Complexity in the Database Allocation Design

- Must take relationship between fragments into account
- Cost of integrity enforcements
- Constraints on response-time, storage, and processing capability

Needed Information to do Allocation

- Database information: tuple size, cardinality of fragment
- Application information: #updates/#retrieval a query performs on a fragment
- Site Information: storage and processing capabilities and cost of processing a unit of work
- Network Information: communication cost to transfer a block of data between sites $i$ and $j$
• No models developed to date can handle all the constraints

• Current models simplify assumptions and work with some specific situations

Formulation of DAP

• DAP can be formulated as an optimization problem

\[
\text{Min(Total Cost)} \\
\text{Subject to} \\
\quad \cdot \text{response time constraint} \\
\quad \cdot \text{Storage constraint} \\
\quad \cdot \text{Processing load constraint}
\]

DAP is NP-complete and several heuristics have been proposed
Constraints

- **Execution time constraint**
  - Execution time of \( q_i \) <= maximum response time of \( q_i \) for all \( q_i \) in \( Q \)

- **Storage constraint**
  \[ \sum_{F} STC_{jk} \leq \text{storage capacity at site } S_k, \forall S_k \in S \]

- **Processing constraint**
  \[ \sum_{q_i \in Q} \text{processing load of } q_i \text{ at site } S_k \leq \text{processing capacity of } S_k, \forall S_k \in S \]

Cost Computation

- **Decision variable** \( x_{ij} \) defined as
  \[ x_{ij} = \begin{cases} 
  1 & \text{if fragment } F_i \text{ is stored at site } S_j \\
  0 & \text{otherwise} 
\end{cases} \]

- **Total Cost** = query processing cost + storage cost
- **Solve the optimization constraint for** \( x_{ij} \)
Cost Model

- Total cost = query processing cost + storage cost

\[ TOC = \sum_{q_i \in Q} QPC_i + \sum_{S_k \in S} \sum_{F_j \in F} STC_{jk} \]

Unit cost of storing data at Sk

\[ STC_{jk} = USC_k \cdot size(F_j) \cdot x_{jk} \]

\[ size(F_j) = card(F_j) \cdot length(F_j) \]

Query Processing Cost

\[ QPC_i = PC_i + TC_i \]

- Computation cost of \( q_i \)
- Transfer cost of \( q_i \)

\[ PC_i = AC_i + IE_i + CC_i \]

- Access cost of \( q_i \)
- Integrity enforcement cost of \( q_i \)
- Concurrency control cost of \( q_i \)
LPC<sub>k</sub>: Cost of processing one unit of work at site S<sub>k</sub>
RR<sub>ij</sub>: Number of read accesses a query q<sub>i</sub> makes to a fragment F<sub>j</sub>
UR<sub>ij</sub>: # of update accesses a query q<sub>i</sub> makes to a fragment F<sub>j</sub>

Access cost of query q<sub>i</sub>

\[ AC_i = \sum_{S_k \in S} \sum_{F_j \in F} (u_{ij} \cdot UR_{ij} + r_{ij} \cdot RR_{ij}) \cdot x_{jk} \cdot LPC_k \]

\[ u_{ij} = \begin{cases} 
1 & \text{if } q_i \text{ updates } F_j \\
0 & \text{0 otherwise} 
\end{cases} \]

Assume cost of an update same as cost of retrieval

\[ r_{ij} = \begin{cases} 
1 & \text{if } q_i \text{ retrieves from } F_j \\
0 & \text{0 otherwise} 
\end{cases} \]

Transmission Cost Model

\[ TC_i = TCU_i + TCR_i \]

