|}{|A|}).$$

3.1.3 Analysis of Join Processing

To examine the amount of data transmission cost incurred by J_C and J_R. Specifically, the criterion of identifying an effectual remote mobile join to reduce the amount of data transmission cost is derived. In practice, the local transmission cost coefficient between local mobile hosts and local fixed hosts c^L_{MF} is very close to the value among local mobile hosts c^L_{MM}. To simplify our discussion, $c^L_{MF} = c^L_{MM}$ and $c^R_{MF} = c^R_{MM}$ are assumed in this paper. Note that such as assumption is made for ease of discussion, and is not essential for the use of remote joins we propose in this paper.

Lemma 1: $c^R_{FF} = \frac{r^{RL}_{MM}}{r^R_{MF}} * c^L_{MF}$.

Proof: According to the definition of $c^L_{MF} = c^L_{MM}$, $c^R_{MF} = c^R_{MM}$, $c^R_{MM} = r^R_{MF} * c^R_{FF}$ and $c^R_{MM} = r^{RL}_{MM} * c^L_{MM}$, it can be seen that $c^R_{MF} = r^{RL}_{MM} * c^L_{MF}$ and $c^R_{MF} = r^R_{MF} * c^R_{FF}$. As a consequence, with $r^R_{MF} * c^R_{FF} = r^{RL}_{MM} * c^L_{MF}$, we have $c^R_{FF} = \frac{r^{RL}_{MM}}{r^R_{MF}} * c^L_{MF}$. **Q.E.D.**

Lemma 2: With $\frac{r^R_{MF} \times (r^{RL}_{MM} - 1)}{r^{RL}_{MM}} < \rho_{1,B} * (\frac{|R_2|}{|R_3|} - \frac{|R_1|}{|A|})$, the amount of data transmission cost incurred by method J_R is smaller than that by method J_C, i.e., $TC(J_R) < TC(J_C)$, where R_2 is a remote fixed host and R_3 is an example of the local mobile host.

With Lemma 2, an effectual remote mobile join is defined as follows.

Definition 2: A remote mobile join is called *effectual* if and only if $TC(J_R)$ is smaller than $TC(J_C)$.

With Lemma 1, Lemma 2, and Definition 2, we can derive following theorem. According to Theorem 1, effectual remote mobile joins can be interleaved into the query scheduling to reduce the data transmission cost of multi-join processing.

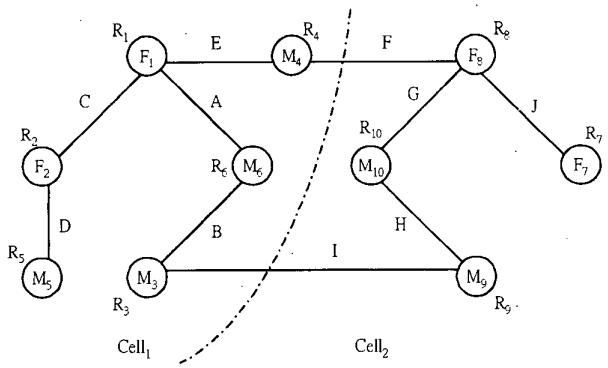


Figure 3. Division of a query

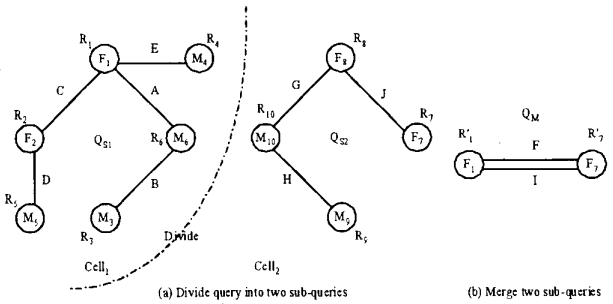


Figure 4. Query processing with QP_C methodology

Theorem 1: A remote mobile join is *effectual* if and only if $\frac{r^R_{MF} \times (r^{RL}_{MM} - 1)}{r^{RL}_{MM}} < \rho_{1,B} * (\frac{|R_2|}{|R_3|} - \frac{|R_1|}{|A|})$, where $|R_3|$ is the size of relations in a remote fixed host, $\rho_{1,B}$ denotes the selectivity of a relation in the local fixed host, and $|R_2|$ is the size of a relation in the local mobile host.

