Parallel Relational Database Systems

I. Introduction

II. Optimization-Parallelization Strategies (Inter-operation)

III. Efficiency of Parallelism

IV. Optimization of Data Communication

A. Hameurlain et al.

IRIT, Paul Sabatier University
Toulouse, France
I. Introduction to Parallel Rel. DB

1. Motivation [Dew 90, Val 93, Lu 94, ...]

- Relational Languages: Declaratives
  - Regular Data Structures: Static Annotation
  - Relational Language: Declarative
    - Automatic Parallelization
  - Decision Support Queries: Complex, Huge DB, Join, Sort, Aggregation

2. Objectives:

- Best Cost / Performance with respect to Mainframe DPS8 / GCOS, IBM 30390 / VMS, ...)

- High Performance:
  - Minimizes the Response Time
  - Maximizes the Parallel System Throughput

- “Scalability”:
  - Adding New Resources (CPU, Disk, Memory)
  - Adding New Users
    - Holding the Same Performance

- Availability
II. Optimization-Parallelization Strategies (Inter-operation)

1. Introduction

Query Compiler Architecture for Parallel Database Systems

SQL Query

Parser

Optimizer

Relational Schemas

Rule Bases

Logical & Physical Optimization

Parallelizer

Intra-operation

Inter-operation

Code Generation

Proc.1 Proc.2 Proc.n

Fragment1 Fragment2 Fragmentn

Proc1 Proc2 ... Proc_n

Cost Evaluator

SPJ Query Parallelisation:
Parallelism Extraction & Resource Allocation

Fragment = Pipeline Chain of the Execution Plan
A. Parallelism Extraction

1. Data Partitioning: Approaches & Methods [Liv 87, Cop 88, Dew 92]
   - Partitioning Degree of each base relation?

2. Parallelism degrees of Joins?

3. Parallelization Strategies (Inter-operator)
   - Approaches
     
     ⇒ Two-Phase Approach: \( \Phi_1 ; \Phi_2 \)
     
     • XPRS [Hon92, Sto 88], Papyrus [Gan 92, Has 94, Chek 95], Gamma Proj. [Kab 98], ...
     
     ⇒ One-Phase Approach: packs \( \Phi_1 & \Phi_2 \) into one process
     
     • [Sch90, Che92, Zia93, Lan93,...]

\( \Phi_1 \) : Physical Optimization (without considering the resources)
\( \Phi_2 \) : Parallelization: Parallelism Extraction & Resource Alloc.

4. Generation of Parallel Programms:

Query = \( R1xR2xR3xR4xR5xR6xR7xR8 \)

B. Resource Allocation (Mapping)

1. Data (relations) Placement: Alloc_R

2. Tasks (Operator) Placement: Alloc_T
SEQ

PAR

PIPE Scan S1 - Build J1 ENDPIPE
PIPE Scan S3 - Build J2 ENDPIPE
PIPE Scan S5 - Build J3 ENDPIPE
PIPE Scan S7 - Build J4 ENDPIPE
ENPAR;

PAR

PIPE Scan S2 - Probe J1 - Build J5 ENDPIPE
PIPE Scan S6 - Probe J3 - Build J6 ENDPIPE
ENDPAR;

PIPE Scan S4 - Probe J2 - Probe J5 - Build J7 ENDPIPE
PIPE Scan S8 - Probe J4 - Probe J6 - Probe J7 ENDPIPE
ENSEQ

Bushy Tree
III. Efficiency of Parallelism

- Shared-Nothing Architecture
- Rel. Size [Bit 83], & Parameters [Sch 90], [Val 88]

\[
\text{LRT } (T \leftarrow R \bowtie S) = T_{ef} + T_d + T_{com} \quad \text{where}
\]
\[
T_{ef} = (|R|/d).th + (|R|/d).CR + (|R|/d).CJO + |T|.I + ||T||/d.CW
\]
\[
T_d = (|T|/d).th \quad \text{and} \quad T_{com} = (|T|/d).trf + p.msg).\left[\frac{d}{p}\right]
\]

- Simple Hash-Join Algorithm (Build + Probe)
  Build (R) holds in memory

\[
LRT (T \leftarrow R \bowtie S) = T_{ef} + T_d + T_{com} \quad \text{where}
\]
\[
T_{ef} = (|R|/d).th + (|R|/d).CR + (|R|/d).CJO + |T|.I + ||T||/d.CW
\]
\[
T_d = (|T|/d).th \quad \text{and} \quad T_{com} = (|T|/d).trf + p.msg).\left[\frac{d}{p}\right]
\]

- Number of Tuples in R = 10^6  \quad \text{Trf : Time to transfer a tuple}
- Number of Pages in R  \quad \text{msg : Time to process a message}
- Time to hash a tuple (200 B) = 8 ms  \quad \text{CPU = 4 MIPS}
- Time to read 1 page (18KB) = 8 ms
- Time to write 1 page = 16 ms
- Time for joining 2 unsorted pages
- Number of proc. of source operation
- Number of proc. of destination operation
**Efficiency of Parallelism:**

- **Intra-Operation with Lower NB of Processors**
- **Pipeline with Large NB of Processors**

➤ **The Plague of Parallelism:** Cost of Data Communication
IV. Optimization of Data Communication

1. Logical Optimization: JSP --> PSJ (Reducing the Vol. of Data)

2. Physical Optimization: the order in which the joins are executed

3. Parallelization Phase:
   Cost of Tuple Redistributing

A Simple SQL Query and Associated Data Flow Graph

Methods: Tree Coloring [HAS 95] Propagation Method [Ham 93]

- Partitioning Attributes &
- Number of Processors

Propagation Method: Partition Attribute & Number of Processors