BELGIUM CAMPUS iTversity

It's the way we're wired

DATABASE DEVELOPMENT

ARE YOU READY?

DBD371/381

DISTRIBUTED QUERY PROCESSING

Overview of Query Processing

LEARNING OBJECTIVES

BELGIUM CAMPUS iTversity
It's the way we're wired

- Overview of query processing in distributed DBMSs
- Query Processing Problem
- Steps Involved in a User Query
- Query Optimization Strategies
- Layers of Query Processing

QUERY PROCESSING

- Query Processing are Activities involved in retrieving data from the database
- Aims of QP:
  - – transform query written in high-level language (e.g. SQL), into correct and efficient execution strategy expressed in low-level language (implementing RA);
  - – execute the strategy to retrieve required data.
- Four main Phases: decomposition, optimization, code generation and execution.

QUERY PROCESSING

- Find all Managers that work at a London branch.

```
SELECT Sno, Fname, Position, Bno
FROM staff s, branch b
WHERE s.bno = b.bno AND
(s.position = ‘Manager’ AND b.city = ‘London’);
```

Three equivalent RA queries are:

Cartesian product and join operations are much more expensive than selection, and third option significantly reduces size of relations being joined together.

(1) $\sigma_{(position='Manager') \wedge (city='London') \wedge (staff.bno=branch.bno)}(Staff \times Branch)$

(2) $\sigma_{(position='Manager') \wedge (city='London')}(Staff \bowtie_{staff.bno=branch.bno} Branch)$

(3) $(\sigma_{position='Manager'}(Staff)) \bowtie_{staff.bno=branch.bno} (\sigma_{city='London'}(Branch))$

•Overview(NP-complete problem)

- **CHALLENGE ON PROCESSING USER QUERIES.**

1. The distribution of operational data on disperse data sources
2. The database relations required stored at multiple sites leads to an exponential increase in the number of possible equivalent or alternatives for a user query
3. It is not computationally reasonable to explore exhaustively all possible query plans in a large search space
4. A strategy is required to produce an optimal query plan with the most cost-effective option, for query processing

OVERVIEW

- Database systems of different types use various techniques to identify optimal query plans
- System considers only the most important (top-k) query answers in a huge answer space
- A distributed database encompasses coherent data, spread across various operational autonomous sites of a computer network.
- A Distributed Database Management System (DDBMS) deals with managing such distributed databases.
- The query processing is controlled by DBMS
- The performance of a DBMS is determined by its ability to process queries in an effective and efficient manner

Overview

- Query processing connects to many database areas:
  - ➢query optimization
  - ➢ indexing methods
  - ➢query language

➢the impact of efficient processing of query is apparent in number of applications

OVERVIEW OF QUERY PROCESSING IN DISTRIBUTED DBMSS

Distributed database design is of major importance for query processing

- ➢ objective of increasing reference locality, and parallel execution for the most important queries.
- Query optimization, a time-consuming task best handled by the query processor,
- Query processing in distributed databases is more complex than in centralized, →various parameters or constraints are involved that affect performance of distributed queries.
- The data distribution policy. (full replication or partition/fragment) decides the manner of distribution of logical units of operational data
- Relations involved in a distributed query may be fragmented and/or replicated, induce communication overhead costs.
- Query response time may become very high.

Query Processing Problem

- The main function of a relational query processor is to transform a high-level into an equivalent lower-level query
- The low-level query implements the execution strategy for the query.
- The transformation must achieve both correctness and efficiency.
- It is correct, if both queries produce the same result.
- The main difficulty is to select the execution strategy that minimizes resource consumption.

QUERY PROCESSING PROBLEM

- consider the following subset of the engineering database:

  EMP(ENO, ENAME, TITLE)

  ASG(ENO, PNO, RESP, DUR)

  <u>User request:</u>

Select Query	Transformations of the query to RA

SELECT ENAME
FROM EMP,ASG
WHERE EMP.ENO = ASG.ENO
AND RESP = ''Manager''
	$\Pi_{ENAME}(\sigma_{RESP=\text{“Manager”} \wedge EMP.ENO=ASG.ENO}(EMP \times ASG))$
	$\Pi_{ENAME}(EMP \bowtie_{ENO} (\sigma_{RESP=\text{“Manager”}}(ASG)))$
	Second query, avoids the cartesian product
	Of EMP and ASG, thus consumes less computing resources

QUERY OPTIMIZATION

- Activity of choosing an efficient execution strategy for processing query.
- *As* there are many equivalent transformations of same high-level query, aim of QO is to choose one that minimizes resource usage.
- Generally, reduce total execution time of query.
- May also reduce response time of query.
- Problem computationally intractable with large number of relations, so strategy adopted is reduced to finding near optimum solution.

