Optimising two-way Joins in Horizontally Partitioned Database Systems

A. SEGEV
School of Business Administration, The University of California, Berkeley, CA 94720, USA

This paper analyses the problem of joining two horizontally partitioned relations in a distributed database system using a semi-join strategy. A mathematical model is developed and the problem is proved to be NP-complete with respect to the number of fragments. Lower bounding and heuristic procedures are proposed, and the results of computational experiments are reported. These results reveal a good performance of the heuristic procedures, and demonstrate the benefit (in terms of communication cost savings) of using semi-join operations to reduce the size of fragments prior to their transmission to the join site.

Received May 1985, revised December 1986

1. INTRODUCTION

In recent years we have been witnessing an accelerated rate of research, development and implementation of distributed computer systems and distributed database management systems (DDBMS). The benefits offered by distributed processing include increased availability and parallelism, reduced system and communication costs, increased end-user involvement, and compatibility, in many cases, with the organisational structure. To realise the benefits offered by distributed processing, numerous problems, technical and managerial, have to be solved. For a discussion of those problems see Refs 25 and 5.

One of the important problems in distributed systems is the efficient processing of queries in a relational DDBMS,[8] where data needed by the query are stored at several sites. Particular attention has been given to the most resource-consuming operation – the relational join. Even with the advances in hardware technology, the importance of optimising the join operations in a distributed query is likely to increase due to the tendency of more geographically dispersed organisations to distribute their data, to store larger amounts of data online, and to expect decision-support capabilities from the DBMS. The problem is how to satisfy ad hoc and routine data retrieval requests in a way that optimises a given performance measure. Models that attempt to characterise and solve this problem have been developed in Refs 2, 4, 6, 10, 19, 27, 28, 32, 33 and others. Models for special cases were introduced in Refs 7, 21 and 15. The results derived by the query optimisation studies differ substantially and depend on the assumptions made, such as the objective function and parameters used by the model. See Ref. 34 for a survey of distributed query optimisation.

The subject of horizontal partitioning (fragmentation) has been discussed in several papers. In this type of partitioning, a relation is fragmented into several sets of tuples (usually disjoint sets), each stored at a different site. Distributed INGRES[29, 30] provides for fragmentation, but the query optimisation algorithm limits fragmentation to a single relation (in the context of increased parallelism). System R*[31] supports fragmentation, but each fragment is treated as a single relation, i.e. the user rather than the system is aware of the fragmentation. A rigorous treatment of fragmentation is given in Ref. 24, but the derived mathematical model represents only a small part of the query-processing problem. An optimisation of set queries in fragmented database systems is presented in Ref. 15. Yu and Chang[33] propose a procedure for processing fragments, and a mathematical model and algorithms for semi-joining fragmented relations without pre-assembly are presented in Ref. 27.

This paper analyses the problem of joining two fragmented relations on a common join attribute so as to minimise the resulting communication costs (this problem will be referred to as the '2-way join problem'). The 2-way join problem is important due to the high frequency of such queries in actual systems, and the need to develop specialised algorithms for the fragmented case. Minimisation of communication cost is a valid objective for systems that use the services of value-added networks (e.g. Tymnet and Telenet), and for systems with a highly congested or slow communication network.

The algorithms proposed in this paper use the semi-join operation as a communication-cost-reduction mechanism. The execution site of a semi-join operation is the site of the fragment being restricted by the operation (we will refer to such a case as a 'local-semi-join strategy'). In Ref. 27 a more general semi-join strategy is analysed. That strategy (referred to as a 'remote-semi-join strategy') permits a semi-join to be executed at an arbitrary site. The details of the remote-semi-join algorithms are given in Ref. 27, where they were shown to outperform (in terms of communication costs) local-semi-join algorithms. Nevertheless, there are situations where a local-semi-join algorithm is preferable; these situations may arise for the following reasons. (1) Remote-semi-join algorithms are complex, harder to implement, and require more computing time than local-semi-join algorithms. (2) As was shown in Ref. 27, the difference in the cost incurred by the two strategies is highly dependent on the selectivity factors. In the case of large selectivity factors, the two strategies are expected to incur the same communication cost, thus favouring a choice of the simpler local-semi-join algorithm. (3) Remote-semi-joins incur a higher overhead cost than local-semi-joins. This higher cost results from the need to coordinate and synchronise a more complex execution strategy. It is important to note that the overhead cost is dependent on the synchronisation procedure and the communication protocol used.

In the remainder of this paper the term 'semi-join' should be understood as a 'local-semi-join'. Following the definition of the 2-way join problem, the problem is shown to be NP-complete (with respect to the number of fragments) and a mathematical model is developed. Lower bounding and heuristic procedures are proposed and the results of computational experiments are reported.

2. RELATIONAL TERMINOLOGY AND PROBLEM DEFINITION

The problem analysed in this paper occurs in relational database systems. The basic relational terminology is described using Fig. 1; more details can be found in Ref. 8 or a basic text such as Ref. 9. A **relation** is a simple file (e.g. R 1 is a part file), and a **database** is a collection of relations. A row of a relation is a **tuple**, and a column is an **attribute** whose values are drawn from a **domain**. **Projecting** a relation on to an attribute results in a set of the attribute's values (duplicates are eliminated), e.g. projecting R 2 on PART # (see Figure 1) results in the set {10, 40}.

R 1	Part #	Colour	Price ($)
	10	Blue	120.00
	20	Black	115.00
	30	Blue	110.00
	40	Red	125.00

R 2	Part #	Supplier #	Qty
	10	5	1000
	10	10	1500
	40	5	1500
	40	20	2000

R 1 semi-joined by R 2:

R 1	Part #	Colour	Price ($)
	10	Blue	120.00
	40	Red	125.00

Figure 1. A semi-join operation.

A **semi-join** operation between relation R 1 and relation R 2 restricts R 1 by values that appear in one of R 1's attributes **and** in one of R 2's attributes (those attributes are the **join attributes**). The semi-join operation has been established as a useful mechanism to reduce the amount of transmitted data, when processing queries in distributed databases.[2, 3, 32] Theoretical work on the semi-join operation can be found in Ref. 4. Fig. 1 shows two sample relations R 1 and R 2, and the effect of semi-joining R 1 by R 2 over the join attributes R 1.PART # and R 2.PART #. Note that the semi-join is an asymmetric operator; semi-joining R 2 by R 1 would have no effect on the size of R 2 in the example of Fig. 1. If relation R 1 is semi-joined by relation R 2 we will refer to R 1 as the '**restricted relation**' and to R 2 as the '**restricting relation**'.

A database is **fragmented** if one or more of its relations are partitioned horizontally. Fig. 2 shows a fragmented database. The figure contains the same relations as Fig.

Site 1

R 1	Part #	Colour	Price
	10	Blue	120.00
	20	Black	115.00
	30	Blue	110.00
	40	Red	125.00

Site 2

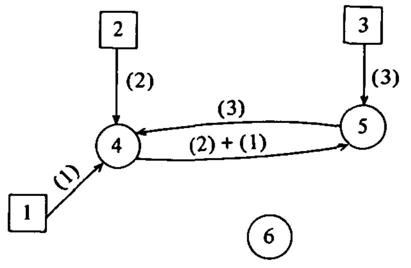
$R2_1$	Part #	Supplier #	Qty
	10	5	1000
	10	10	1500

Site 3

$R2_2$	Part #	Supplier #	Qty
	40	5	1500
	40	20	2000

Figure 2. A fragmented database.