- Update cost: Need to perform updates to all replicas; no large results sent back

\[ TCU_i = \sum_{S_k \in S} \sum_{F_j \in F} u_{ij} \cdot x_{jk} \cdot g_{a(i),k} + \sum_{S_l \in S} \sum_{F_j \in F} u_{ij} \cdot x_{jk} \cdot g_{k,ot(i)} \]

\[ g_{ij}: \text{communication cost per message between } S_i \text{ and } S_j \]
Retrieval Cost Model

\[ TCR_i = \sum_{F_j \in F} \min_{S_k \in S} \left( r_j \times x_{jk} \times g_{sel_i}(F_j) \right) + r_j \times x_{jk} \times \frac{sel_i(F_j) \times \text{length}(F_j)}{\text{fsize}} \times g_{k,sel_i} \]

Cost of sending a query\[\rightarrow\]
Cost of sending the results back\[\rightarrow\]

Pick the least cost site among all sites with the replicas

\[ g_{ij}: \text{communication cost per message between } S_i \text{ and } S_j \]
\[ f_{\text{size}}: \text{#Bytes in a message} \]
\[ \text{length}(F_j): \text{#bytes in fragment } F_j \]
\[ \text{Sel}_i(F_j): \text{Selectivity Factor of } q_i \text{ on } F_j \]

Heuristic Approaches

- Allocation of Horizontal Fragments
- Allocation of Vertical Fragments

(Material not in the textbook)
Allocation of Horizontal Fragments (1)

No replication: Best Fit Strategy
- The number of local references of $R_i$ at site $j$ is

$$B_{ij} = \sum_k f_{kj} n_{ki}$$

- $R_i$ is allocated at site $j^*$ such that $B_{j^*}$ is maximum.

**Advantage:** A fragment is allocated to a site that needs it most.

**Disadvantage:** It disregards the “mutual” effect of placing a fragment at a given site if a related fragment is also at that site.
Allocation of Horizontal Fragments (2)

All beneficial sites approach (replication)

\[
B_{ij} = \sum_k f_{ij}r_{ki} - c \sum_k \sum_{j \neq j} f_{kj}u_{ki}
\]

Cost of retrieval references  Cost of update references from other sites

\(R_i\) is allocated at all sites \(j^*\) such that \(B_{ij^*} > 0\). When all \(B_{ij}\)'s are negative, a single copy of \(R_i\) is placed at the site such that \(B_{ij^*}\) is maximum.

Allocation of Horizontal Fragments (3)

Another Replication Approach:

| \(d_i\) | The degree of redundancy of \(R_i\) |
| \(F_i\) | The reliability and availability benefit of having \(R_i\) fully replicated. |
| \(\beta(d_i)\) | The reliability and availability benefit when the fragment has \(d_i\) copies. |

\[
\beta(d_i) = (1 - 2^{-d_i}) \cdot F_i \quad \beta(1) = 0, \quad \beta(2) = \frac{F_i}{2}, \quad \beta(3) = \frac{3}{4} \cdot F_i, \ldots
\]

The benefit of introducing a new copy of \(R_i\) at site \(j\):

\[
B_{ij} = \sum_k f_{ij}r_{ki} - c \sum_k \sum_{j \neq j} f_{kj}u_{ki} + \beta(d_i)
\]

Also takes into account the benefit of replication
Allocation of Horizontal Fragments (4)

- All Beneficial Sites Approach:
  1. Determine the set of all sites where the benefit of allocating one copy of the fragment is higher than the cost.
  2. Allocate a copy of the fragment to each site in the set.

- Alternatively:
  1. Determine the solution of the non-replicated problem.
  2. Progressively introduce replicated copies starting from the most beneficial; the process is terminated when no additional replication is beneficial.

How about Heuristics for Vertical Allocation?
SUMMARY

Design of a distributed DB consists of four phases:

- Phase 1: Global schema design (same as in centralized DB design)

- Phase 2: Fragmentation
  - Horizontal Fragmentation
    - Primary: Determine a complete and minimal set of predicates
    - Derived: Use semijoin
  - Vertical Fragmentation
    Identify fragments such that many applications can be executed using just one fragment.

- Phase 3: Allocation
  The primary goal is to minimize the number of remote accesses.

- Phase 4: Physical schema design (same as in centralized DB design).