IMPORTANCE OF SITE SELECTION AND COMMUNICATION FOR A CHOSEN RELATIONAL ALGEBRA QUERY

Assume Horizontal fragmentation for relations EMP and ASG

$$EMP_1 = \sigma_{ENO \leq \text{"E3"}}(EMP)$$
$$EMP_2 = \sigma_{ENO > \text{"E3"}}(EMP)$$
$$ASG_1 = \sigma_{ENO \leq \text{"E3"}}(ASG)$$
$$ASG_2 = \sigma_{ENO > \text{"E3"}}(ASG)$$

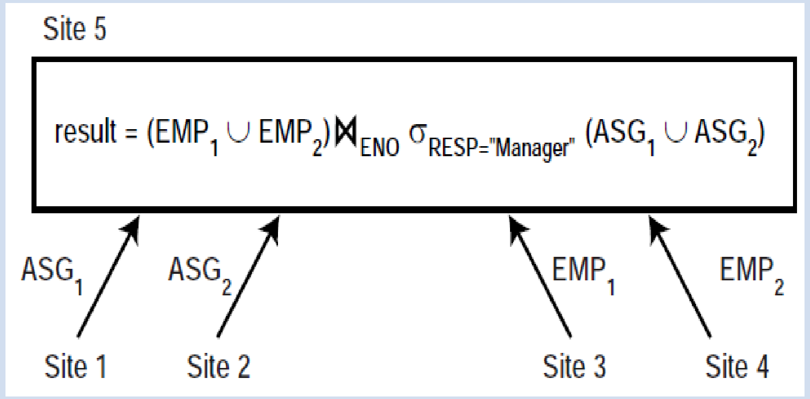
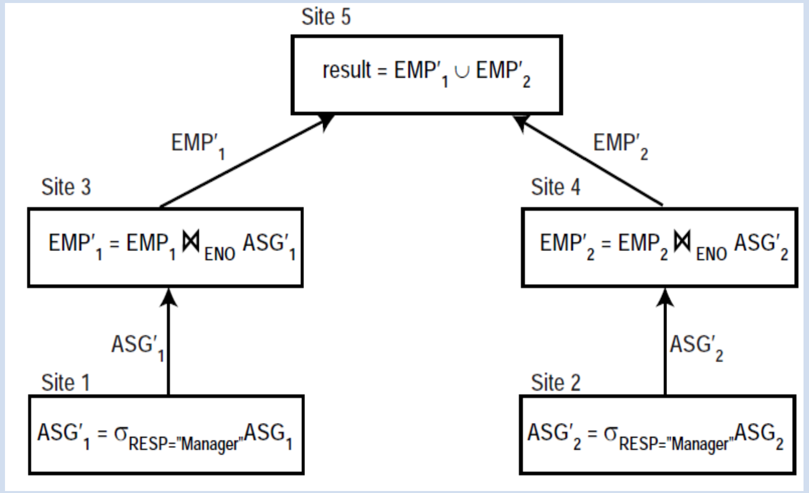
$$\Pi_{ENAME}(EMP \bowtie_{ENO} (\sigma_{RESP=\text{"Manager"}}(ASG)))$$

Assume ASG1, ASG2, EMP1, and EMP2 are stored at sites 1, 2, 3, and 4, respectively

Say result is expected at site 5

Execution Strategies

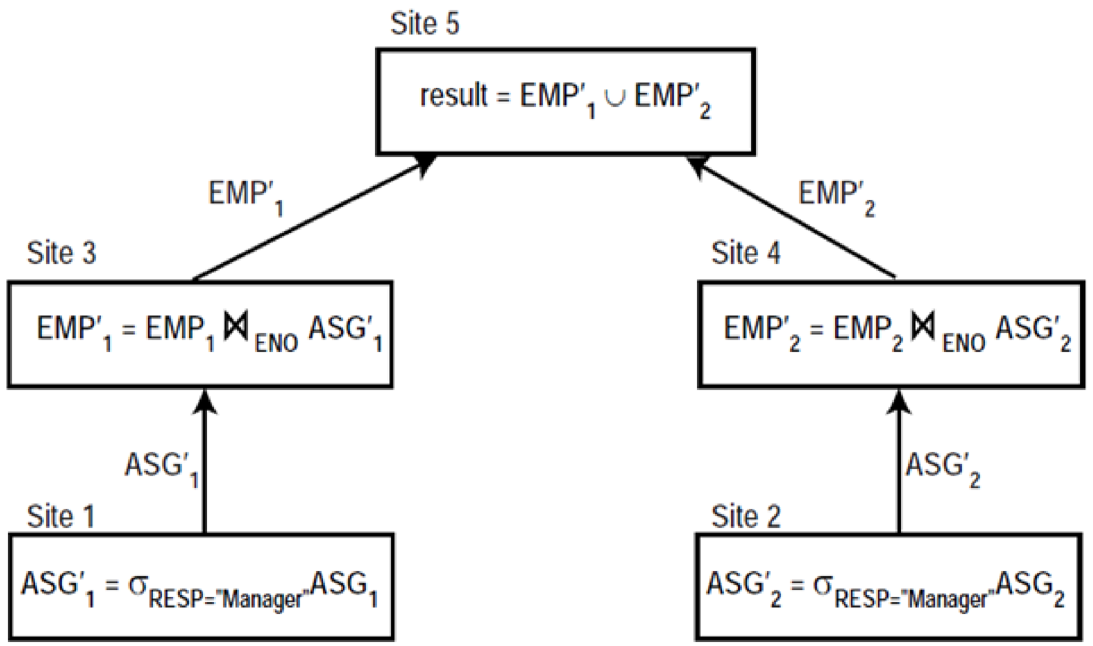
Equivalent Distributed Execution Strategies