1, but relation R 2 has been partitioned into two fragments, $R2_1$ and $R2_2$, which are stored at site 2 and site 3 respectively. Suppose that R 1 is to be semi-joined by R 2 over PART #. It can be done by performing two semi-joins, one between R 1 and $R2_1$ and the other between R 1 and $R2_2$. Note, however that the resulting tuples of the two semi-join operations have to be unioned. Moreover, **every** fragment of the restricting relation must participate in a semi-join of the restricted relation (or fragment); otherwise no tuple of the restricted relation (or fragment) can be eliminated.



$\boxed{i}$ Fragment i (relation R 1)

ⓘ Fragment i (relation R 2)

$\underrightarrow{(i)}$ Transmission of join attribute i

Figure 3. Graph representation of transfers related to fragment i.

A possible solution to the 2-way join problem, e.g. Ref. 10, is to assemble one relation and duplicate it at every fragment site of the other relation, and then carry out the joins in parallel. For a large number of fragments, however, this strategy may incur very high communication costs. The strategy pursued in this paper is to use semi-join operations to reduce the size of fragments prior to a final join at the query site.

Fig. 3 shows a sample semi-join strategy. The fragments were numbered sequentially such that a fragment number is identical to the site number at which it is stored. Fragments 1, 2 and 3 belong to relation R 1, and fragments 4, 5 and 6 belong to relation R 2. Fig. 3 represents the case where a decision was made to restrict fragments 4 and 5 by semi-join operations. As stated earlier, a fragment chosen to be restricted must be semi-joined by every fragment of the other relation. That is why, in Fig. 3, join attributes 1, 2 and 3 are transmitted to sites 4 and 5. It should be emphasised that the transmissions shown in Fig. 3 are not necessarily optimal in this example, but are rather used to illustrate the type of transmissions that will be used to execute a semi-join strategy.

The following assumptions will underlie the models developed in this paper.

Assumption 1
The routing of messages is transparent to the model. If a message has to be sent from site a to site b, the model is concerned with the cost of the associated transmission and not its routing.

Assumption 2
A non-redundant materialisation of fragments is given.

Assumption 3
If a fragment is to be restricted, it will be semi-joined at this site. The advantage of this assumption is in significantly reducing the complexity of the resulting algorithms, and the actual execution of the semi-join strategy. This assumption implies that the transmission of a single join attribute constitutes a minimum spanning tree (see Fig. 3).

3. MATHEMATICAL NOTATION

Let the two relations of the 2-way join problem be R 1 and R 2. The following notation will be used henceforth:

S 1 The index set of sites among which relation R 1 is fragmented.
S 2 The index set of sites among which relation R 2 is fragmented.
T S 1 $\cup$ S 2

$$\Gamma_i \begin{cases} S2, & i \in S1 \\ S1, & i \in S2 \end{cases}$$

q The index of the query site
C_{ij} The unit transmission cost between site i and site j
F_i Size of the fragment stored at site i
D_i Size of the projection of fragment i over its joint attribute
α_i The selectivity factor for a semi-join of fragment i, i.e. the size of the result is $F_i \alpha_i$.

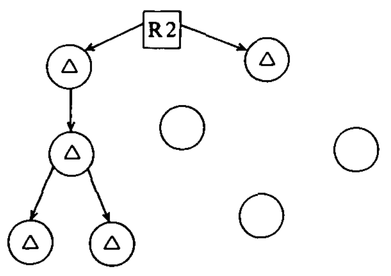
Note. There is a one-to-one correspondence between a site number, fragment number and join attribute number e.g. fragment 5 is stored at site 5 and the number of its join attribute is 5.

The plus-and-minus signs, in notation of the type '$k \in S1 \pm i$', will be used to represent the union of set S 1 and the singleton i, and the removal of element i from the set i, respectively. If S is a set, |S| is its cardinality.

Additional notation and definitions will be introduced as necessary.

4. THE COMPLEXITY OF THE 2-WAY JOIN PROBLEM

Although the 2-way join problem involves only two relations it is a hard problem. To demonstrate the complexity of the problem, consider the following special case. Two relations R 1 and R 2, are to be joined, one fragmented and the other non-fragmented (assume that R 2 is not fragmented, thus S 2 = {1}). Let us assume that a decision has to be made on a restriction of each fragment of R 1 by a semi-join. A feasible solution to this problem (referred to as problem P_8) is illustrated in Fig. 4. The join attribute of the non-fragmented relation R 2 spans all the fragments to be semi-joined. The arcs representing final transfers to the query site were omitted. Letting



$\boxed{R2}$ The non-fragmented relation
(△) Fragments of R 1 to be semi-joined
○ Other fragments of R 1
⟶ Transmission of R 2's join attribute

Figure 4. Graph representation of a solution to problem P_s

$$X_i = \begin{cases} 1, & \textit{if fragment i is restricted by a semi-join} \\ 0, & \textit{otherwise} \end{cases}$$

$$W_{ij} = \begin{cases} 1, & \textit{if R2's join attribute is sent from site i to site j} \\ 0, & \textit{otherwise} \end{cases}$$

f_{ij} = *flow variables for arc* (i, j)

this special case can be formulated as the following linear integer program:

Problem P_8

$$\min\{ \sum_{i \in S1} (F_i \alpha_i X_i + (1 - X_i) F_i) C_{iq} + \sum_{i \in S1+(1)} \sum_{j \in S1} D_i C_{ij} W_{ij} \} \quad (1)$$

subject to:

$$\sum_{i \in S1+(1)} W_{ij} = X_j, \quad j \in S1 \quad (2)$$

$$\sum_{i \in S1+(1)} f_{ij} - \sum_{i \in S1} f_{ji} = X_j, \quad j \in S1 \quad (3)$$

$$f_{ij} \leqslant |S1| \, W_{ij}, \quad i \in S1 + \{1\}, j \in S1 \quad (4)$$

$$W_{ij}, X_j \in \{0, 1\}, f_{ij} \geqslant 0, \quad i \in S1 + \{1\}, j \in S1 \quad (5)$$

The first term in the objective function (1) accounts for the cost of transmitting fragments to the query site, and the second term accounts for the cost of transmitting R 1's join attribute to semi-join sites. Constraints (2) state that if a fragment is restricted by a semi-join, R 2's join attribute will be transmitted to that fragment's site from one and only one site (more than a single transfer is superfluous). The constraints in (3) and (4) are the flow constraints that prevent cycles (see Ref. 14) in the graph representing the transmissions of R 2's join attribute.

Theorem 1: Problem P_8 is NP-complete.

Proof: Rewrite the objective function (1) as:

$$\min\{\sum_{i\in S1} F_i C_{iq} + \sum_{i\in S1} \bar{C}_i X_i + \sum_{i\in S1+\{1\}} \sum_{j\in S1} \tilde{C}_{ij} W_{ij}\} \quad (6)$$

Where $\bar{C}_i = (\alpha_i - 1) C_{iq} F_i$ is the cost of adding node i to the spanning tree, and $\tilde{C}_{ij} = D_i C_{ij}$ is the cost of including edge (i, j) in the tree. Note that $\bar{C}_i \leqslant 0$, $\forall$ i $\in$ S1.