It can be verified that by judiciously applying effectual remote mobile joins, method J_R can reduce the amount of data transmission cost as a whole. As can been seen later, Theorem 1 derived above can be employed to determine the threshold for whether method J_R should be utilized.

3.2 Query Processing with Divide-and-Conquer (denoted by QP_C)

Consider the illustrative query in Figure 3 as an example where the destination site is F_1. In scheme QP_C, the J_C method is utilized. First, the query is divided into two sub-queries and each sub-query is processed with forward scheduling algorithm. In Figure 4a, Q_{S1} and Q_{S2} belong to $Cell_1$ and $Cell_2$, respectively. R_1, R_2, R_3, R_4, R_5 and R_6 are located at Q_{S1} and R_7, R_8, R_9 and R_{10} in contrast belong to sub-query Q_{S2}. After each partial result of subquery is generated, we merge these residue relations to be a new query. Then, the forward scheduling algorithm is utilized again for the new query processing. Let R_F denote the relation in a fixed host and R_M be the relation in a mobile host. Note that since the amount of $|R_F|$ is usually much larger than $|R_M|$, the partial result of each sub-query will naturally be located at the fixed host. Therefore, we assume that the query result R'_1 of Q_{S1} is located in F_1 and the result R'_7 of Q_{S2} is located in F_7. Then, we merge these two relations into a new query Q_M as shown in Figure 4b. With the join processing of merging R'_7 to R'_1, the final query result is generated in F_1. Procedure QP_C and algorithm forward scheduling (abbreviatedly as FS) are outlined below.

Procedure QP_C : Determine the scheduling of multi-join queries based on the cell architecture.
Step 1: Based on cell architecture, divide the original query into several sub-queries.
Step 2: Process each sub-query with algorithm forward scheduling.
Step 3: Merge residue relations from each sub-query into be a new query, which is referred to as a conquer query.
Step 4: Do the query processing of the conquer query with forward scheduling algorithm again and generate the query result.
Step 5: Send the query result to the needed destination.

Algorithm *Forward Scheduling* (algorithm FS): Determine the join sequence starting from performing the minimum-cost join.
Step 1: Perform effectual semijoins in the query.
Step 2: With join processing, merge relations from the path of minimum transmission cost.
Step 3: Reorganize the query.
Step 4: If the query is empty, go to Step 5. Otherwise, go back to Step 2.
Step 5: End

3.3 Query Processing with Effectual Remote Mobile Joins (denoted by QP_R)

Clearly, scheme QP_C does not exploit the relationship among remote relations, and may thus consume much valuable communication cost for the join processing in the merged query Q_M. Instead of partitioning the query into several sub-queries based on the cell architecture as in scheme QP_C, the concept of the effectual remote mobile join will be employed in algorithm QP_R. According to Theorem 1, an effectual remote mobile join can successfully reduce the transmission cost. The corresponding figures of each step in QP_R procedure are illustrated in Figure 5. For ease of exposition, $L_d(\ )$ denotes a set of local joins in the destination cell and $L_r(\ )$ is the set of local joins in a remote cell. In addition, $R(\ )$ represents a set of the remote joins

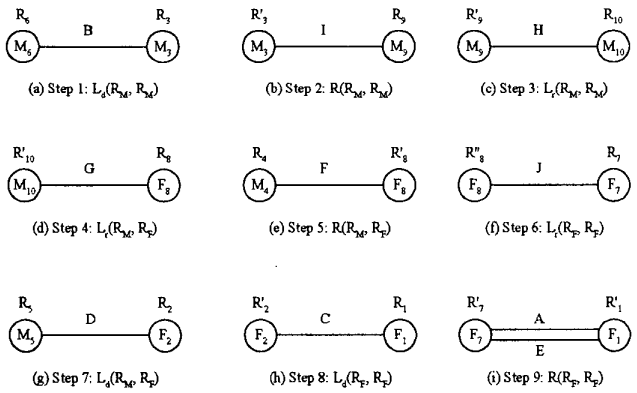


Figure 5. Query processing with QP_R methodology

across different cells. For example, $L_d(R_M, R_M)$ denotes the set of joins among local mobile relations in the destination cell.