Strategy A	StrategyB
	
1. Transfer EMP to site 5 2. Transfer ASG to site 5 3. Produce ASG₀ by selecting ASG 4. Join EMP and ASG₀	Strategy B is better Provides better distribution of work among sites

Site 5
result = $(EMP_1 \cup EMP_2) \bowtie_{ENO} \sigma_{RESP="Manager"} (ASG_1 \cup ASG_2)$
ASG_1
ASG_2
EMP_1
EMP_2
Site 1
Site 2
Site 3
Site 4

Site 5
result = $EMP'_1 \cup EMP'_2$
EMP'_1
EMP'_2
Site 3
$EMP'_1 = EMP_1 \bowtie_{ENO} ASG'_1$
Site 4
$EMP'_2 = EMP_2 \bowtie_{ENO} ASG'_2$
ASG'_1
ASG'_2
Site 1
$ASG'_1 = \sigma_{RESP="Manager"} ASG_1$
Site 2
$ASG'_2 = \sigma_{RESP="Manager"} ASG_2$

What is the total cost for Strategy B?



- Assumptions:
  1. tuple access, denoted by **tupacc**, is 1 unit
  2. tuple transfer, denoted **tuptrans,** is 10 units.
  3. relations EMP and ASG have 400 and 1000 tuples, respectively,
  4. There are 20 managers in relation ASG.
  5. data is uniformly distributed among sites.
  6. relations ASG and EMP are locally clustered on attributes RESP and ENO

The total cost of strategy B can be derived as follows:

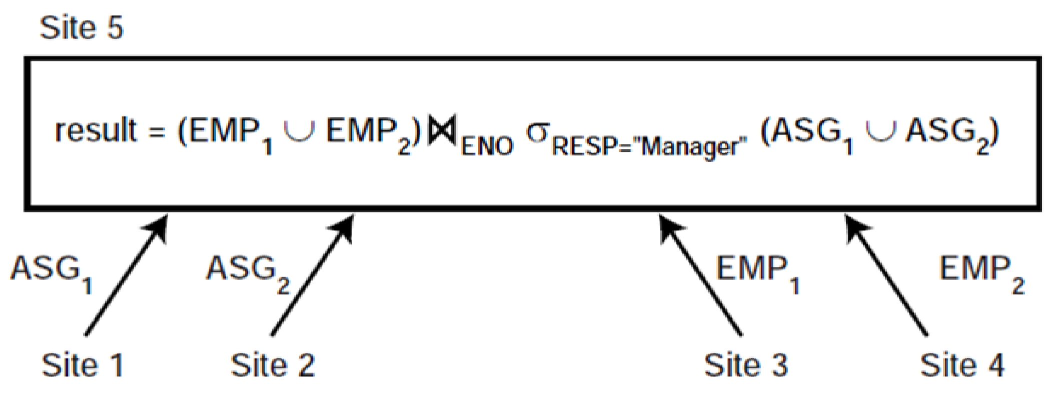
1. Produce ASG′ by selecting ASG requires $(10+10) * tupacc$ = 20
2. Transfer ASG′ to the sites of EMP requires $(10+10) * tuptrans$ = 200
3. Produce EMP′ by joining ASG′ and EMP requires $(10+10) * tupacc * 2$ = 40
4. Transfer EMP′ to result site requires $(10+10) * tuptrans$ = 200

The total cost is 460

QUERY PROCESSING PROBLEM

- To evaluate the resource consumption of these two strategies, we use a simple cost model.
- Assumptions:
  1. tuple access, denoted by **tupacc,** is 1 unit
  2. tuple transfer, denoted **tuptrans,** is 10 units.
  3. relations EMP and ASG have 400 and 1000 tuples, respectively,
  4. There are 20 managers in relation ASG.
  5. data is uniformly distributed among sites.
  6. relations ASG and EMP are locally clustered on attributes RESP and ENO
- . Therefore, there is direct access to tuples of ASG (respectively, EMP) based on the value of attribute RESP (respectively, ENO).

STRATEGY A


Site 5
result = $(EMP_1 \cup EMP_2) \bowtie_{ENO} \sigma_{RESP="Manager"} (ASG_1 \cup ASG_2)$
ASG_1
ASG_2
EMP_1
EMP_2
Site 1
Site 2
Site 3
Site 4

- **Assumptions**:
  1. tuple access, denoted by **tupacc**, is 1 unit
  2. tuple transfer, denoted **tuptrans,** is 10 units.
  3. relations EMP and ASG have 400 and 1000 tuples, respectively,
  4. There are 20 managers in relation ASG.
  5. data is uniformly distributed among sites.
  6. relations ASG and EMP are locally clustered on attributes RESP and ENO

The cost of strategy A can be derived as follows:

1. Transfer EMP to site 5 requires $400 * tuptrans$ $= 4,000$
2. Transfer ASG to site 5 requires $1000 * tuptrans$ $= 10,000$
3. Produce ASG′ by selecting ASG requires $1000 * tupacc$ $= 1,000$
4. Join EMP and ASG′ requires $400 * 20 * tupacc$ $= 8,000$

The total cost is $\overline{23,000}$

QUERY PROCESSING

- The distribution of data imposes challenges to query processing, since for a user query there are multiple semantically valid query equivalent plans (QEPs) possible.
- The alternatives are equivalent in terms of outcome as they retrieve same set of database objects or records.
- Selecting best alternatives for processing from generated pool is a decisive task to query optimizers.
- Query optimizer's primary objective is to choose optimal (best) solution
- An alternative is considered better or fitter than others based on the objective function values for the query result generation.

