

Exploiting the Features of Asymmetry for Query Processing in a Mobile Computing Environment

Wen-Chih Peng and Ming-Syan Chen
Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, ROC

E-mail: {mschen@cc.ee.ntu.edu.tw, wcpeng@arbor.ee.ntu.edu.tw}

Abstract. With the cutting edge technology advance in wireless and mobile computers, the query processing in a mobile environment involves join processing among different sites which include static servers and mobile computers. Because of the need of energy saving and also 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 this paper, we first explore three asymmetric features of a mobile environment. Then, in light of these features, we devise query processing schemes. Performance of these query processing schemes is comparatively analyzed and sensitivity analysis on several parameters is conducted.

1 Introduction

In a mobile computing environment, a mobile user with a power-limited palm computer (or a mobile computer) can access various information via wireless communication. It is noted that mobile computers use small batteries for their operations without directly connecting to any power source and the bandwidth of wireless communication is in general limited. As a result, an important design issue in a mobile system is to conserve the energy of a mobile unit while allowing mobile users of the ability to access information from anywhere at anytime [3][9].

The query processing in a traditional distributed system has received a considerable amount of attention and been extensively studied in the literature [4][5][6][11]. The objective in distributed query processing is to reduce the amount of data transmission required. Note, however, that the cost models developed for query processing in a traditional distributed database do not reflect many important features in a mobile computing system. Explicitly, the prior studies in distributed query processing [5][6][11] did not fully explore the asymmetric features of a mobile environment, which are, as explained below, particularly important in devising the corresponding query processing schemes. Specifically, the energy consumption of mobile computers, the most important cost criterion, was not dealt with for the query processing in traditional distributed databases, making the corresponding distributed query processing schemes devised not applicable to a mobile computing environment. To remedy this, we shall

explore in this paper three important asymmetric features of a mobile computing system, and in light of these features, develop efficient query processing schemes for mobile computing systems. The three asymmetric features, which we shall explicitly address and reflect in the design of query processing schemes, are as follows.

1. **Asymmetric feature of computing capability between the server and a mobile computer:** Mobile computers use small batteries for their operations without directly connecting to any power source. In contrast, the server is not strictly constrained by energy consumption and thus possesses much more power than a mobile computer. 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. This feature distinguishes the query processing in a mobile environment from that in a traditional distributed system.
2. **Asymmetric feature of energy consumption between message sending and receiving:** The energy required for message sending is more than that required for message receiving at a mobile computer [8]. This feature also has to be modelled when the costs of the corresponding operations are evaluated.
3. **Asymmetric feature of energy consumption between activeness and idleness of a mobile computer:** The energy consumed by a mobile computer in its active mode is much more than that consumed in its idle mode [7][8]. In view of this, a mobile computer may be designed to stay in its idling mode by migrating its processing work to the server if so appropriate.

Consequently, we derive in this paper a cost model which considers these three asymmetric features of a mobile computing system. The cost model derived paves the way to the development of the query processing schemes in a mobile computing system. For ease of exposition, a semijoin which is initiated by the server and is beneficial to reduce the cost of a join operation is termed a *server-initiated*, or *SI profitable* semijoin. For query processing, judiciously applying SI profitable semijoin can reduce the amount of data transmission required and energy consumption. According to those asymmetric features of a mobile computing system, we devise some specific criteria to identify SI profitable semijoins. For query processing which refers to the processing of multi-join queries, we develop three query processing schemes. In particular, we formulate the query processing in a mobile computing system as a two-phase query processing procedure that can determine a join sequence and interleave that join sequence with SI profitable semijoins to reduce both the amounts of data transmission and energy consumption. Performance of these query schemes is comparatively analyzed and sensitivity analysis on several parameters is conducted. It is shown by our simulation results that by exploiting the three asymmetric features, the proposed two-phase query processing scheme is very powerful in reducing both the amounts of energy consumption and data transmission incurred.

This rest of this paper is organized as follows. Preliminaries are given in Section 2. In Section 3, we develop query processing schemes for multi-join queries. Performance studies on various query processing schemes are conducted in Section 4. This paper concludes with Section 5.