It will be shown that if problem P_s can be solved in polynomially bounded time, so can the problem of the Steiner Tree in graphs, which is known to be NP-complete.[12]

Consider a graph G(V, E), a set of nodes N $\subseteq$ V, and an edge cost function: $Cost(e_{ij}\in E) = \tilde{C}_{ij}$. The Steiner Tree problem is to find a tree sub-graph G(V′, E′) with a minimum sum of edge costs, such that V′ $\subseteq$ V, E′ $\subseteq$ E, and N $\subseteq$ V′. Let

$V_0(Steiner) = \{i \mid X_i = 1$ *in an optimal solution to the Steiner problem*$\}$

$V_0(P_s) = \{i \mid X_i = 1$ *in an optimal solution to problem* $P_s\}$

We transform an arbitrary instance of the Steiner Tree problem into problem P_s by assigning the following node weights

$$\bar{C}_i = \begin{cases} M, & i\in N \\ 0, & i\in V-N \end{cases}$$

where M is a large negative number. Clearly, if M is large enough, X_i will be 1, $\forall$ j $\in$ N. Moreover, X_i will be 1 for i $\in$ V $-$ N only if that reduces the cost of spanning the nodes in N. Hence, V_0(Steiner) $= V_0(P_s)$. To obtain the optimal cost of the Steiner Tree problem $M\sum_{i\in S1} X_i$ should be subtracted from the optimal cost of P_s.

It is easy to see that problem P_s belongs to the class NP (the corresponding decision problem can be solved by a polynomial non-deterministic algorithm) and the Theorem follows. Q.E.D.

5. A MATHEMATICAL MODEL OF THE 2-WAY JOIN PROBLEM

Two types of mathematical formulations of the 2-way join problem are presented in this section. The first is a tree-based formulation, whose main utilisation will be in the development of a heuristic procedure. A flow-based formulation will be used to derive a lower bound on the value of an optimal solution to the 2-way join problem.

5.1. A tree-based formulation

We define the following decision variables to be used in the formulation of problem P_{T1} below.

$$X_i = \begin{cases} 1, & \textit{if fragment i is restricted by a semi-join} \\ 0, & \textit{otherwise} \end{cases}$$

$$W_{ikt} = \begin{cases} 1, & \textit{if join attribute i is transferred from site k to site t} \\ 0, & \textit{otherwise} \end{cases}$$

g_{ikt} = *amount of flow on arc* (i, k, t)

Problem P_{T1}

$$\min\{\sum_{i\in T} F_i C_{iq}(1-(1-\alpha_i)X_i) + \sum_{i\in T} \sum_{k\in\Gamma_i+i} \sum_{t\in\Gamma_i} D_i C_{kt} W_{ikt}\} \quad (7)$$

subject to:

$$\sum_{k\in\Gamma_t+i} W_{ikt} = X_t, \quad i\in T, t\in\Gamma_i \quad (8)$$

$$\sum_{k\in\Gamma_t+i} g_{ikt} - \sum_{k\in\Gamma_i} g_{itk} = X_t, \quad i\in T, t\in\Gamma_i \quad (9)$$

$$g_{ikt} \leqslant W_{ikt}|\Gamma_i|, \quad i\in T, k\in\Gamma_i+i, t\in\Gamma_i \quad (10)$$

$$g_{ikk} = W_{ikk} = 0, \quad i\in T, k\in\Gamma_i+i \quad (11)$$

$$X_i, W_{ikt}\in\{0,1\}, g_{ikt}\geqslant 0, \quad i\in T, k\in\Gamma_i+i, t\in\Gamma_i \quad (12)$$

The first sum in the objective function (7) accounts for the cost of transmitting fragments, possibly unrestricted, to the query site. The triple summation in (7) accounts for the cost of transferring join attributes to semi-join sites. Constraints (8) say that if a fragment is to be restricted by a semi-join, all the necessary join attributes must be transmitted to the semi-join site. The flow constraints (9) to (11) prevent cycles and (together with (8)) assure a tree structure for the transfers of each join attribute. Constraints (12) are the binary and non-negativity constraints.

The above formulation is separable into two symmetric sub-problems P_{T11} and P_{T12}. The formulation for problem P_{T11} is given below. To get the formulation for problem P_{T12}, simply replace every S1 by S2 and S2 by S1 in problem P_{T11}. The above separability means that the decisions on the restriction of fragments of one relation are independent of the corresponding decisions for the other relation.

Problem P_{T11}

$$\min\{\sum_{i\in S1} F_i C_{iq}(1-(1-\alpha_i)X_i) + \sum_{i\in S2} \sum_{k\in S1+i} \sum_{t\in S1} D_i C_{kt} W_{ikt}\} \quad (13)$$

subject to:

$$\sum_{k\in S1+i} W_{ikt} = X_t, \quad i\in S2, t\in S1 \quad (14)$$

$$\sum_{k\in S1+i} g_{ikt} - \sum_{k\in S1} g_{itk} = X_t, \quad i\in S2, t\in S1 \quad (15)$$

$$g_{ikt} \leqslant W_{ikt}|\Gamma_i|, \quad i\in S2, k\in S1+i, t\in S1 \quad (16)$$

$$g_{ikk} = W_{ikk} = 0, \quad i\in S2, k\in S1+i \quad (17)$$

$$X_i, W_{ikt}\in\{0,1\}, g_{ikt}\geqslant 0, \quad i\in S2, k\in S1+i, t\in S1 \quad (18)$$

5.2. A flow-based formulation

The basic idea underlying the flow-based formulation is to define, for each fragment i and fragment j $\in \Gamma_i$, a path from node j to node i, with at most one unit of flow of commodity j (representing the transmission of join attribute j to a semi-join site). Let us denote that path as *path* (j, i) and define the flow variables as follows:

f_{jikt} = *flow of commodity j on arc* $(k, t) \in$ *path* (j, i).

The flow-based formulation is given as problem P_{T21} below. To get the symmetric problem P_{T22}, replace S1 by S2 and S2 by S1 in problem P_{T21}.

Problem P_{T21}

$$\min\{\sum_{i\in S1} F_i C_{iq}(1-(1-\alpha_i)X_i) + \sum_{i\in S2} \sum_{k\in S1+i} \sum_{t\in S1} D_i C_{kt} W_{ikt}\} \quad (19)$$

subject to:

$$\sum_{t \in S1} f_{jtkt} - \sum_{t \in S1+j} f_{jttk} = 0, \quad k \in S1, j \in S2, i \in S1 \quad (20)$$

$$\sum_{t \in S1} f_{jtkt} - \sum_{t \in S1+j} f_{jttk} = X_i, \quad k = j, j \in S2, i \in S1 \quad (21)$$

$$\sum_{t \in S1} f_{jtkt} - \sum_{t \in S1+j} f_{jttk} = -X_i, \quad k = i, j \in S2, i \in S1 \quad (22)$$

$$f_{jtkt} \leqslant 1, \quad j \in S2, i \in S1, k \in S1+j, t \in S1 \quad (23)$$

$$W_{jtk} \geqslant f_{jttk}, \quad j \in S2, i \in S1, t \in S1+j, k \in S1 \quad (24)$$

$$g_{tkk} = W_{ikk} = 0, \quad i \in S2, k \in S1+i \quad (25)$$

$$X_i,\ W_{jkt} \in \{0, 1\}, f_{jtkt} \geqslant 0, \quad i \in S1, j \in S2, k \in S1+j, t \in S1 \quad (26)$$

The formulation of Problem $P_{T_{21}}$ is valid, since the only feasibility constraint is that if fragment i is to be restricted (i.e. $X_i = 1$), then for each fragment $j \in \Gamma_i$ there must be a path from site j to site i, representing the transmission of join attribute j, and this is guaranteed by constraints (20) to (23). Constraints (24) assure that the cost of transferring join attribute j from site t to site k will be taken into account at most once.

5.3. Equal unit transmission costs

Most of the studies in the area of Distributed Query Optimisation assume equal unit transmission costs ($C_{ij} = C$, $\forall$ i, j $\in$ t), e.g. Refs 1–3, 10, 19 and 28. The 2-way join problem can easily be solved when equal unit transmission costs are assumed. In this case the routeing problem is eliminated, and the only decision to be made is whether or not to restrict fragments by semi-join operations. Algorithm 1 derives the solution to the problem.