QUERY PROCESSING IN DISTRIBUTED DBMSS

- The role of a distributed query processor is to map a high-level query on a distributed database (global relations) into a sequence of database operators (relational algebra) on relation fragments
- Several important functions characterize this mapping.

1. Decomposition of the query into a sequence of relational operators called an algebraic query
2. Data localization so that the operators on relations are translated to bear on local data (fragments).
3. Algebraic query on fragments must be extended with communication operators and optimized with respect to a cost function to be minimized.

Query processing and optimization is constrained and affected by different cost parameters such as,

- ➢ communication cost,
- ➢ local processing cost,
- ➢ optimization cost,
- ➢ query localization cost

Steps Involved In a user Query

1. Identification of the relevant logical units of the query
2. Identification of the data relevant to the query (available at different sites)
3. Transmission of data between sites along with local data processing (distributed query processing (DQP))
4. The distributed query is parsed before arriving at an effective query processing strategy for it
5. Decomposition of the distributed query into local sub-queries to be executed at their respective sites
6. Logical order of relational operators and the site (control site) at which the results of the sub-queries are integrated
7. Integrated result is provided as the answer of the query.

QUERY PROCESSING

- DQP strategy aims to generate query processing plans that reduce the amount of data transfer between participating sites by selecting the appropriate copy of data for query result retrieval and thereby reduces the overall distributed query response time
- inter-site communication/transfer of relevant data is dominant factor or cost constraint, to the overall query processing and optimization.
- The degree of data transfer is directly proportional to the heterogeneity in sites accessed in a query plan
- In DQP, the costs incurred are CPU, I/O and the site-to-site communication (dominant cost).

QUESTION

- Consider a relation that is fragmented horizontally by plant number: employee (name, address, salary, plant number)
- Assume that each fragment has two replicas: one stored at the "New York" site and one stored locally at the plant site. (Local plant sites:Toronto,Edmonton, Vancouver, Montreal, San Jose, Boca)
- Describe a good processing strategy for the following queries entered at the San Jose site.
  - a. Find all employees at the Boca plant.
  - b. Find the average salary of all employees.
  - c. Find the highest-paid employee at each of the following sites: Toronto, Edmonton, Vancouver, Montreal.
  - d. Find the lowest-paid employee in the company.

SOLUTIONS

- ➢a. Find all employees at the Boca plant.

- a. i. Send the query Π_{name} *(employee)* to the Boca plant.

  ii. Have the Boca location send back the answer.

- ➢b. Find the average salary of all employees.

- b. i. Compute average at New York.

  ii. Send answer to San Jose.

SOLUTIONS

- c. Find the highest-paid employee at each of the following sites: Toronto, Edmonton, Vancouver, Montreal.
  - c. i. Send the query to find the highest salaried employee to Toronto,Edmonton, Vancouver, and Montreal.
  - ii. Compute the queries at those sites.
  - iii. Return answers to San Jose.
- d. Find the lowest-paid employee in the company.
  - d. i. Send the query to find the lowest salaried employee to NewYork.
    ii. Compute the query at New York.
    iii. Send answer to San Jose.

Example:

- Relation-Site matrix gives details about allocation of relation and sites.
- eg. Relation R1 is stored in S1, S2, S3, S5, S6, S8, S10 and it is assumed that R1 is replicated entirely not in fragmented or partitioned form.
- If a user or application poses a query on the DDBS, it is critical to identify the relevant sites of a relation
- The first step in the query retrieval of results is to identify the sites on which particular relation are stored.