2 Preliminaries

To facilitate the presentation of this paper, some preliminaries are given in this section. The notation, definitions and assumptions required are described in Section 2.1. By taking the asymmetric features described above into consideration, a cost model for join and query processing in a mobile computing system is devised in Section 2.2.

2.1 Notation, Definition and Assumption

A join query graph can be denoted by a graph $G = (V, E)$, where V is the set of nodes and E is the set of edges. Each node in a join query graph represents a relation. Two nodes are connected by an edge if there exists a join predicate on some attribute of the two corresponding relations. We use $|R_i|$ to denote the cardinality of a relation R_i and $|A|$ to denote the cardinality of the domain of an attribute A . The notation $R_i \bowtie R_j$ is used to mean the join between R_i and R_j , and $|R_i \bowtie R_j|$ denotes cardinality of the result relation of $R_i \bowtie R_j$. To determine the effect of a join operation specified by a query graph, we employ the results stated in the theorem developed in [5].

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 and $R_i \rightarrow R_j$ means a simple join from R_i to R_j . Note that the reduction of the relation R_j by the semijoin $R_i - A \rightarrow R_j$ is proportional to the reduction of $R_j(A)$. The estimation of the size of the relation reduced by a semijoin is thus similar to estimating the reduction of projection on the semijoin attributes. After the semijoin $R_i - A \rightarrow R_j$, the cardinality of R_j can be estimated as $|R_j| \rho_{i,a}$.

2.2 Cost Model for Query Processing in a Mobile Computing System

Table 1 shows the description of parameters in the cost model for a mobile computing system. Energy consumption in data receiving at a mobile computer, denoted by e_{RC} , refers to the energy consumed in receiving data via wireless communication. Energy consumption for receiving a relation R at a mobile computer, denoted by $e_{RC}(R)$, is formulated as $e_r * |R|$ where e_r is the receiving energy coefficient, representing the energy consumed in receiving one tuple of data. Also, energy consumption for sending a relation R out from a mobile computer, denoted by $e_{SD}(R)$, refers to the energy required in transmitting the

Description	Symbol
Processing ratio of server to mobile, i.e., server/mobile	r_{SM}
Energy consumption ratio of data sending to data receiving	r_E
Idling coefficient for a mobile computer	δ
Energy consumption in data receiving at a mobile computer	e_{RC}
Energy consumption in data sending at a mobile computer	e_{SD}
Processing time function at the server	T_S
Amount of data transmission for query processing	d_t
Amount of energy consumption for query processing	E_c

Table 1. Description of symbols for the cost model in a mobile computing system

relation R over wireless link. To capture the asymmetric feature of energy consumption between data sending and receiving of a mobile computer, we use the send-receive energy ratio r_E (i.e., $\frac{e_{SD}}{e_{RC}}$) to represent the ratio of the energy consumption of sending data to that of receiving data. The value of r_E is in general larger than one, and can more explicitly be approximated to a value between 2 and 10 [1]. Hence, $e_{SD}(R)$ is formulated as $r_E * e_r * |R|$ where $e_r * r_E$ is the sending energy coefficient. Similarly to most relevant works [7], the processing time required to perform the given operations on the relation in a server is modeled as a function of the input relations involved. For example, the processing time of joining R_i and R_j in a server, denoted by $T_S(R_i \bowtie R_j)$, can then be expressed by $t_{tuple} * (|R_i| + |R_j| + |R_i \bowtie R_j|)$, where t_{tuple} is the coefficient for the processing time required per tuple.

To capture the asymmetric feature of computing capability between the server and a mobile computer, the server-mobile processing ratio r_{SM} represents the ratio of the processing power of a server to that of a mobile computer. Clearly, the value of r_{SM} is larger than 1 and can be obtained empirically. Hence, the processing time of joining R_i and R_j at a mobile computer can be expressed by $r_{SM} * T_S(R_i \bowtie R_j)$. To reflect the asymmetric feature of energy consumption between activeness and idleness of a mobile computer, we define the idling coefficient δ to be the ratio of the energy consumed by a mobile computer in its idle mode to that in its active mode. The idle coefficient δ of mobile computers can be approximated to a value between 0.02 and 0.5 and is in fact system dependent [7][8]. Consequently, the energy consumption at a mobile computer in its idle mode while a join is performed in a server can be estimated as $\delta * T_S(R_i \bowtie R_j)$, whereas the energy consumed in processing the join between R_i and R_j at a mobile computer is approximated as $\frac{1}{\delta} * r_{SM} * T_S(R_i \bowtie R_j)$. Note that all these parameters can be estimated from the specifications of mobile computers [2][10]. Using the cost model developed, we are able to evaluate the energy consumption and data transmission costs (E_c and d_t , respectively) of the corresponding query processing schemes in a mobile computing system, and develop an efficient solution procedure for multi-join query processing accordingly.