Algorithm 1

```
1. For every $i \in T$ Do:
   (a) Let $COST1 = F_i C_{iq} = F_i C$
       $COST2 = \sum_{j \in \Gamma_i} D_j C_{ji} + F_i \alpha_i C_{iq}$
             $= \sum_{j \in \Gamma_i} D_j C + F_i \alpha_i C$
       If $COST1 < COST2$ transfer fragment i to the
       query site without a semi-join. Otherwise, transfer
       the $|\Gamma_i|$ join attributes directly to site i, and semi-join
       fragment i there.
```

6. LOWER BOUNDING PROCEDURES

As was proved in Section 4, the 2-way join problem is NP-complete. It is therefore unlikely that an optimal algorithm will be used for real-time query optimisation. Even in the case of query compilation, the problem's complexity will make an optimal algorithm too expensive. Consequently, it is expected that heuristic algorithms will be used for most real-world applications. A lower bound, when compared to the value of heuristic solutions, provides information about the quality of the solutions generated by the heuristic algorithms. For example, if a heuristic's solution and the lower bound are 1% apart, we know that the heuristic's value is at most 1% away from the optimum, and there is probably no sense in devoting extensive research effort to devise better algorithms for the same model. In the absence of lower bounds for the problem, the only way to compare the heuristics' solutions to the optimal solution is by complete enumeration of the solution space. Conducting a significant empirical analysis using complete enumeration for sizeable problems is likely to be too costly for most researchers. Lower bounds are also needed if one wants to compute the optimal solution using a branch and bound algorithm.[20]

The lower bounds derived in this paper are based on a Lagrangian relaxation coupled with a subgradient procedure (a description of the procedure is given below). Lagrangian relaxation[16] has been applied successfully in many combinatorial optimisation problems such as location problems,[11, 17] the travelling salesman and related problems,[13, 18] and the design of computer networks.[14]

A summarised description of the Lagrangian relaxation follows. Interested readers can consult Ref. 16 for more details about the theory. The application of the procedure to the 2-way join problem is detailed in the appendix.

The general integer linear programming problem can be written as:

$$(P)\ \underset{X \geqslant 0}{Min}\ \mathbf{CX}$$

subject to:

$$\mathbf{AX} \geqslant \mathbf{b}$$

$$\mathbf{BX} \geqslant \mathbf{d}$$

$$X_i \text{ integer}, i \in I$$

Where **X, b, c** and **d** are vectors, **A** and **B** are matrices of conformable dimensions, and the index set I denotes the variables required to be integer. It is assumed that the constraints $\mathbf{BX} \geqslant \mathbf{d}$ have a special structure. We define the Lagrangian relaxation of problem (P) relative to $\mathbf{AX} \geqslant \mathbf{b}$ and a conformable non-negative vector λ to be:

$$(PR_\lambda)\ \underset{\mathbf{X} \geqslant 0}{Min}\ \mathbf{CX} + \lambda(\mathbf{b} - \mathbf{AX})$$

subject to:

$$\mathbf{BX} \geqslant \mathbf{d}$$

$$X_i \text{ integer}, i \in I$$

Denoting the value of an optimal solution to problem (P) by Z(P), we have the following relation: $Z(PR_\lambda) \geqslant Z(P)$. Consequently the Lagrangian relaxation of (P) constitutes a lower bound on the optimal value of (P). For the relaxation to be useful, problem (PR_λ) has to be significantly easier to solve than problem (P). For a given Lagrangian relaxation, the tightest lower bound which can be achieved is $\max_\lambda\{Z(PR_\lambda)\}$. The vector λ which achieves that maximisation is the best set of Lagrangian multipliers.

The subgradient procedure is used to approximate the best Lagrangian multipliers. It is an iterative procedure which updates the vector λ at each iteration based on the value $Z(PR_\lambda)$. The main steps of the subgradient procedure are described below. For a detailed description of the procedure and its theory see Refs 18, 26. The procedure starts with an initial vector λ^0 and the optimal solution $\mathbf{X}^0$ to problem (PR_{λ^0}). For each iteration t, the subgradient direction vector is calculated as $\mathbf{\Phi}^t = \mathbf{b} -$

AX^t. Using a step size S^t, the new vector λ^{t+1} is given by $\lambda^{t+1} = \lambda^t + S^t\Phi^t$. A common step size is

$$S^t = C^t \frac{\bar{Z}(P) - Z(PR_{\lambda^t})}{\|\Phi^t\|^2},$$

where $\bar{Z}(P)$ is an upper bound (feasible solution) on the value of (P, $\|\,.\,\|$ denotes the norm function, and C^t is a scalar which is initially equal to 2 and is updated every k iterations.

Two lower bounds on the value of an optimal solution to the 2-way join problem were derived through Lagrangian relaxations of problems P_{T1} and P_{T2}. The mathematical details of the relaxations are given in Appendix 1.

7. A HEURISTIC PROCEDURE BASED ON PROBLEM P_{T1}

The heuristic proposed in this section is a set selection procedure, where the set to be selected consists of the variables X_i that assume the value 1 in problem P_{T11} (the same procedure is also applied to problem P_{T12}). Given a feasible solution to problem P_{T11}, we define the set $S = \{i \mid X_i = 1, i \in S1\}$ associated with that solution. Also, let Z(S) denote the value of an optimal solution to problem P_{T11} for a given set S. The value of Z(S) is obtained by substituting the X_i values in problem P_{T11}, and solving the resulting problem which is formulated below.

$$Z(S) = \min\{ \sum_{i \in S1-S} F_i C_{iq} + \sum_{i \in S} F_i \alpha_i C_{iq} + \sum_{i \in S2} \sum_{k \in S1+i} \sum_{t \in S1} D_i C_{kt} W_{ikt}\} \quad (27)$$

subject to:

$$\sum_{k \in S1+i} W_{ikt} = 0, \quad i \in S2, t \in S1-S \quad (28)$$

$$\sum_{k \in S1+i} W_{ikt} = 1, \quad i \in S2, t \in S \quad (29)$$

$$\sum_{k \in S1+i} g_{ikt} - \sum_{k \in S1} g_{itk} = 0, \quad i \in S2, t \in S1-S \quad (30)$$

$$\sum_{k \in S1+i} g_{ikt} - \sum_{k \in s1} g_{itk} = 1, \quad i \in S2, t \in S \quad (31)$$

$$g_{ikt} \leqslant W_{ikt}|S1|, \quad i \in S2, k \in S1+i, t \in S1 \quad (32)$$

$$g_{ikk} = W_{ikk} = 0, \quad i \in S2, k \in S1+i \quad (33)$$

$$X_t, W_{ikt} \in \{0, 1\}, g_{ikt} \geqslant 0, \quad i \in S2, k \in S1+i, t \in S1 \quad (34)$$

The above problem of finding Z(S) is separable over the index $i \in S2$ into |S2| Minimum Spanning Tree problems. Defining $\beta_{ikt} = D_i C_{kt}$, each of the |S2| sub-problems is given by:

$$Z_i(S) = \min\{ \sum_{k \in S+i} \sum_{t \in S} \beta_{ikt} W_{ikt}\} \quad (35)$$

subject to:

$$\sum_{k \in S+i} W_{ikt} = 1, \quad t \in S \quad (36)$$

$$\sum_{k \in S+i} g_{ikt} - \sum_{k \in S} g_{itk} = 1, \quad t \in S \quad (37)$$

$$g_{ikt} \leqslant W_{ikt}|S|, \quad k \in S+i, t \in S \quad (38)$$

$$g_{ikk} = W_{ikk} = 0, \quad k \in S1+i \quad (39)$$

$$W_{ikt} \in \{0, 1\}, g_{itk} \geqslant 0, \quad k \in S+i, t \in S \quad (40)$$

After solving the Minimum Spanning Tree problems, Z(S) is given by:

$$Z(S) = \sum_{i \in S1-S} F_i C_{iq} + \sum_{i \in S} F_i \alpha_i C_{iq} + \sum_{i \in S2} Z_i(S) \quad (41)$$

A greedy[20] heuristic based on the foregoing analysis is formally stated as Algorithm H 1 below.

