Towards a “scaling-in” approach for data processing

Jignesh M. Patel
University of Wisconsin
Azure G5
32 Cores
448GB RAM
$8.69/hr

~2006

Azure D15_v2
20 Cores
140GB RAM
$1.253/hr

2016

Image credits: https://openclipart.org/detail/20542/dell-1u-server and https://openclipart.org/detail/216936/dell-2u
Scope: BI analytics

Data Warehouse

Efficiently

Everywhere

Effortlessly
Efficiently

Everywhere

Effortlessly

In-memory is really “in”

2016 and beyond
Efficiently
On modern hardware
many cores, large memory configurations

Effortlessly
Azure: Pricing vs Configuration (Series: D, F)
Bubble volume = #cores

Everywhere

$/core/hour
0 0.2 0.4 0.6 0.8

GB/core
0 2 4 6 8 10 12 14 16

F16 16 16
D5v2 20
D15v2 32
G5
Efficiently

Everywhere

Effortlessly

many cores, large memory configurations

infinite and instant elasticity

Credits: https://openclipart.org/detail/29048/recepticle
Efficiently
On modern hardware
many cores, large memory configurations

Everywhere
Containers of any size
infinite and instant elasticity

Effortlessly
Auto-tuning
a true utility model
Storage Manager Design

A database = A collection of tables
A table = set of blocks
A block = bag of tuples
A block = a mini self-contained database. Only tuples from one table per block.
A sub-block = a storage organization within a block e.g. row sub-block, column sub-block, or index sub-block.

Accessibility to a block’s data wrapped by a template metaprogramming layer.
SELECT * FROM R, S
WHERE R.b > 10
AND S.c > 100
AND R.a = S.a
Query Execution

Clean Separation of Data Flow and Control

The “traditional” way

SELECT * FROM R, S
WHERE R.b > 10
AND S.c > 100
AND R.a = S.a

The Quickstep way

Scheduler Queue:
\[ \sigma (s1) \sigma (r2) \sigma (r1) \]

Buffer Pool
\[ r1 \ s1 \ s2 \ s3 \ s4 \ r' \ s' \ s'' \]

Advantages
+ Cleaner Abstraction
+ Dynamic Optimization
+ Better p9X
+ Manageability and Debug-ability

Pool of Worker Threads

TMB
SSB 10GB on 4 core container

Execution Time (seconds)

Query

Q1.1  Q1.2  Q1.3  Q2.1  Q2.2  Q2.3  Q3.1  Q3.2  Q3.3  Q3.4  Q4.1  Q4.2  Q4.3

- Quickstep
- MonetDB
- PostgreSQL 9.6 (β1)
- Spark
Priority scheduling: Example of elastic behavior
Efficiently
Bare metal performance on core data kernels, and on any hardware

Everywhere
Elasticity for inter and intra-operator parallelism, even for complex operators

Effortlessly
Unified memory management, hybrid storage formats, learning-based scheduler/optimizer

An experimental data platform for the cloud-native data world
www.quickstep.io
Thanks!