3 Query Processing in a Mobile Computing System

In this section, we consider the processing of multi-join queries that involves one server and many mobile computers. The *destination mobile computer* refers to the mobile computer that issues the query and is expected to receive the query result. A *participating mobile computer* refers to the mobile computer that contains a relation involved in the query processing. Explicitly, we first examine in Section 3.1 a query processing scheme (to be referred to as scheme QP_C). A two-phase query processing scheme which employs a simple join operation (to be referred to as scheme QP_S) is devised in Section 3.2. Next, we devise another two-phase query processing scheme that can determine a join sequence with SI profitable semijoins interleaved (to be referred to as scheme $QP_{S,J}$) to reduce both the amounts of data transmission and energy consumption in Section 3.3.

3.1 Query Processing at the Destination Mobile Computer (denoted by QP_C)

In scheme QP_C , all participating mobile computers and the server send their relations to the destination mobile computer for query processing. Clearly, though minimizing the energy consumption of participating mobile computers, scheme QP_C results in a significant amount of energy consumption at the destination mobile computer (i.e., receiving relations from participating mobile computers and performing join operations). In our experimental studies in Section 4, scheme QP_C will be implemented and evaluated for comparison purposes.

3.2 Query Processing at the Server (denoted by QP_S)

Clearly, despite its simplicity, scheme QP_C does not exploit the asymmetric feature of computing capability between the server and mobile computers, and may thus consume much valuable processing power at the destination mobile computer. Explicitly, since the server is not strictly constrained by energy consumption, one can fully utilize the processing capability of the server. In view of this, we decompose the processing of a query into two phases, namely the *relation transfer phase*, denoted by RT, and the *final phase*, denoted by FP. In the relation transfer phase, each participating mobile computer sends its own relation to the server for a join operation. Thus, the server obtains the resulting relation of the multi-join query among the participating mobile computers. In the *final phase*, the join between the server and the destination mobile computer is performed.

Note that since the server in QP_S takes over the query processing which costs much energy of the destination mobile computer in QP_C , the energy consumption of the destination mobile computer in QP_S is significantly reduced. It can be seen that, in scheme QP_S , each participating mobile computer sends its own relation to the server without considering the use of semijoins. As can be seen in [5], judiciously interleaving a join sequence with semijoins is able to reduce the amount of data transmission required. Note, however, that without

considering the asymmetric features and energy consumption, the algorithm in [5] is not applicable to the query processing in a mobile computing system. As a result, we will devise in the following subsection scheme QP_{SJ} to determine a proper join sequence with semijoins interleaved in the RT phase for further reducing both the amounts of data transmission and energy consumption.

3.3 Query Processing with SI Profitable Semijoins (denoted by QP_{SJ})

In this subsection, we shall first derive a theorem to identify SI profitable semijoins, and then, in light of the theorem derived, develop a solution procedure that can interleave a join sequence with SI profitable semijoins for efficient query processing.

Definition 1: A semijoin $R_i - A \rightarrow R_j$, where the server owns relation R_i and the mobile computer owns relation R_j , is called *SI profitable* if its energy consumption of the mobile computer for performing this semijoin, i.e., $e_{RC}(\rho_{i,a} * |A|) + \frac{1}{\delta} * r_{SM} * T_S(R_j \bowtie (\rho_{i,a} * |A|)) + e_{SD}(\rho_{i,a} |R_j|)$, is smaller than that for sending R_j to the server, i.e., $e_{SD}(|R_j|)$. Note that energy consumption of an SI semijoin by the mobile computer consists of the energy consumed in performing the semijoin and that in sending the resulting relation of that semijoin to the server for a join operation.