Algorithm H 1 (ADD heuristic)

```
1. Let $S = \varnothing, \bar{S} = S1 - S, FC = \sum_{i \in S1} F_i C_{iq}$.
2. For every $i \in \bar{S}$, evaluate $\hat{C}_i = Z(S+i)$ as in (41).
3. Let $\bar{i}$ be the index that achieves $\min_{i \in \bar{S}}\{\hat{C}_i\}$.
4. If $\hat{C}_{\bar{i}} \geqslant FC$ Goto Step 6. Else,
       $FC = \hat{C}_{\bar{i}}, S = S + \bar{i}, \bar{S} = \bar{S} - \bar{i}$.
5. If $\bar{S} \neq \varnothing$ Goto Step 2.
6. $Z(S) = FC$; stop.
```

Note that when Algorithm H 1 terminates, the first three elements of S are optimal, i.e., they will be selected by any algorithm that generates optimal solutions to problem P_{T11}. However, Algorithm H 1 may terminate with $S = \varnothing$, while the optimal solution is to select all the nodes. This is illustrated by example 1.

Table 1. Data for Example 1

		$\tilde{C}_{ij}$ j				
$\bar{C}_i$	i	n	1	2	3	4
—	n	∞	3	6	3	4
2	1	3	∞	3	3	2
5	2	6	3	∞	4	2
3	3	3	3	4	∞	2
3	4	4	2	2	2	∞

Example 1

Consider problem P_s of Section 4. The non-fragmented relation is stored at site n, and the fragments of the second relation at sites 1 to 4, $\bar{C}_i$ and $\tilde{C}_{ij}$ are as defined in Theorem 1 of Section 4. The data for the example are given in Table 1. We ignore the constant term in equation (6), i.e. $\sum_{i \in S1} F_i C_{iq} = 0$. Applying Algorithm H 1 to the example results in the following:

Step 1: $S = \varnothing$, $\bar{S} = \{1, 2, 3, 4\}$, $FC = 0$.
Step 2: $\hat{C}_1 = 1$, $\hat{C}_2 = 1$, $\hat{C}_3 = 0$, $\hat{C}_4 = 1$.
Step 3: $\bar{i} = 3$.
Step 4: $\hat{C}_3 = FC$; Z(S) = 0.

It is easy to see by complete enumeration that S = {1, 2, 3, 4} is an optimal solution with Z(S) = −4.

8. COMPUTATIONAL RESULTS

Two lower bounding procedures and an ADD heuristic have been proposed in this paper. Those procedures were programmed in FORTRAN and comparative computational experiments have been carried out. The results of those experiments are summarised in Table 2.

The experiments were based on the following data: the fragment size was drawn from a uniform distribution with a range of [10, 20]; the size of the join attribute

Table 2. Performance of the ADD procedure relative to the lower bounds

		ADD/PL1			ADD/PL2			
|T|	F1, F2	Av.	Min.	Max.	Av.	Min.	Max.	CPU
5	2, 3	1.03	1.00	1.07	1.03	1.00	1.07	0
10	2, 8	1.06	1.00	1.14	1.06	1.00	1.12	0
	5, 5	1.02	1.00	1.05	1.04	1.00	1.10	0
15	2, 13	1.04	1.00	1.10	1.05	1.00	1.12	0.01
	7, 8	1.02	1.00	1.08	1.03	1.00	1.09	0.04
20	5, 15	1.11	1.00	1.30	1.06	1.00	1.13	0.05
	10, 10	1.06	1.02	1.12	1.04	1.00	1.09	0.09
30	5, 25	1.23	1.03	1.40	1.10	1.00	1.20	0.17
	15, 15	1.18	1.05	1.42	1.05	1.00	1.12	0.21
40	5, 35	1.20	1.04	1.40	1.06	1.00	1.10	0.57
	20, 20	1.20	1.07	1.35	1.02	1.00	1.07	0.63

(1) |T| is the total number of fragments, F1 and F2 are number of fragments of relation 1 and 2 respectively.

(2) ADD is the value of the solution generated by the ADD procedure.

(3) PL1 and PL2 are the values of the lower bounds generated by applying the subgradient procedure to problems P_{L1} and P_{L2}.

(4) CPU is the average CPU time measured in IBM 3032 seconds.

ranged from 5% to 50% of the fragment size; the unit transmission costs were uniformly distributed between 0 and 5; and the selectivity factors were drawn from a uniform distribution with a range of [0, 1].

The results presented in Table 2 are the average value of six problems. Table 2 shows that relaxation PL2 (derived from the flow-based formulation) produced significantly tighter bounds than relaxation PL1 (derived from the tree-based formulation), and the gap between the ADD heuristic and the lower bounds is smaller when the relations consist of about the same number of fragments. The gap between the ADD procedure and the optimal solution is on the average at most 4.9% and for 70% of the problems the gap is at most 5%.

An additional set of computational experiments was carried out to test the effectiveness of the semi-join strategy as a function of the selectivity factors and the size of join attributes. The effectiveness of a semi-join strategy was evaluated by comparing its cost to the cost of a direct join (without a restriction by semi-join operations) at the query site. Two of the model's parameters were varied: the selectivity factors, and the size of join attributes.

The values of the selectivity factors are likely to have a significant effect on the cost of solutions generated by the ADD heuristic. It is likely that, for large selectivity factors, the semi-join operations will not significantly reduce the size of fragments, leading to solutions with costs close to the cost of the direct-semi-join strategy. The communication costs of semi-join operations are incurred by transmitting join attributes. Hence it is likely that small (relative to the fragment size) join attributes

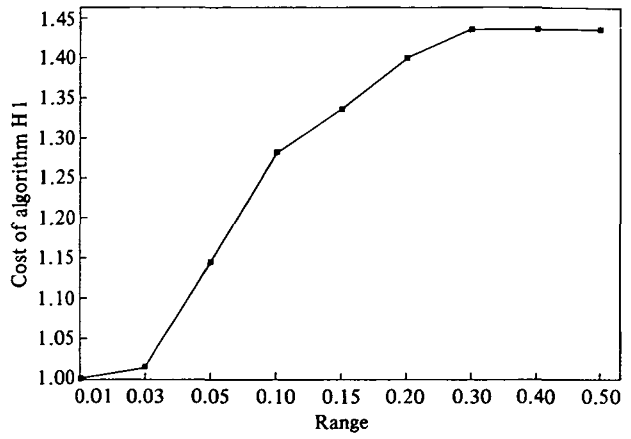


Figure 5. Effect of selectivity factors.