First, in Step 1 of Figure 5a, connected relations among fixed hosts and mobile hosts in the cell of query destination are merged with algorithm FS. For ease of our discussion, we assume that the join result of $R_6.B = R_3.B$ is merged to M_3. The relationship $R'_3.I = R_9.I$ among mobile hosts located in different cells is exploited by the join processing in Step 2. The result is in M_9 as shown in Figure 5b if $R(R_M, R_M)$ can induce effectual remote mobile joins. In Step 3 of Figure 5c, we merge $R'_9.H = R_{10}.H$ of the connected mobile hosts in remote cells to the mobile host M_{10}. Then, Figure 5d shows that R'_{10} is merged to the fixed host F_8 in Step 4. Using effectual remote mobile joins $R(R_M, R_F)$ in Step 5, mobile relations in the local cell are merged into fixed hosts in the remote cell. Figure 5f indicates the operation of merge relations in remote fixed hosts to F_7 in Step 6. Furthermore, the merge operations among local mobile hosts and local fixed hosts are performed in Step 7 as shown in Figure 5g. Similarly, the merged result R'_2 is assumed to be located in F_2. Then, we merge relations of the fixed hosts in the local cell to F_1 with $L_d(R_F, R_F)$ of Step 8 in Figure 5h. Finally, Figure 5i illustrates the final step of merging the relations in remote fixed hosts to the local fixed host F_1. The final result is generated in Step 9 of QP_R. Procedure QP_R is outlined below. Note that in each step, the merging processing is based on algorithm FS.

Procedure QP_R : Determine the scheduling of multi-join queries with remote mobile joins

Step 1: Merge relations in mobile hosts which are connected with each other in the destination cell of query.

Step 2: If there exists effectual remote mobile joins among relations in mobile hosts, merge those relations to the mobile hosts in remote cell.

Step 3: Merge relations in mobile hosts which are connected with each other in remote cells.

Step 4: Merge relations from mobile hosts to fixed hosts, where mobile hosts and fixed hosts are connected with each other in remote cells.

Step 5: If there exists effectual remote mobile joins among mobile hosts and fixed hosts, merge relation in mobile hosts of the destination cell to the fixed hosts in remote cells.

Step 6: Merge relations in fixed hosts which are connected with each other in remote cells.

Step 7: Merge relations from mobile hosts to fixed hosts, where mobile hosts and fixed hosts are in the destination cell of query.

Step 8: Merge relations in fixed hosts which are in the destination cell of query.

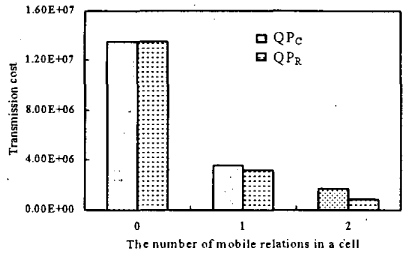
Step 9: Merge residue relations in fixed hosts to the fixed host of the destination cell.

4 Performance studies

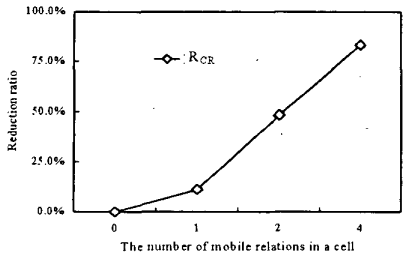
Symb.	Description	Default
$\|N_M\|$	Number of mobile relations in a cell	2
p_{QG}	Intensity of a query graph	0.5
$\|R_M\|$	The ave. tuples for relations in mobile hosts	500
$\|R_F\|$	The ave. tuples for relations in fixed hosts	5×10^5
$\|K\|$	The ave. size of cardinality for attributes	2,500
c^L_{FF}	Local trans. cost coeffi. among fixed hosts	1
r^{RL}_{FF}	$r^{RL}_{FF} = \frac{c^R_{FF}}{c^L_{FF}}$	30
r^L_{MF}	$r^L_{MF} = \frac{c^L_{MM}}{c^L_{FF}}$	10
r^R_{MF}	$r^R_{MF} = \frac{c^R_{MM}}{c^R_{FF}}$	1.5

Table 3. Default values of model parameters

Simulations were performed to evaluate the effectiveness of join processing methods and query processing schemes. The number of relations in a query was pre-determined. The occurrence of an edge between two relations in the query graph was determined according to a given probability, denoted by p_{QG}. The number of cells to be evaluated is assumed to be two and only one fixed server host is located in each communication cell. In addition to two mobile hosts in each cell, we also assume that each host only contains one relation. With merge operations, we can merge several fixed hosts in the same cell together and combine several remote cells to be one unit of cell. As such, despite of its simplicity, our model can still reflect some degree of the reality. For ease of exposition, unless mentioned otherwise the default value of each parameter is given in Table 3. The selectivity of relation attributes in mobile hosts is randomly generated in the range of 0.1 to 0.2 while that in fixed hosts is in the range of 0.8 to 0.95. In addition, the communication costs across remote hosts are more expensive than those across local hosts. Thus, r^{RL}_{FF} and r^{RL}_{MM} are in general larger