With Definition 1, we can derive the following theorem.

Theorem 1: A semijoin $R_i - A \rightarrow R_j$ is *SI profitable* if and only if $\rho_{i,a}$ is less than $\frac{(r_E * e_r - \frac{1}{\delta} * r_{SM} * t_{tuple})}{(r_E * e_r + \frac{1}{\delta} * r_{SM} * t_{tuple})}$.

Theorem 1 leads to the following corollary.

Corollary 1.1. An *SI profitable* semijoin, $R_i - A \rightarrow R_j$, implies that the amount of data transmission by performing this semijoin (i.e., $\rho_{i,a}(|A| + |R_j|)$) is smaller than that of sending R_j to the server (i.e., $|R_j|$).

In the RT phase, by utilizing the server to perform the multi-join query, the server site contains the resulting relation. To facilitate our presentation, use $E_c^{d,j}$ to represent the energy consumption of a join operation between R_d and R_j , where d is the destination site. The energy consumption of an SI profitable semijoin, $R_d - A \rightarrow R_j$, can be expressed as $E_c^{d,j}(SJ)$ and the corresponding data transmission cost can be denoted by $d_t^{d,j}(SJ)$. The energy consumption of a join $R_j \rightarrow R_d$ can be expressed as $E_c^{d,j}(J)$ and the corresponding data transmission cost can be denoted by $d_t^{d,j}(J)$.

First, we develop a directed graph with proper weights in edges. Assume that the destination mobile computer in the query graph (V, E) is denoted by d and the set of edges of node d is represented as E_d . The destination mobile computer and its edges are omitted in the directed graph since they will not be involved in the RT phase. The resulting directed graph is thus $(V-d, E-E_d)$. An edge connecting two nodes n_i and n_j is denoted by (n_i, n_j) , and the weight of edge (n_i, n_j) , denoted by $E_c^{i,j}$, can be set to either the value of $E_c^{i,j}(SJ)$ (meaning that an SI profitable semijoin SJ will be performed) if $\rho_i < \frac{(r_E * e_r - \frac{1}{\delta} * r_{SM} * t_{tuple})}{(r_E * e_r + \frac{1}{\delta} * r_{SM} * t_{tuple})}$, or

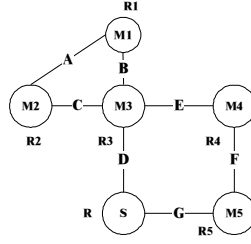


Fig. 1. An illustrative query graph.

the value of $E_c^{i,j}(J)$ (meaning that a simple join J will be performed) otherwise. After constructing the directed graph, algorithm M is applied to determine a join sequence with SI profitable semijoins interleaved in the RT phase. Note that an edge (S, n_j) in a directed graph is being *shrunk* if (S, n_j) is removed from the graph and S and n_j are merged together. When a join operation between the two relations corresponding to nodes S and n_j in a given directed graph is carried out, we can obtain the resulting query graph by shrinking the edges between S and n_j . Algorithm M is outlined below.

Algorithm M: Determine the join sequence with SI profitable semijoins interleaved in the RT phase.

Input: A directed graph $= (V - d, E - E_d)$.

Output: SEQ /* SEQ contains the resulting sequence of joins semijoins */

1. **begin**
2. SEQ = ϕ ;
3. **for** each vertex $w \in V - d$ **do**
4. **begin**
5. $w.mark := false$;
6. $w.ct := \infty$; /* $w.ct$ is the cost of join operation from S to w . */
7. $w.op = J$; /* $w.op$ represents the join operation */
8. **end**
9. $S.ct = 0$; /* The cost of join operation from S to itself is set to zero */
10. **while** \exists an unmarked vertex **do**
11. **begin**
12. let w be an unmarked vertex such that $w.ct$ is the minimal among all the corresponding costs of unmarked vertices;
13. **if** $(w.\rho < \frac{(r_E * e_r - \frac{1}{\delta} * r_{SM} * t_{tuple})}{(r_E * e_r + \frac{1}{\delta} * r_{SM} * t_{tuple})})$
14. $w.op = SJ$;
15. SEQ = SEQ $\cup w$;
16. $w.mark := true$;
17. **for** all edges (w, z) with z is unmarked **do**
18. **begin**
19. **if** $w.ct + weight(w, z) < z.ct$ **then**
20. $z.ct := w.ct + weight(w, z)$; /* Update the weight of the edge (w, z) */
21. **end**
22. **end**
23. **end**