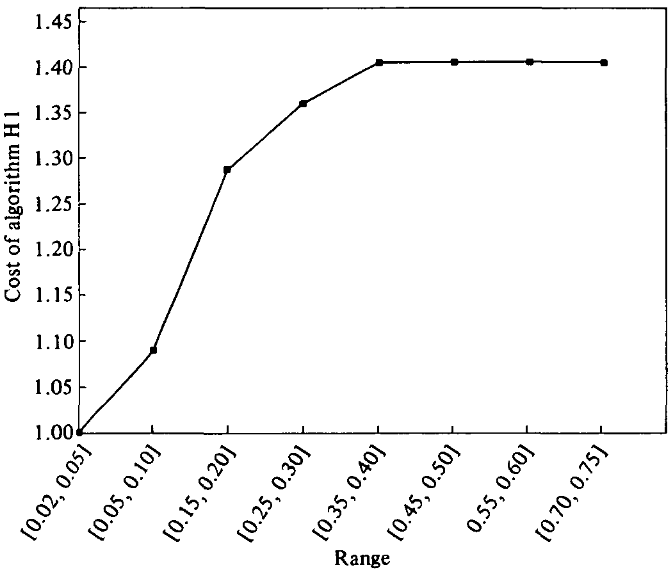


Figure 6. Effect of join attribute size.

will cause the heuristic procedure to generate solutions which include more semi-join operations than solutions for the case of large join attributes. Consequently, one would expect that the cost of solutions in the case of large join attributes will be close to the cost of the direct-semi-join strategy.

The input data that were fixed for the sensitivity analysis experiments consisted of the number of fragments of the first relation (|S1| = 15), the number of fragments of the second relation (|S2| = 10), and the unit transmission costs ($C_{ij} \sim U(0, 5)$). The effect of changes in the values of the selectivity factors α_i is shown in Fig. 5. Every point on the graph in Fig. 5. is the average cost of nine sample problems. All costs were normalised by dividing them by the value of the minimum average cost. The selectivity factors were derived by generating $\alpha_i \sim U(0, b)$, where b is the range's upper limit as specified on the horizontal axis of the graph in Fig. 5. The fragment size F_i was drawn from a uniform distribution with a range of [10, 20], and the size of the join attribute D_i was derived by generating $d_i \sim U(0.02, 0.20)$ and multiplying it by F_i.

The results of the experiments validated the intuitive conjecture: for large values of the selectivity factors ($\alpha_i \in [0.b]$, b $\geqslant$ 0.3) the values of solutions, generated by the heuristic procedure, converged to the value of no-semi-join solution, i.e. performing a direct join at the query site. Fig. 5 demonstrates the effectiveness of semi-join operations when the selectivity factors are small (up to about 50% difference in cost occurred in the experiments).

The effect of the size of the join attribute relative to its fragment's size is shown in Fig. 6. The input data to the experiments were the same as for the case of Fig. 5, except that the selectivity factors were drawn from a fixed range ($\alpha_i \sim U(0, 0.05)$), and the size of join attributes (as a fraction of the fragments' size) was drawn from a uniform distribution with a range of [a, b], where a and b are as specified on the horizontal axis of the graph in Fig. 6. The behaviour of the heuristic, as exhibited by the graphs in Fig. 6, is very similar to the case where the selectivity factors were varied. For large size of the join attributes (even for small size of the selectivity factors), semi-join operations are not beneficial, and the values of heuristic solutions converge to the value of the no-semi-join solution.

9. SUMMARY AND FUTURE RESEARCH

This paper has analysed the 2-way join problem, and proved its NP-completeness. Using the semi-join operation as a size-reduction operation, a mathematical model has been developed. Lower bounding and heuristic procedures have been proposed and the results of computational experiments presented. Those experiments have revealed good performance by the heuristic procedure, and demonstrated the benefit of using semi-join operations to reduce the size of fragments prior to their transfer to the query site. It was evident from the computational experiments that the benefit of the semi-join strategy is strongly dependent on the size of selectivity factors and join attributes. The experiments showed that large selectivity factors and/or join attributes make the semi-join strategy non-beneficial. In an actual environment, one may experiment with semi-join procedures and determine a threshold point above which semi-join operations will not be performed.

The importance of fragmentation has been stressed in Section 1. Given that type of data topology, 2-way joins are an important component of the system, since it is reasonable to assume that many of the queries will refer to two relations. The number of fragments in a fragmented database system could be large for a geographically dispersed organisation. In that case, the problem's complexity will render the use of optimal procedures too costly, and good heuristic procedures will have to be employed. The lower bounds developed in this paper can provide a system designer with valuable help in evaluating the performance of a particular algorithm.

The algorithms proposed in this paper follow the local-semi-join strategy, where the execution site of a semi-join operation is the site of the fragment being restricted by the operation. In Ref. 27 a more general semi-join strategy is analysed. The strategy (referred to as a 'remote-semi-join strategy') permits a semi-join to be executed at an arbitrary site. As discussed in Section 1, there are cases where the less sophisticated local-semi-joins are more useful. An interesting research project is to model the overhead costs associated with a particular query-optimisation algorithm.

REFERENCES

1. P. M. G. Apers. Query processing and data allocation in distributed database systems. *Ph.D. thesis*, Vrije University, Amsterdam (1982).
2. P. M. G. Apers, A. R. Hevner and S. B. Yao, Optimization, algorithms for distributed queries. *IEEE Trans. on Software Engineering* **9**, 57–68 (1983).
3. A. P. Bernstein, N. Goodman, E. Wong, C. L. Reeve and J. Rothnie, Query processing in a system for distributed databases (SDD-1). *ACM Trans. on Database Systems* **6**, 602–625 (1981).
4. A. P. Bernstein and D. W. Chiu, Using semi-joins to solve relational queries. *Journal of the ACM* **28**, 25–40 (1981).
5. S. Ceri and G. Pelagatti, *Distributed Databases – Principles and Systems*. McGraw-Hill, New York (1984).
6. J-M. Chang. A heuristic approach to distributed query processing. *Proc. of the 8th Intern. Conf. on Very Large Data Bases*, 54–61 (1982).
7. A. L. P. Chen and V. O. K. Li, Optimizing star queries in a distributed database system. *Proc. of the 10th Intern. Conf. on Very Large Data Bases*, 429–438 (1984).
8. E. F. Codd, A relational model for large shared data banks. *Communications of the ACM* **13**, 909–917 (1970).
9. C. J. Date, *An Introduction to Database Systems*. Addison-Wesley, New York (1981).
10. R. S. Epstein, M. Stonebraker and E. Wong, Distributed query processing in a relational database system. *Proceedings ACM-SIGMOD*, 169–180 (1979).
11. D. Erlenkotter, A dual based procedure for uncapacitated facility location. *Operations Research* **26**, 992–1009 (1978).
12. R. M. Garey and D. S. Johnson, *Computers and Intractability: a Guide to the Theory of NP-completeness*. W. H. Freeman, New York (1979).
13. B. Gavish and S. K. Srikanth, *Optimal Solution Methods for Large Scale Multiple Traveling Salesman Problem*. Working paper, the Graduate School of Management, the University of Rochester (1980).
14. B. Gavish, Topological design of centralized computer networks – formulations and algorithms. *Networks* **12**, 355–377 (1982).
15. B. Gavish and A. Segev, Set query optimization in distributed database systems. *ACM Trans. on Database Systems* **11**, 265–293 (1986).
16. A. M. Geoffrion, Lagrangian relaxation and its uses in integer programming. *Mathematical Programming Study* **2**, 82–114 (1974).
17. A. M. Geoffrion and R. McBride, Lagrangian relaxation applied to capacitated facility location problems. *AIIE Transactions* **10**, 40–47 (1978).
18. M. Held and R. M. Karp, The traveling salesman problem and minimum spanning trees. *Operations Research* **18**, 1138–1162 (1970).
19. A. R. Hevner and S. B. Yao, Query processing in distributed database systems. *IEEE Transactions on Software Engineering* **5**, 177–187 (1979).
20. E. Horowitz and S. Sahni, *Fundamentals of Computer Algorithms*. Computer Science Press, Rockville, MD (1978).
21. L. Kerschberg, P. D. Ting and S. B. Yao, Query optimization in star computer networks. *ACM Trans. on Database Systems* **7**, 678–711 (1982).
22. I-S. Paik and C. Delobel, A Strategy for optimizing the distributed query processing. *Proceedings of the First International Conference on Distributed Computing Systems*. Huntsville, Alabama (1979).
23. R. W. Peebles and E. D. Manning. A computer architecture for large (distributed) databases. *Proceedings of the*

Conference on Very Large Databases, Framingham, Mass., (1975).