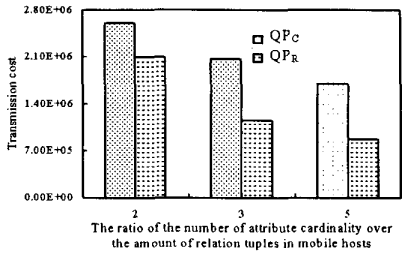


(a) Transmission cost of QP_C and QP_R

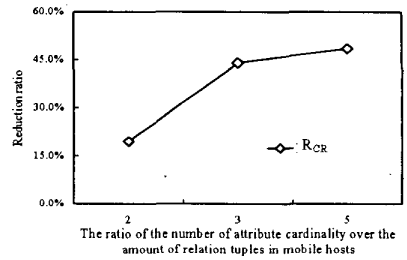


(b) Reduction ratio between QP_C and QP_R

Figure 6. Performance studies on various values of $|N_M|$ in each cell



(a) Transmission cost of QP_C and QP_R



(b) Reduction ratio between QP_C and QP_R

Figure 7. Performance studies on attribute cardinalities

than one. Similarly, r_{MM}^{RL} and r_{MF}^{L} are larger than one due to the asymmetry features between mobile hosts and fixed hosts. Moreover, the density of query is given as $p_{QG} = 0.5$ and each execution cost is the result of the average from 20 query executions. To simplify our presentation, the execution cost of algorithms A denoted by Cost(A), where A can be QP_C or QP_R. To exhibit the benefit of relation replication, the reduction ratio $R_{CR} = |\frac{Cost(QP_C) - Cost(QP_R)}{Cost(QP_C)}|$ is used as a metric to compare QP_C and QP_R.

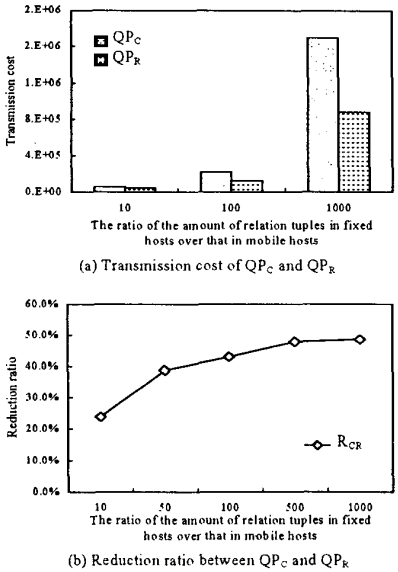
Figure 6 shows the performance results for the number of mobile relations $|N_M|$ in each cell. Explicitly, more mobile relations in a cell will lead to a higher possibility to have the effectual mobile joins as reducers in the query processing. As a result, with the growth of $|N_M|$ in each cell, the transmission costs required by both algorithms QP_C and QP_R decrease as shown in Figure 6a. In Figure 6b, it can be seen that with the presence of effectual remote mobile joins, QP_R outperforms QP_C. A higher reduction ratio R_{CR} is observed for large numbers of $|N_M|$.

Figure 7 shows the performance results for the ratio of attribute cardinalities over the amount of relation tuples in the mobile hosts. Consequently, with the growth of attribute cardinalities, both of the transmission costs of QP_C and QP_R decrease, as shown in Figure 7a. Figure 7b shows that due to the use of the remote mobile joins, the advantage of QP_R over QP_C increases as the number of attribute cardinalities increases.

The horizontal axis in Figure 8 indicates the value of $\frac{|R_F|}{|R_M|}$. With fixed size of the relation tuples in mobile hosts, the increase of the number of tuples in fixed hosts will lead to more transmission costs required in the query processing of both QP_C and QP_R as shown in Figure 8a. Specifically, as shown in Figure 8b, QP_R exhibits a better scheduling than QP_C for a multi-join query processing with the growth of $\frac{|R_F|}{|R_M|}$. Note that effectual remote mobile joins are more powerful to deal with the large amount of relation tuples in remote fixed hosts, thereby reducing the amount of data transmission costs incurred. Consequently, QP_R can lead to the design of an efficient and effective query processing procedure for a mobile computing environment.