QUERY PROCESSING

- **Q1:**

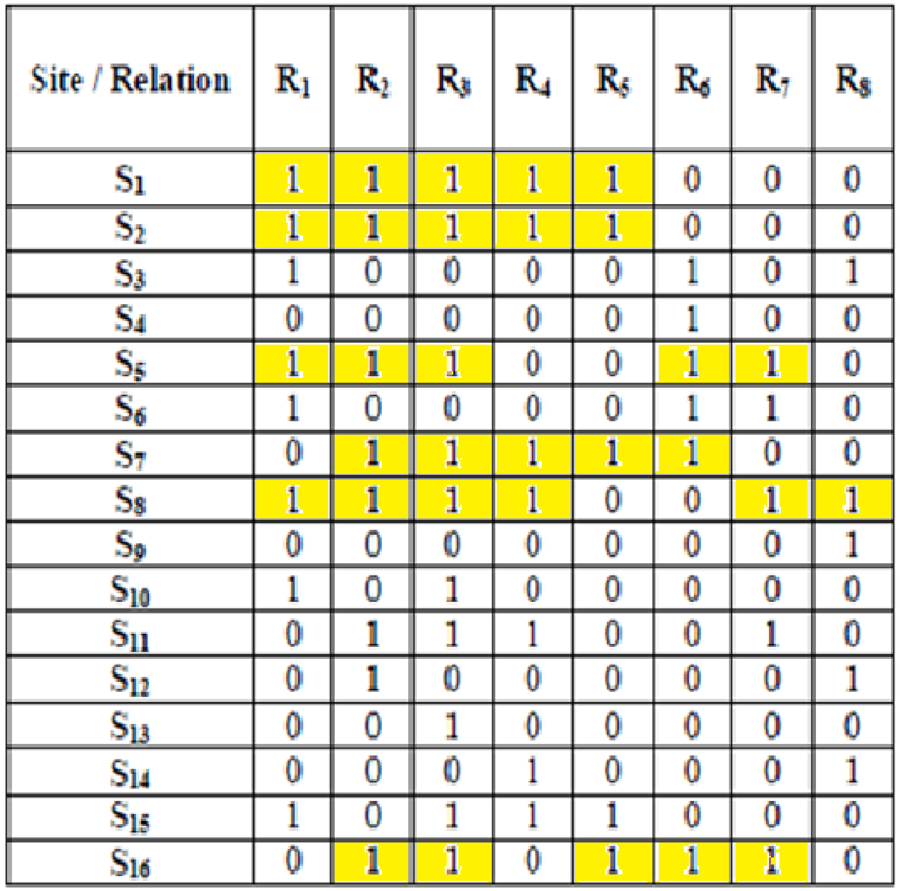
SELECT a, m
FROM R_1, R_2, R_3, R_4, R_5, R_6, R_7, R_8,
WHERE R_1.a=R_4.t
AND R_4.p=R_2.x **AND** R_1.a==R_7.q
AND R_2.x=R_3.n **AND** R_4.x=R_5.s
AND R_8.w=R_6.d **AND** R_7.j=R_6.k

- A query plan represents the set of data sites on which relevant data is stored and result will be aggregated.
- In a query plan [1,1,2,2,2,3,5,3], relations R1 and R2 are accessed from site S1, relations R3, R4 and R5 are accessed from site S2, relations R6 and R8 are accessed from site S3 and relation R7 is accessed from site S5.

QUERY PROCESSING

BELGIUM CAMPUS iTversity
It's the way we're wired

Eg Relation R1 is stored in S1, S2, S3, S5, S6, S8, S10



Site / Relation	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8
S_1	1	1	1	1	1	0	0	0
S_2	1	1	1	1	1	0	0	0
S_3	1	0	0	0	0	1	0	1
S_4	0	0	0	0	0	1	0	0
S_5	1	1	1	0	0	1	1	0
S_6	1	0	0	0	0	1	1	0
S_7	0	1	1	1	1	1	0	0
S_8	1	1	1	1	0	0	1	1
S_9	0	0	0	0	0	0	0	1
S_{10}	1	0	1	0	0	0	0	0
S_{11}	0	1	1	1	0	0	1	0
S_{12}	0	1	0	0	0	0	0	1
S_{13}	0	0	1	0	0	0	0	0
S_{14}	0	0	0	1	0	0	0	1
S_{15}	1	0	1	1	1	0	0	0
S_{16}	0	1	1	0	1	1	1	0