24. G. Pelagatti and F. A. Schreiber, A model of an access strategy in a distributed database system. *IFIP-TC2, Database Architecture*, Venice (1979).
25. J. B. Rothnie and N. Goodman, A Survey of R&D in distributed database management. *Proceedings of the International Conference on Very Large Databases*, Tokyo (1977).
26. C. Sandi, Subgradient optimization. In *Combinatorial Optimization*, edited N. Christofides, A. Mingizzi, P. Toth and C. Sandi. John Wiley, New York (1979).
27. A. Segev, Optimization of join operations in horizontally partitioned database systems. *ACM Trans. on Database Systems* **11**, 48–80 (1986).
28. P. G. Selinger and M. E. Adiba, Access path selection in distributed data base management systems. *Proceedings of the International Conference on Databases*, 204–215 (1980)
29. M. Stonebraker and E. Neuhold, A distributed database version of INGRES. *Proceedings of the Third Berkeley Workshop on Distributed Data Management and Computer Networks* (1977).
30. M. Stonebraker, J. Woodfill, J. Ranstrom, M. Murphy, J. Kalash, M. Carey and K. Arnold, Performance analysis of distributed data base systems. *Database Engineering*. IEEE Computer Society, 58–65 (Dec. 1982).
31. R. Williams *et al., R*: an Overview of the Architecture*. IBM Research Report RJ3325 (1981).
32. E. Wong, Retrieving dispersed data from SDD-1: a system for distributed databases. *Proceedings of the Third Berkeley Workshop on Distributed Data Management and Computer Networks*, 217–235 (1977).
33. C. T. Yu and C. C. Chang, On the design of a query processing strategy in a distributed database environment. *Proc. of ACM Sigmod*, 30-39 (1983).
34. C. T. Yu and C. C. Chang. Distributed query processing. *ACM Computing Surveys* **16**, 399–433 (1984).

APPENDIX: DERIVING THE LAGRANGIAN RELAXATIONS

This appendix describes the derivation of two Lagrangian relaxations to the 2-way-join problem. The first relaxation is applied to problem P_{T1}, and the second to problem P_{T2}.

LAGRANGIAN RELAXATION OF PROBLEM P_{T1}

It was shown in Section 5.1 that problem P_{T1} is separable into two symmetric subproblems P_{T11} and P_{T12}. The following Lagrangian relaxation applies to problem P_{T11}. To get the relaxation for problem P_{T12}, one has simply to change the subscripts as discussed in Section 5.1.

Multiplying constraints (15) by a conformable vector of Lagrangian multipliers μ_{it}, and adding them to the objective function (13) results in the following Lagrangian problem.

Problem P_{L1}

$$\min\{\sum_{i\in S1} F_i C_{iq} + \sum_{t\in S1} a_t X_t + \sum_{i\in S2}\sum_{k\in S1+i}\sum_{t\in S1}(b_{ikt}W_{ikt} + d_{ikt}g_{ikt})\} \qquad \text{(A 1)}$$

subject to: (14), (16)–(18)

Where:

$$a_t = -\sum_{k\in S2}\mu_{kt} - F_t C_{tq}(1-\alpha_t)$$

$$b_{ikt} = D_i C_{kt}$$

$$d_{ikt} = \begin{cases}\mu_{it} - \mu_{ik}, & i\in S2, k\in S1, t\in S1\\ \mu_{it}, & i\in S2, k = i, t\in S1\end{cases}$$

Let W^*_{ikt} be the optimal values of variables W_{ikt} in problem P_{L1}; then the optimal values of the variables g_{ikt} are given by:

$$g^*_{ikt} = \begin{cases}0, & \text{if } W^*_{ikt} = 0\\ 0, & \text{if } W^*_{ikt} = 1 \text{ and } d_{ikt} \geqslant 0\\ |S1|, & \text{if } W^*_{ikt} = 1 \text{ and } d_{ikt} < 0\end{cases} \qquad \text{(A 2)}$$

To get a compact formulation we define the following variables:

$$\bar{h}_{ikt} = \begin{cases}0, & \text{if } d_{ikt} \geqslant 0\\ |S1|\,d_{ikt}, & \text{otherwise}\end{cases}$$

$$h_{ikt} = b_{ikt} + \bar{h}_{ikt}$$

Using the relationship $d_{ikt}g_{ikt} = \bar{h}_{ikt}W_{ikt}$, problem P_{L1} reduces to the following problem:

Problem $\hat{P}_{L1}$

$$\min\{\sum_{i\in S1} F_i C_{iq} + \sum_{i\in S1} a_i X_i + \sum_{i\in S2}\sum_{k\in S1+i}\sum_{t\in S1} h_{ikt}W_{ikt}\} \qquad \text{(A 3)}$$

subject to: (14) and (18).

The optimal solution to problem $\hat{P}_{L1}$ is given by Algorithm A 1 below.

Algorithm A 1

```
1. For every $t\in S1$ Do Steps 2 and 3.
2. For every $j\in S2$ Calculate $h_{j\tilde{k}t} = \min_{k\in S1+j}\{h_{jkt}\}$.
3. If $(a_t + \sum_{j\in S2} h_{j\tilde{k}t}) \geqslant 0$, then:
       $X_t = 0$ and $W_{jkt} = 0,\ \forall j\in S2$ and $k\in S1+j$
   Else:
       $X_t = 1$
       $W_{j\tilde{k}t} = 1,\ \forall j\in S2$
       $W_{jkt} = 0,\ \forall j\in S2$ and $k \neq \tilde{k}$
```

The optimal value of (A 3), as calculated by Algorithm A 1, is a lower bound on the value of an optimal solution to problem P_{T1}. The optimal values of the variables g_{ikt} are calculated from (A 2).

LAGRANGIAN RELAXATION OF PROBLEM P_{T2}

The basic idea underlying the following relaxation is to decompose problem P_{T21} (and similarly problem P_{T22})

into |S2| × |S1| network flow sub-problems plus other problems with low polynomial complexity. To get the standard network flow problems, the coupling variables X_i should be taken care of. One alternative is to relax constraints (21) and (22), and a second one is to 'transfer' the X_i variables from equations (21) and (22) to the flow-conservation equations of special dummy nodes. The second alternative is likely to produce tighter lower bounds and is adopted here. The augmented network is constructed as follows. For every pair of fragments, $j \in T$ and $i \in \Gamma_j$ (corresponding to *path(j, i)*), define a dummy node d_{ji}, and and two arcs (j, d_{ji}) and (d_{ji}, i) with zero arc costs (see Fig. A 1). Also, replace constraints (21) and (22) by the following set of constraints:

$$\sum_{t\in T} f_{jikt} - \sum_{t\in S1+j} f_{jitk} = 1, \quad k = j, j\in S2, i\in S1 \qquad \text{(A 4)}$$

$$\sum_{t\in T} f_{jikt} - \sum_{t\in S1+j} f_{jitk} = -1, \quad k = i, j\in S2, i\in S1 \qquad \text{(A 5)}$$

$$f_{jid_{ji}i} = 1 - X_i, \quad i\in S1, j\in S2 \qquad \text{(A 6)}$$

$$f_{jijd_{ji}} = 1 - X_i, \quad i\in S1, j\in S2 \qquad \text{(A 7)}$$

Clearly, if $X_i = 1$, no flow will pass through node d_{ji}, $\forall\, j\in S2$. If $X_i = 0$, *path(j, i)* must consist of arcs (j, d_{ji}) and (d_{ji}, i), and the cost of flow along this path is zero. Hence, the resulting formulation is equivalent to the previous one.