5 Conclusions

In this paper, we have explored some unique features of a mobile environment, and then, in light of these features, we devised query processing methods for both join and query processing. Judiciously interleaving effectual remote mobile joins into a query scheduling can significantly reduce the total amount of data communication among different cells. It was verified that the total data transmission cost of the processing in a distributed mobile query was reduced by the algorithms designed by using effectual remote joins. A simulator was developed to evaluate the performance of algorithms devised. Our results show that the approach of interleaving the processing of distributed mobile queries with effectual remote mobile joins is not only efficient but also



(a) Transmission cost of QP_C and QP_R

(b) Reduction ratio between QP_C and QP_R

Figure 8. Performance studies on the number of tuples in fixed hosts.

effective in reducing the total amount of data transmission cost required to process distributed mobile queries.

6. Acknowledgment

The authors are supported in part by the Ministry of Education Project No. 89-E-FA06-2-4-7 and the National Science Council, Project No. NSC 89-2219-E-002-028 and NSC 89-2218-E-002-028, Taiwan, Republic of China.

References

[1] R. Alonso and S. Ganguly. Query Optimization in Mobile Environments. In *Fifth Workshop on Foundations of Models and Languages for Data and Objects*, pages 1–17, September 1993.

[2] D. Barbara. Mobile Computing and Databases–A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 11(1):108–117, January/February 1999.

[3] G. Bojan and M. M. Qutaibah. Combinatorial Optimization of Distributed Queries. *Transactions on Knowledge and Data Engineering*, pages 915–927, December 1995.

[4] S. Ceri and G. Pelagatti. *Distributed Databases Principles and Systems*. McGraw-Hill.

[5] M.-S. Chen and P. S.Yu. Interleaving a Join Sequence with Semijoins in Distributed Query Processing. *IEEE Transactions on Parallel and Distributed Systems*, 3(5):611–621, September 1992.

[6] M.-S. Chen and P. S.Yu. Combining Join and Semijoin Operations for Distributed Query Processing. *IEEE Transactions on Knowledge and Data Engineering*, 5(3):534–542, June 1993.

[7] M. H. Dunham and V. Kumar. Location Dependent Data and its Management in Mobile Databases. In *Proceedings of the Ninth International Workshop on Database and Expert Systems Applications*, pages 26–29, August 1998.

[8] M. J. Franklin, B. T. Jonsson, and D. Kossmann. Performance Tradeoffs for Client-Server Query Processing. In *Proceeding of ACM SIGMOD*, pages 149–160, June 1996.

[9] S. Ganguly. Design and Analysis of Parametric Query Optimization Algorithms. In *Proceedings of the 24th International Conference on Very Large Data Bases*, pages 228–238, 1998.

[10] T. Imielinski and B. R. Badrinath. Querying in Highly Mobile and Distributed Environment. In *Proceeding of the 18th International Conference on Vary Large Data Bases*, pages 41–52, August 1992.

[11] J. Jing, A. Helal, and A. Elmagarmid. Client-Server Computing in Mobile Environments. *ACM Computing Surveys*, 31(2):117–157, June 1999.

[12] D. N. Knisely, S. Kumar, S. Laha, and S. Nanda. Evolution of Wireless Data Services: IS-95 to CDMA2000 . In *IEEE Communications Magazine*, pages 140–149, October 1998.

[13] D. Kossmann. The State of the Art in Distributed Query Processing. In *ACM Computing Survey*, September 2000.

[14] W.-C. Peng and M.-S. Chen. Exploiting the Features of Asymmetry for Query Processing in a Mobile Computing Environment. In *Proc. of Fifth IFCIS Conference on Cooperative Information Systems (CoopIS'2000)*, 2000.

[15] A. P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao. Modeling and Querying Moving Objects. In *Proceeding of the 13th International Conference on Data Engineering*, pages 422–432, April 1997.

[16] N. R. Sollenberger, N. Seshadri, and R. Cox. The Evolution of IS-136 TDMA for Third-Generation Wireless Services. In *IEEE Personal Communications*, pages 8–18, June 1999.

[17] WAP Forum. http://www.wapforum.org/.

[18] O. Wolfson, S. Chamberlain, S. Dao, L. Jiang, and G. Mendez. Cost and Imprecision in Modeling the Position of Moving Objects. In *Proceeding of the 14th International Conference on Data Engineering*, pages 588–596, February 1998.