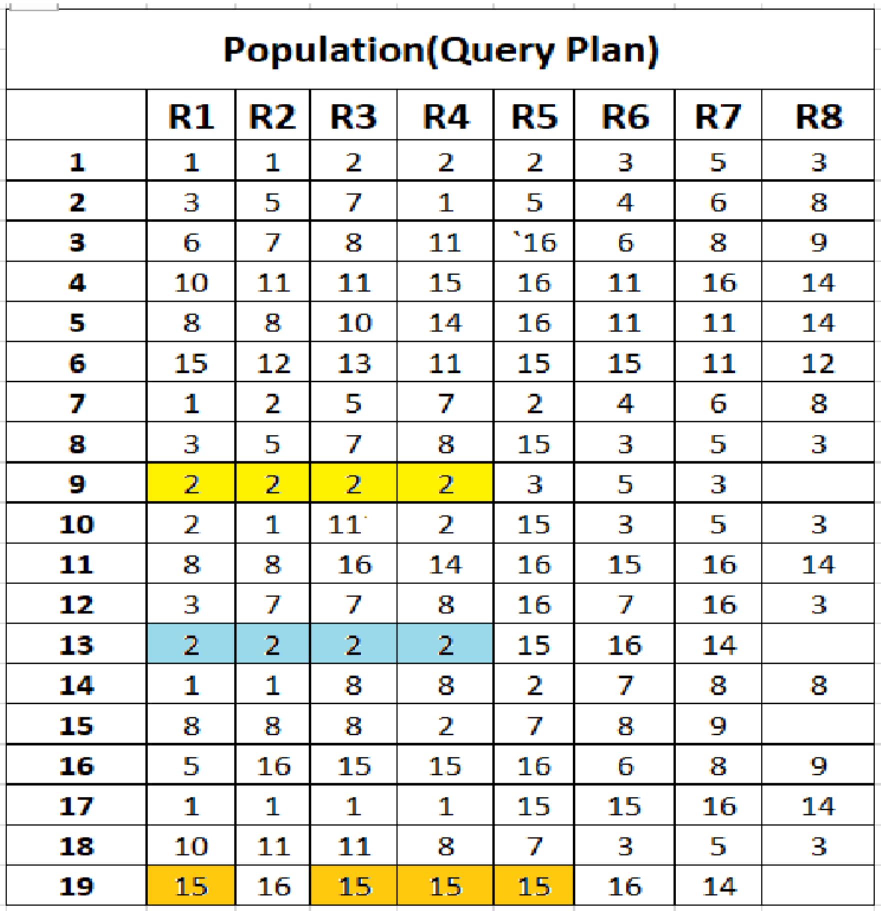
Relation-Site matrix

8 database relations among 16 sites

In a query plan [1,1,2,2,2,3,5,3], relations R1 and R2 are accessed from site S1, relations R3, R4 and R5 are accessed from site S2, relations R6 and R8 are accessed from site S3 and relation R7 is accessed from site S5.

A plan that involves less or may be least number of sites is measured better than the other alternatives.

Sites with higher concentration of same relations is considered better.



Population(Query Plan)

	R1	R2	R3	R4	R5	R6	R7	R8
1	1	1	2	2	2	3	5	3
2	3	5	7	1	5	4	6	8
3	6	7	8	11	`16	6	8	9
4	10	11	11	15	16	11	16	14
5	8	8	10	14	16	11	11	14
6	15	12	13	11	15	15	11	12
7	1	2	5	7	2	4	6	8
8	3	5	7	8	15	3	5	3
9	2	2	2	2	3	5	3	
10	2	1	11	2	15	3	5	3
11	8	8	16	14	16	15	16	14
12	3	7	7	8	16	7	16	3
13	2	2	2	2	15	16	14	
14	1	1	8	8	2	7	8	8
15	8	8	8	2	7	8	9	
16	5	16	15	15	16	6	8	9
17	1	1	1	1	15	15	16	14
18	10	11	11	8	7	3	5	3
19	15	16	15	15	15	16	14	

QPlan decided based on the size of relations stored in participating site and communication cost is evaluated based on the tuples selected during local processing.

COSTS

- **Query Affinity Cost (QAC):**
  1. This is first design objective; it indicates the **degree of heterogeneity** in a QEP on the number of sites accessed during result generation for given user query.
  2. For a given user query, a query equivalent plan that involves less or may be least number of sites is measured better than the other alternatives.
  3. If a QEP is accessing similar sites for relevant relation for result generation of a user query it is better than a QEP in which more sites are accessed for the same set of relations. A plan having sites with higher concentration of same relations is considered better.
- **Query Localization Cost (QLC):** The ‘Query Localization’ indicates two design objectives,
  1. it quantifies the degree of communication between two different sites in a query plan (QP)
  2. It plays crucial role on deciding the control site for query answering of respective QP.
  3. The control sites for query plan execution is decided based on the size of relations stored in participating site and communication cost is evaluated based on the tuples selected during local processing.
- **Local Processing Cost (LPC):**
  1. LPC, is primarily concern by database relations stored on a local site and selectivity of database operators.
  2. various relational operators such as Selection, Projection etc., quantify the value of LPC of a QEP.
  3. The operator selectivity is the measure of LPC.
  4. LPC is dependent on number of memory accesses or memory fetch for transferring set of tuples from secondary memory to main memory, while retrieving data for local relations.

COSTS

- Components of LPC,
    1. local processing computation on relations at remote sites
    2. local processing computation on control site(for final result preparation).
- The second component LPC is important as, final result is integrated at the control sites and supplied to the user in desired structure.

Characterization of Query Processors

- Characteristics of query processors that can be used as a basis for comparison between Centralized and Distributed query processors
    1. Languages:
    2. Types of Optimization
    3. Optimization Timing
    4. Statistics
    5. Decision Sites
    6. Exploitation of the Network Topology
    7. Exploitation of Replicated Fragments
    8. Use of Semijoins

Languages

- In a centralized The input language to the query processor is based on relational calculus and decomposition to object algebra may also needed
- In a distributed context,
  - ➢The output language is generally some internal form of relational algebra augmented with communication primitives.
  - ➢The operators of the output language are implemented directly in the system.
  - ➢ Query processing must perform efficient mapping from the input language to the output language

Dynamic versus Static Optimization

- Two choices of decomposing and optimizing, when first three phases of QP can be carried out:
  - –dynamically every time query is run.
  - –Statically when query is parsed, validated and optimized once after submission.
- **Advantages** of dynamic QO arise from fact that information required to select an optimum strategy is up-to-date.
- **Disadvantages** are that performance of query is affected, time may limit finding optimum strategy.
- **Advantages of static QO** are removal of runtime overhead, and more time to find optimum strategy.
- **Disadvantages** arise from fact that chosen execution strategy may no longer be optimal when query is run.
- Could use a hybrid approach to overcome this.