To tighten the formulation, the following set of redundant constraints is added:

$$\sum_{t\in S1+j} W_{jti} \geqslant X_i, \quad i\in S1, j\in S2 \qquad \text{(A 8)}$$

The constraints in (A 8) state that if fragment i is to be restricted by a semi-join, the joining attribute of every fragment of the other relation must be available at site i.

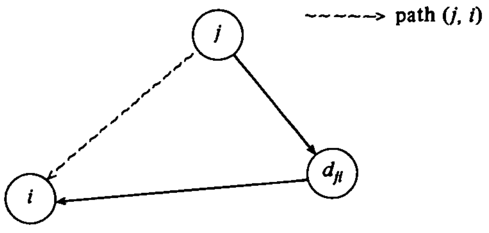


Figure A 1. Constructing the augmented network.

The Lagrangian relaxation of problem P_{T21} is achieved by multiplying constraints (A 6) and (A 7) by vectors of Lagrangian multipliers λ_{ji} and μ_{ji} respectively, and the sum (over $i\in S1$) of constraints (24) by a vector of non-negative multipliers δ_{jtk}, and adding those constraints to the objective function (19). The relaxed problem is stated as problem P_{L2} below.

Problem P_{L2}

$$\min\{A(\lambda,\mu) + \sum_{i\in S1} a_i X_i + \sum_{j-S2}\sum_{k\in S1+j}\sum_{t\in S1} b_{jkt} W_{jkt}$$

$$+ \sum_{j\in S2}\sum_{i\in S1}\sum_{t\in S1+j+d_{ij}}\sum_{k\in S1+d_{ij}} d_{jitk} f_{jitk}\} \qquad \text{(A 9)}$$

subject to: (20, (23), (25), (26), (A 4), (A 5) and (A 8)

where:

$$A(\lambda,\mu) = \sum_{i\in S1} F_i C_{iq} - \sum_{j\in S2}\sum_{i\in S1} (\lambda_{ji} + \mu_{ji})$$

$$a_i = \sum_{j\in S2} (\lambda_{ji} + \mu_{ji}) - F_i C_{iq}$$

$$b_{jkt} = D_j C_{kt} - \delta_{jkt}$$

$$d_{jitk} = \begin{cases} \dfrac{\delta_{jtk}}{|S1|}, & j\in S2, i\in S1, t\in S1+j, k\in S1 \\ \lambda_{ji}, & j\in S2, i\in S1, t = d_{ji}, k = i \\ \mu_{ji}, & j\in S2, i\in S1, t = j, k = d_{ji} \\ \infty, & \textit{otherwise} \end{cases}$$

Problem P_{L2} is separable into two sub-problems, P_{L21} and P_{L22}, as follows:

Problem P_{L21}

$$\min\{A(\lambda,\mu) + \sum_{t\in S1} a_t X_t$$

$$+ \sum_{j\in S2}\sum_{k\in S1+j}\sum_{t\in S1} b_{jkt} W_{jkt}\} \qquad \text{(A 10)}$$

subject to: (A 8), and

$$X_t, W_{jkt} \in \{0, 1\}, \quad j\in S2, k\in S1+j, t\in S1 \qquad \text{(A 11)}$$

Problem P_{L22}

$$\text{Min}\{\sum_{j\in S2}\sum_{i\in S1}\sum_{t\in S1+j+d_{ij}}\sum_{k\in S1+d_{ij}} d_{jitk} f_{jitk}\} \qquad \text{(A 12)}$$

subject to: (20), (A 4), (A 5) and

$$f_{jid_{ji}i} - f_{jijd_{ji}} = 0, \quad i\in S1, j\in S2 \qquad \text{(A 13)}$$

$$0 \leqslant f_{jikt} \leqslant 1, \quad i\in S1, j\in S2, k\in S1+j+d_{ij}, t\in S1+d_{ij} \qquad \text{(A 14)}$$

Problem P_{L21} is separable over $t\in S1$, and is solved optimally by Algorithm A 2 below.

Algorithm A 2

```
1. For every $t\in S1$ Do Steps 2 to 4.
2. (a) Let $J_{jt}^- = \{k \mid b_{jkt} \leqslant 0, k\in S1+j\}$
   (b) Let $b_{j\bar{k}t} = \min_{k\in S1+j} \{b_{jkt}\}$
   (c) Let $\bar{b}_{jt} = \sum_{k\in J_{jt}^-} b_{jkt}$, where the summation is understood to be $\infty$ if $J_{jt}^- = \varnothing$.
   (d) Let $\bar{b}_{jt} = Min\{\bar{b}_{jt}, b_{j\bar{k}t}\}$
3. If $a_t \geqslant 0$ set $X_t = 0$; Else,
       $X_t = \begin{cases} 1, & \text{if } \sum_{j\in S2} \bar{b}_{jt} + a_t < 0 \\ 0, & \textit{otherwise} \end{cases}$
4. For every $j\in S2$:
       $W_{jkt} = \begin{cases} 1, & \text{if } k\in J_{jt}^- \\ 1, & \text{if } J_{jt}^- = \varnothing, X_t = 1, \text{and } k = \bar{k} \\ 0, & \textit{otherwise} \end{cases}$
```

Problem P_{L22} is a Multi-Commodity Network Flow problem with no interaction between the commodities.

Hence the problem is separable into |S 1| × |S 2| single-commodity network flow problems. Let P_{ji} denote each one of those problems and Z*(P) denote the value of an optimal solution to a generic problem P. Then, the value of an optimal solution to problem P_{L2} is given by:

$$Z^*(P_{L2}) = Z^*(P_{L21}) + \sum_{i \in S1} \sum_{j \in S2} Z^*(P_{ji})$$

Solving the problems P_{ji} can be made more efficient by the use of Lemma A 1.

Lemma A 1

Let SPATH(j, i) denote the problem of finding the shortest path from node i to node j in the non-augmented network. Then,

$$Z^*(P_{ji}) = \min\{Z^*(SPATH(j, i)), (\lambda_{ji} + \mu_{ji})\}$$

Proof

It follows from constraints (A 4) and (A 5) and the non-negativity of δ_{jtk}'s that an optimal solution to problem P_{ji} will contain no arc entering node j or leaving node i. That implies that exactly one arc must leave node i and enter node j. If the path {j, d_{ji}, i} is part of the optimal solution to problem P_{ji}, it must be the only path (all other paths are excluded due to the non-negativity of δ_{jtk}'s). Since arc flows are integral and limited to one unit, if node d_{ji} is not on an optimal path (j, i), the optimal path constitutes a shortest path from node j to node i. That path does not contain cycles, since arc costs are non-negative in the non-augmented network.

Q.E.D.

Lemma A 1 implies that we have to solve |S 1| × |S 2| Shortest Path problems instead of |S 1| × |S 2| Network Flow problems. Moreover, since those Shortest Path problems are independent, the solution of problem P_{ji} can be achieved by solving |S 2| shortest problems, each from node j ∈ S 2 to all nodes i ∈ S 1.