SOS10 Panel Session: Challenges in Data-Intensive Computing

Bill Blake
Maui, 07 March 2006
Peta-Scale Data-Intensive Computing is a Reality in Commercial IT shops Today

• This is driven by the need to understand customers and manage the business “at the sub-transaction level”

• Examples:
  > Wireless Telephone Companies
  > Web Stores
  > Credit/Financial Analysis
  > Retailers
  > ISPs
The “Within Application” Approach

• Business Intelligence demands drive the industry towards specialized database/storage blades
  > focus of virtualization centered on the analytic DB

• Systems will be highly-specific
  > Tuned to meet the performance needs of applications versus general purpose “virtualization”
Data Flow – The Traditional Way

Applications → Client → Processing → SAN → Storage

- SMP HOST 1
- SMP HOST 2
- SMP HOST N

Local Applications

Hours or Days
What is Needed Now?

Well, if moving all the data to the processors doesn’t do the job…

Then why not move the processors to where the large data resides?

It is hard to do this in a data-intensive way when not working within the application
Streaming Data Flow

BI Applications → Client → Netezza Performance Server

- ODBC 3.X
- JDBC Type 4
- SQL/92

Local Applications

Fast Loader/Unloader

SPU

FPGA

Streaming data, joins, & aggs @ 50MB/sec

Up to 120 TB/hour Cross Section BW per 8650 system

Bulk data movement: 250 GB/hour - uncompressed (1 TB/hour Target)
Netezza Found Clues In Late ’90s Computer Science Research

• **Active Disk architectures**
  > Integrated processing power and memory into disk units
  > Scaled processing power as the dataset grew

• **Decision support algorithms offloaded to Active Disks** to support key decision support tasks
  > Active Disk architectures use stream-based model ideal for software architecture of relational databases

*In Netezza’s NPS® System: “Snippet Processing Units” take streams as inputs and generate streams as outputs*
Netezza Performance Server

Gigabit Ethernet

Snippet Processing Unit (SPU)

- Snippet Queue
- PowerPC Query Engine
  - Joining
  - Sorting
  - Grouping
- Main Memory
- Streaming Record Processor
- Disk Control
- Transaction/Lock Manager
- Replication Manager
- Primary
- SPU Swap
- Mirror
- Dual Disk

A Closer Look Inside
Netezza Performance Server

Netezza added:
- Highly optimized query planning
- Code generation
- Stream processing

Result: 10X to 100X performance speedup over existing systems
Asymmetric Massively Parallel Processing® Architecture

Netezza Performance Server® System

- SQL Compiler
- Query Plan
- Optimize
- Admin

Execution Engine

Front End

DBOS

SMP Host

Gigabit Ethernet

Massively Parallel Intelligent Storage

Source Systems
- ETL Server
- DBA CLI
- 3rd Party Apps
- High Performance Loader

Client
- SOLARIS
- AIX
- TRU64
- HP-UX
- WINDOWS
- LINUX

ODBC 3.X
JDBC Type 4
SQL/92

High-speed Loader/Unloader

Snippet Processing Unit (SPU)
- Processor & streaming DB logic

High-Performance Database Engine
- Streaming joins, aggregations, sorts, etc.

1000+

DBOS Front End

1
2
3

11

High Performance Loader
Binary Compiled Queries Executed on Massively Parallel Grid

```c
void GenPlan1(CPlan *plan, char *bufStarts, char *bufEnds, bool lastCall) {
    // Setup for next loop (nodes 00..07)
    // node 00 (TScanNode)
    TScanNode *node0 = (TScanNode*)plan->m_nodeArray[0];
    // For ScanNode:
    TScan0 *Scan0 = BADPTR(TScan0*);
    CTable *tScan0 = plan->m_nodeArray[0]->m_result;
    char *nullsScan0P = BADPTR(char *);
    // node 01 (TRestrictNode)
    TRestrictNode *node1 = (TRestrictNode*)plan->m_nodeArray[1];
    // node 02 (TProjectNode)
    TProjectNode *node2 = (TProjectNode*)plan->m_nodeArray[2];
    // node 03 (TSaveTempNode)
    TSaveTempNode *node3 = (TSaveTempNode*)plan->m_nodeArray[3];
    // For SaveTemp Node:
    TSaveTemp3 *SaveTemp3 = BADPTR(TSaveTemp3*);
    CTable *tSaveTemp3 = node3->m_result;
    CRecordStore *recStore3 = tSaveTemp3->m_recStore;
    // node 04 (THashNode)
    ...
    ...
It’s All About Scaling, Streaming and Asymmetry

- Sandia: TeraFLOP → PetaFLOP
  > Specialized Node Function
  > Linux + light weight kernels
  > System Interconnection is “secret sauce” for high BW low latency MPP performance gains

- Netezza: TeraByte → PetaByte
  > Specialized Node Function
  > Linux + light weight kernels
  > Storage/processor/DB integration is “secret sauce” for streaming query processing MPP perf gains
Thank You