Types of Optimization

- Query optimization aims at choosing the “best” point in the solution space of all possible execution strategies.
- An immediate method for query optimization is to search the solution space, exhaustively predict the cost of each strategy, and select the strategy with minimum cost
- The problem is that the solution space can be large;
- In a distributed environment the number of relations or fragments increases.
- An “exhaustive” search approach is often used whereby (almost) all possible execution strategies are considered
- To avoid the high cost of exhaustive search, randomized strategies, such as
    1. iterative improvement
    2. simulated annealing
    3. use of heuristics,

- In both centralized and distributed systems, a common heuristic is to minimize the size of intermediate relations.

Simulated Annealing and Iterative Improvement

1. Concentrate on searching for the optimal solution around some particular points.
2. Do not guarantee that the best solution is obtained,
3. Avoid high cost of optimization, in terms of memory and time consumption.

Optimization Timing

- A query is optimized at different times relative to the actual time of query execution.
    1. compilation time (Static QO) before executing the query)
    2. **During** execution (dynamic QO) as the query is executed
    3. Hybrid query optimization

Static query optimization

- Static query optimization is done at query compilation time. .
  - ❑Since the sizes of the intermediate relations of a strategy are not known until run time, they must be estimated using database statistics.
  - ❑Errors in these estimates can lead to the choice of suboptimal strategies.

DYNAMIC QUERY OPTIMIZATION

- Dynamic query optimization proceeds at query execution time.
- At any point of execution, the choice of the best next operator can be based on accurate knowledge of the results of the operators executed previously.
- Main advantage over static query optimization is that the actual sizes of intermediate relations are available to the query processor, thereby minimizing the probability of a bad choice.
- Main shortcoming is that query optimization, an expensive task, must be repeated for each execution of the query.
- Approach is best for ad-hoc queries.

Hybrid query optimization

- Hybrid query optimization attempts to provide the advantages of static query optimization while avoiding the issues generated by inaccurate estimates.
- The approach is basically static, but dynamic query optimization may take place at run time when a high difference between predicted sizes and actual size of intermediate relations is detected.

Statistics

- Effectiveness of query optimization relies on statistics on the database.
- Dynamic query optimization requires statistics in order to choose which operators should be done first.
- Static query optimization more demanding since the size of intermediate relations must also be estimated based on statistical information.
- In distributed databases, statistics for query optimization typically bear on fragments, and include fragment cardinality and size as well as the size and number of distinct values of each attribute.
- The accuracy of statistics is achieved by periodic updating.
- With static optimization, significant changes in statistics used to optimize a query might result in query re-optimization.

Decision Sites

- When static optimization is used, either a single site or several sites may participate in the selection of the strategy to be applied for answering the query.
    1. use the centralized decision approach, in which a single site generates the strategy.
    2. Decision process could be distributed among participating sites
- The centralized approach is simpler but requires knowledge of the entire distributed database, while the distributed approach requires only local information.
- Hybrid approaches → one site makes the major decisions and other sites can make local decisions

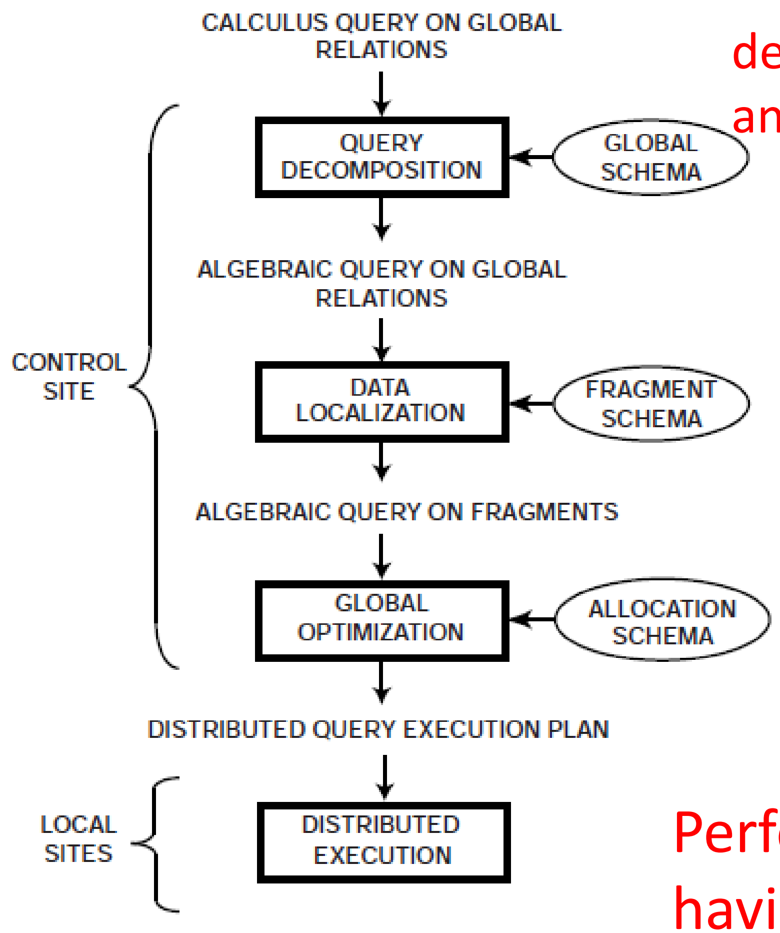
Comparison between Centralized and Distributed query processors

