Dynamic Distributed Database over Cloud Environment

...

A Fragmentation, Allocation and Replication Algorithm

Divija Nagaraju - 14IT112
Aparna P L - 14IT132
Mukta Kulkarni - 14IT220
Pooja MS - 14IT230

Introduction

- Efficiency of database systems can be improved by using distributed methods
- Database fragments located at different geographical positions
- Different sites are managed through a network using a DDBMS
- The functionality of distributed systems depends on fragmentation, allocation and replication strategies used
- Hence, considered as an active research area

Challenges

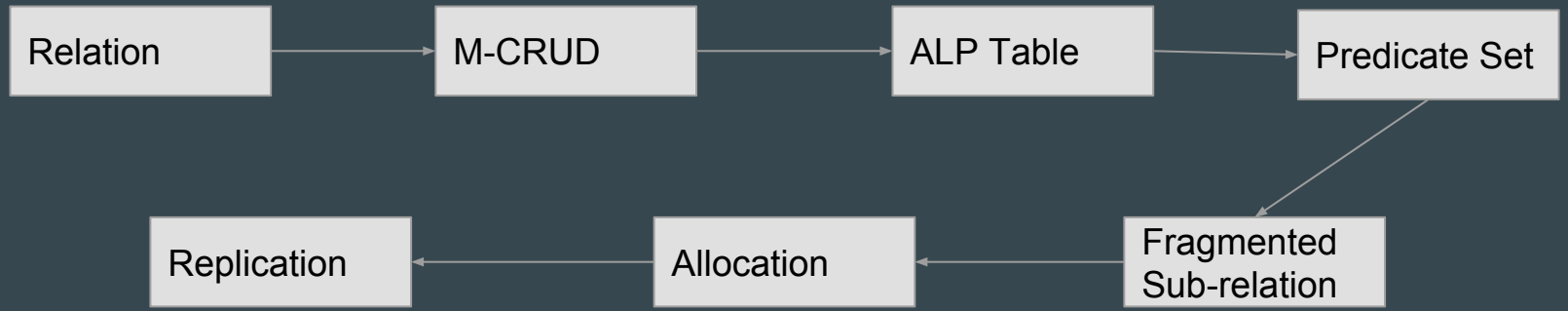
The main challenges facing the DDBS design are:

- How to fragment database tables
- Which type of fragmentation will be used
- When to replicate fragments
- What is the optimal number of replica that can be taken for each fragment
- How to allocate fragments to sites where they are mostly frequently accessed
- Do we group distributed database sites into disjoint clusters
- Which type of clustering needs to be used in such a case

Problem Statement

“Formulation of an enhanced strategy for fragmentation, allocation and replication in a distributed database system that could be extended to a cloud environment.”

Methodology



Methodology

The data to location CRUD matrix is modified to predicate to applications per site form.

New Matrix called: **Modified - Create Read Update Delete Matrix (MCRUD)**

Obtain attribute “A” with high **Attribute Locality Preference (ALP)**

Fragment over **predicates of Attribute “A”.**

Allocate each predicate to a **site with maximum CRUD cost**

Replicate to site with **more Read Operations**, if sites have same CRUD cost

MCRUD and ALP evaluation

Cost per functionality

Create=2

Read=1

Update=3

Delete=2

$$C_{i,j,k,r} = f_C C + f_R R + f_U U + f_D D \quad (1)$$

$$S_{i,j,k} = \sum_{r=1}^{A_{ijk}} C_{i,j,k,r} \quad (2)$$

$$S_{i,j,m} = \text{Max}\,(S_{i,j,k}) \quad (3)$$

$$ALP_{ij} = S_{i,j,m} - \sum_{k \neq m}^{A_{ijk}} S_{i,j,k} \quad (4)$$

$$ALP_i = \sum_{j=1}^{l} ALP_{i,j} \quad (5)$$

Here f_C = frequency of create operation
f_R = frequency of read operation
f_U = frequency of update operation
f_D = frequency of delete operation
C = weight of create operation
R = weight of read operation
U = weight of update operation
D = weight of delete operation
$C_{i,j,k,r}$ = cost of predicate j of attribute i accessed by application r at site k
$S_{i,j,k}$ = sum of all applications' cost of predicate j of attribute i at site k
$S_{i,j,m}$ = maximum cost among the sites for predicate j of attribute i
ALP_{ij} = actual cost for predicate j of attribute i
ALP_i = total cost of attribute i (locality precedence)

Analysis: Sample MCRUD Matrix

Site.Application / Entity.Attribute.Predicates	Site1			Site2			Site3		
	Ap1	Ap2	Ap3	Ap1	Ap2	Ap3	Ap1	Ap2	Ap3
Accounts .Account.No<10000	C		RU						R
Accounts .Account.No>=10000		R							
Accounts.Type=Ind	CRD	RU	RUD		R				
Accounts.Type=Cor		RU	R				CRUD	RU	R
. . .									
Accounts.Balance<50000	R		R			CRUD			R
Accounts.Balance>=50000		CR							
Accounts.BrName=Dhk	CRUD	RU	CRUD			R	R		
Accounts.BrName=Ctg		R		CRUD	CRUD	R		R	
Accounts.BrName=Khl							CRUD	RD	CRU

Analysis: Obtained ALP

Attribute Name	Precedence
AccountNo	6
Type	22
CustId	6
OpenDate	7
Balance	10
BrName	50

Result

Fragment over Attribute “BrName”

Fragment 1: Predicate Dhk

Fragment 2: Predicate Ctg

Fragment 3: Predicate Khl

Fragment Number	Allocated to	Replicated to
Fragment 1	1	3
Fragment 2	2	1
Fragment 3	1	3