Relation R_i	$ R_i $	Attribute X	Selectivity $\rho_{i,x}$
R ₁	107	A	0.8
		B	0.75
R ₂	102	A	0.85
		C	0.75
R ₃	106	C	0.8
		D	0.7
		E	0.4
R ₄	100	E	0.8
		F	0.95
R ₅	120	F	0.8
		G	0.85
R	131	D	0.9
		G	0.5

Table 2. A profile for query where $|A| = 19, |B| = 15, |C| = 17, |D| = 19, |E| = 16, |F| = 15, |G| = 18, r_{SM} = 5, \delta = 0.5, e_r = 0.1, r_E = 5$ and $t_{tuple} = 0.01$.

Steps	M ₂	M ₃	M ₄	M ₅	Operation included into SEQ
0	∞	53	∞	49.8	ϕ
1	∞	53	99.8	49.8	R-G \rightarrow R ₅ , R ₅ \rightarrow R
2	104	53	88.28	49.8	R ₃ \rightarrow R*
3	104	53	88.28	49.8	R** - E \rightarrow R ₄ , R ₄ \rightarrow R**
4	104	53	88.28	49.8	R ₂ \rightarrow R***

Table 3. An execution example of algorithm M.

To show the execution of algorithm M, consider the query graph in Figure 1 whose profile is given in Table 2. The corresponding directed graph is shown in Figure 2a. It can be verified that since $\rho_{R,G} = 0.5 < \frac{(r_E * e_r - \frac{1}{\delta} * r_{SM} * t_{tuple})}{(r_E * e_r + \frac{1}{\delta} * r_{SM} * t_{tuple})} = 0.6$, the weight of edge from S to M₅ in Figure 2a is $E_c^{S,5}(S,J) = 0.1 * 0.5 * 18 + \frac{1}{0.5} * 5 * 0.01 * (120 + 0.5 * 18 + 0.5 * 120) + 0.1 * 5 * 0.5 * 120 = 49.8$. Also, since $\rho_{R,D} = 0.9 > \frac{(r_E * e_r - \frac{1}{\delta} * r_{SM} * t_{tuple})}{(r_E * e_r + \frac{1}{\delta} * r_{SM} * t_{tuple})} = 0.6$, the weight of edge from S to M₃ is $E_c^{S,3}(J) = 0.1 * 5 * 106 = 53$. Similarly, the weights of other edges in the directed graph in Figure 2a can be obtained. Then, algorithm M is used to generate the proper join sequence with SI profitable semijoins interleaved in the RT phase. First, the costs from S to other vertices are evaluated in the directed graph for initialization (from line 3 to line 8 in algorithm M). The execution scenario of algorithm M is shown in Figure 2 where R*, R** and R*** denote the resulting relations in the end of each step. Note that since the cost of (S, M₅) is the minimal (line 12 in algorithm M), as can be seen in Step 0 from Table 3, the vertex M₅ is selected first. Also, it can be verified that the semijoin operation R-G \rightarrow R₅ is SI profitable (line 13 in algorithm M) since the corresponding selectivity factor