Characteristics	Centralized	Distributed
Languages	relational calculus and decomposition to object algebra	Relational algebra augmented with communication primitives.
Types of Optimization	Use Heuristic that minimize the size of intermediate relations.	Use Heuristic that minimize the size of intermediate relations.
Optimization Timing		
Statistics	statistics for query optimization bear on Relations and , Relation cardinality	Statistics for query optimization bear on fragments, fragment cardinality, size and number of distinct values of each attribute
Decision Sites	centralized decision approach	Hybrid approaches Decision process could be distributed among participating sites
Exploitation of the Network Topology	-	Topology that restrict data communication cost (intersite Communication)
Exploitation of Replicated Fragments		Localize replicated Fragments at run time to minimize communication times

Layers of Query Processing

- The problem of query processing can itself be decomposed into several subproblems, corresponding to various layers
- Each layer solves a well-defined subproblem
- Four main layers are involved in distributed query processing
    1. query decomposition,
    2. data localization,
    3. global query optimization
    4. Distributed Query Execution
- first three layers map the input query into an optimized distributed query execution plan.
- They are performed by a central control site and use schema information stored in the global directory
- The fourth layer performs distributed query execution by executing the plan and returns the answer to the query and is done by the local sites and the control site

Layers of Query Processing



decompose the calculus query into an algebraic query on global relations.

localize the query's data using data distribution information in the fragment schema

find an execution strategy for the query which is close to optimal.

Performed by all the sites having fragments involved in the query

Query Decomposition

- The first layer decomposes the calculus query into an algebraic query on global relations.
- The information needed for this transformation is found in the global conceptual schema describing the global relations.
- The information about data distribution is not used here but in the next layer.
- The techniques used by this layer are those of a centralized dbms.
- Query decomposition can be viewed as four successive steps.

Query decomposition Steps

1. The calculus query is rewritten in a normalized form that is suitable for subsequent manipulation and involves the manipulation of the query quantifiers and qualification by applying logical operator priority.
2. The normalized query is analyzed semantically so that incorrect queries are detected and rejected as early as possible.
3. The correct query (still expressed in relational calculus) is simplified by eliminating redundant predicates.
4. The calculus query is restructured as an algebraic query

Data Localization

- The main role of the second layer is to localize the query’s data using data distribution information in the fragment schema
- It determines which fragments are involved in the query and transforms the distributed query into a query on fragments.
- Fragmentation is defined by fragmentation predicates that can be expressed through relational operators.
- A global relation can be reconstructed by applying the fragmentation rules, and then deriving a program, called a localization program, of relational algebra operators, which then act on fragments.

Data Localization

- Generating a query on fragments is done in two steps.
    1. The query is mapped into a fragment query by substituting each relation by its reconstruction program (also called materialization program),
    2. The fragment query is simplified and restructured to produce another "good" query.
- Simplification and restructuring may be done according to the same rules used in the decomposition layer.
- .

Global Query Optimization

1. The input to the third layer is an algebraic query on fragments.
2. The goal of query optimization is to find an execution strategy for the query which is close to optimal.
3. An execution strategy for a distributed query can be described with relational algebra operators and communication primitives (send/receive operators) for transferring data between sites.
4. An important aspect of query optimization is join ordering,
5. permutations of the joins within the query may lead to improvements thru reducing the size of the join operands and then the communication cost
6. The output of the query optimization layer is an optimized algebraic query with communication operators included on fragments.
7. Query is represented and saved (for future executions) as a distributed query execution plan

Distributed Query Execution

- The last layer is performed by all the sites having fragments involved in the query.
- Each subquery executing at one site, called a local query, is then optimized using the local schema of the site and executed.
- At this time, the algorithms to perform the relational operators may be chosen.

SUMMARY

- The main assumption is that the input query is expressed in relational calculus since that is the case with most current distributed DBMS.
- The goal of distributed query processing may be summarized as follows: given a calculus query on a distributed database, find a corresponding execution strategy that minimizes a system cost function that includes I/O, CPU, and communication costs.
- Query processors may differ in various aspects such as type of algorithm, optimization granularity, optimization timing, use of statistics, choice of decision site(s), exploitation of the network topology, exploitation of replicated fragments, and use of semijoins.
- We have proposed a generic layering scheme for describing distributed query processing.
- Four main functions have been isolated: query decomposition, data localization, global query optimization, and distributed query execution.
- These functions successively refine the query by adding more details about the processing environment.

QUESTIONS

This Photo by Unknown Author is licensed under CC BY

REVISION QUESTIONS

- How can parallelism improve the performance of query processing in a distributed database environment?
- What is the difference between inter-query and intra-query parallelism?
- Explain clearly the terms decomposition and data localization in distributed query processing.
- What factors are used to calculate the total cost of a query in a distributed database system?
- Compare exhaustive search and heuristic types of optimizers.
- State the advantages and disadvantages of static and dynamic query optimization.
- How does hybrid optimization overcome the shortcomings of static and dynamic query optimization?
- Draw a diagram showing the layers of query processing in a distributed database environment.