(i.e., 0.5) is smaller than $\frac{(r_E * e_r - \frac{1}{8} * r_{SM} * t_{tuple})}{(r_E * e_r + \frac{1}{8} * r_{SM} * t_{tuple})} = 0.6$. The semijoin $R-G \rightarrow R_5$ is thus performed. In Figure 2a, R_5 is selected for the semijoin and sends the resulting relation to the server. After the join with the relation at site M_5 , the cost from S to M_4 is derived (from line 17 to line 21 in algorithm M). It can be seen that Figure 2a leads to the configuration in Figure 2b in which the weight of the edge from S to M_4 becomes 99.8 (i.e., $49.8+50=99.8$). Then, we shall determine the minimal cost among all the edges connecting to those unvisited vertices (line 12 in algorithm M) and perform the corresponding operations. This procedure repeats until all the vertices of the directed graph are visited (line 10 to in algorithm M). From Table 3, it can be seen that M_3 has the minimal cost in Step 1 and is next be selected. Since the semijoin $R^*-D \rightarrow R_3$ is not SI profitable according to Theorem 1, the simple join from M_3 to S is executed in Figure 2b, leading to the configuration in Figure 2c. Following the same procedure, the sequence of joins and semijoins can be derived. Note that while having the same cost as QP_S in the FP phase, scheme QP_{SJ} outperforms QP_S in the RT phase by incurring not only a smaller amount of energy consumption, i.e., $49.8+53+35.28+51=189.08 < 60+53+50+51=214$, but also a smaller amount of data transmission, i.e., $69+106+46.4+102=323.4 < 120+106+100+102=428$, showing the very advantage of employing proper SI profitable semijoins in scheme QP_{SJ} .

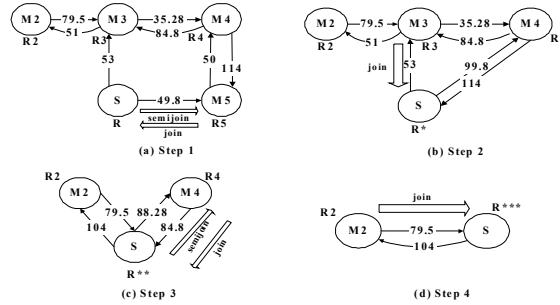
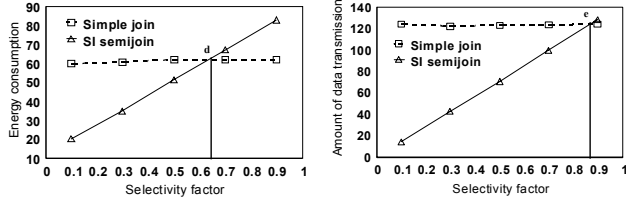


Fig. 2. Execution scenario of algorithm M.

4 Performance Evaluation

Extensive performance studies are conducted in this section. The simulation model built to evaluate the query processing in a mobile computing system is described in Section 4.1. Experimental results of query processing schemes, including those of QP_C , QP_S and QP_{SJ} , are presented in Section 4.2.



(a). The energy consumption of SI semijoin and simple join. (b). The amount of data transmission of SI semijoin and simple join.

Fig. 3. The performance of SI semijoins and simple joins with $r_{SM} = 5$.

4.1 System Model

Simulations were performed to evaluate the effectiveness of query processing schemes. The simulation program was coded in C++, and input queries were generated as follows. 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. Without loss of generality, only queries with connected query graphs were deemed valid and used for our study. Based on the above, the cardinalities of relations and attributes were randomly generated from a uniform distribution within some reasonable ranges. These settings are similar to those prior works in query processing [5]. The values of related parameters employed in a mobile computing system will be given in each experiment separately. The number of relations involved in query processing, denoted by n , is chosen to be 4, 5, 6, 7 and 8, respectively.

4.2 Experimental Results of Query Processing

In this section, we first evaluate the performance of an SI profitable semijoin. Then, performance studies of QP_C , QP_S and QP_{SJ} are conducted.

Experiments of SI Profitable Semijoin In this experiment, we set the value of r_{SM} to 5, the value of δ to be 0.5, the value of t_{tuple} to 0.01, the value of e_r to 0.1, and the value of r_E to 5. The amounts of data transmissions and energy consumptions of an SI (server-initiated) semijoin and a simple join are examined with the selectivity factor varied. According to Definition 1, an SI semijoin is called profitable if the amount of energy consumed is reduced by including this semijoin. From Figure 3a, it can be seen that the amount of energy consumed by an SI profitable semijoin is smaller than that by a simple join when the value of the selectivity factor is smaller than the corresponding value at point d (i.e., 0.63). Thus, the corresponding value at point d can be used as a threshold to identify SI profitable semijoins. Note that according to

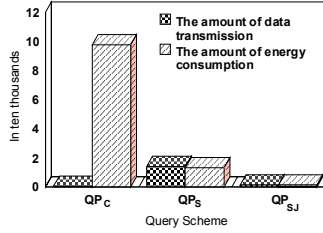


Fig. 4. The performance of QP_C , QP_S and $QP_{S,C}$ when the number of relations is 5.

Theorem 1, the corresponding selectivity factor at point d can be estimated as $\frac{(r_E * e_r - \frac{1}{\delta} * r_{SM} * t_{tuple})}{(r_E * e_r + \frac{1}{\delta} * r_{SM} * t_{tuple})} = \frac{5 * 0.1 - \frac{1}{0.5} * 5 * 0.01}{5 * 0.1 + \frac{1}{0.5} * 5 * 0.01} = 0.667$, which is very close to the value of 0.63 at point d that is empirically determined from Figure 3a. It is shown in Figure 3b the amount of data transmission incurred by an SI profitable semijoin is much smaller than that by a simple join, and the corresponding selectivity factor at point e is larger than that at point d, agreeing with Corollary 1.1.

Performance of QP_S and $QP_{S,J}$ We now examine the performance of query processing among one server and many mobile computers. The number of relations in a query is set to 5, and 300 queries are random generated. For each query, the three query schemes, i.e., QP_C , QP_S and $QP_{S,J}$ are performed. Without loss of generality, we set the value of r_{SM} to 5, the value of δ to 0.5, the value of e_r to 0.1, and the value of r_E to 5. Figure 4 shows the amounts of data transmission and energy consumption incurred by QP_C , QP_S and $QP_{S,J}$.

From Figure 4, it can be seen that scheme QP_C incurs the largest amount of energy consumption among all schemes. Also, as validated by the experimental results, by exploiting the asymmetric feature of computing capability between the server and mobile computers, scheme QP_S can save the energy consumption of the destination mobile computer and participating mobile computers. Furthermore, though reducing the amount of energy consumption, scheme QP_S increases the amount of data transmission required. Note that by employing SI profitable semijoins, scheme $QP_{S,J}$ can further reduce both the amounts of data transmission and energy consumption, showing the very advantage of interleaving a join sequence with SI profitable semijoins in the RT phase.

5 Conclusions

In this paper, we first explored three asymmetric features of a mobile environment. Then, in light of these features, we devised query processing schemes. Explicitly, according to those asymmetric features of a mobile computing system, we devised some specific criteria to identify SI profitable semijoins. We also

proposed and investigated three query processing schemes for the processing of multi-join queries in a mobile computing system. In particular, we formulated the query processing in a mobile computing system as a two-phase query processing procedure that can determine a join sequence and interleave that join sequence with SI profitable semijoins to reduce the corresponding costs. Performance of these query schemes was comparatively analyzed and sensitivity analysis on selectivity factor was conducted. It was shown by our simulation results that by exploiting the three asymmetric features, the proposed query scheme, QP_{SJ} , is very powerful in reducing both the amounts of energy consumption and data transmission incurred.

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-002-007 and NSC 89-2213-E-002-032, 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. Applications of mobile computing. <http://www.nokia.com/3g/index.html>.
3. D. Barbara. Mobile Computing and Databases—A Survey. *IEEE Transactions on Knowledge and Data Engineering*, 11(1):108–117, January/February 1999.
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. 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.
7. R. Jain and N. Krishnakumar. Asymmetric Costs and Dynamic Query Processing in Mobile Computing Environments. In *Proceeding of fifth WIN-LAB Workshop*, April 1995.
8. R. Jain and N. Krishnakumar. *An Asymmetric Cost Model for Query Processing in Mobile Computing Environments*. Wireless Information Networks, J. Holtzman, Kluwer, 1996.
9. J. Jing, A. Helal, and A. Elmagarmid. Client-Server Computing in Mobile Environments. *ACM Computing Surveys*, 31(2):117–157, June 1999.
10. Palm Pilots of 3com. <http://www.3com.com/palm/index.html>.
11. C. Wang and M.-S. Chen. On the Complexity of Distributed Query Optimization. *IEEE Transactions on Knowledge and Data Engineering*, 8(4):650–662, August 1996.