Load Testing for Performance: Background & DB2 Experiences

Matt Emmerton
IBM Canada

Agenda

Introduction

Key Terms & Processes

Application to TPC-C and SAP SD

Conclusion

Agenda

Introduction

Key Terms & Processes

Application to TPC-C and SAP SD

Conclusion

Metrics

- Two typical metrics
  - Throughput (transactions per second)
  - Response Time (seconds per transaction)
- Measure different aspects of performance
  - Problems require different solutions

Bottlenecks

- Bottlenecks are performance-limiting factors
- Challenges
  - Complicated environments have many “layers”
  - Each has its own set of factors with different causes and resolutions
- Detection Strategies
  - Top-down/bottom-up (OS/application)
  - Inside-out/outside-in (client/server)
- Detection Methods
  - Observation : continuous **targeted** monitoring
    - Needs to know how things behave today so that you can assess how workload changes tomorrow will affect the system
  - Analysis : computed from system design parameters
    - Needs to refine system design parameters with limitations observed in practice

Projections

- How do you project future load?
  - Linear extrapolation
    - Ignores effects of scalability
    - Provides best-case scenario, but inaccurate
  - Full-scale measurement
    - Example: Double the system, double the load
    - Provides actual-case scenario, but costly
  - Reduced-scale measurement
    - Example: Take a slice of the system and increase the load
    - Provides average-case scenario, cost-effective
    - May not expose the full complement of problems
    - If improperly designed can expose “phantom” issues

Scalability

- The property of being able to “scale” (increase) the capabilities of a system while maintaining the same performance characteristics
- Not as easy as it sounds!
  - Non-linear scalability is typical
  - Many subsystems experience exponential response time growth under load ➔ poor scalability

Subsystem Scalability (I)

- Some sub-systems can scale linearly
  - Network / Storage
    - Bandwidth / request rates typically scale linearly
    - Latency can go exponential if infrastructure unable to handle request rates

- Scaling issues found here are typically “simple” to solve
  - Hardware upgrades to achieve bandwidth and/or parallelism

Subsystem Scalability (II)

- Some sub-systems do not scale linearly
  - Processors / Memory
    - Physical capacity
    - NUMA effects
  - Software
    - Critical sections (locks, atomics)
    - Single-threaded subsystems
- While they do scale linearly in capacity, they do not scale linearly in terms of efficiency
- Scaling issues found here are typically “complex” to solve
  - Require re-architecture or re-design of systems and/or software
- **Obtaining good scalability in these sub-systems is key to large system scalability**

NUMA (Non-Uniform Memory Access)

- System design where not all memory accesses have the same latency
- Historically used to characterize main memory access
  - But shared L2/L3 memory can also exhibit NUMA effects
- What drives memory access latency?
  - Physical distance
    - Number of “hops”, memory controller interaction, system buses, high-speed interconnects, etc.
  - Degree of concurrent access
    - Impacts cache coherency processing
- Latency also depends on type of operation
    - Read and write access suffer from physical distance effects
    - Write operations suffer from concurrency effects

Agenda

Introduction

Key Terms & Processes

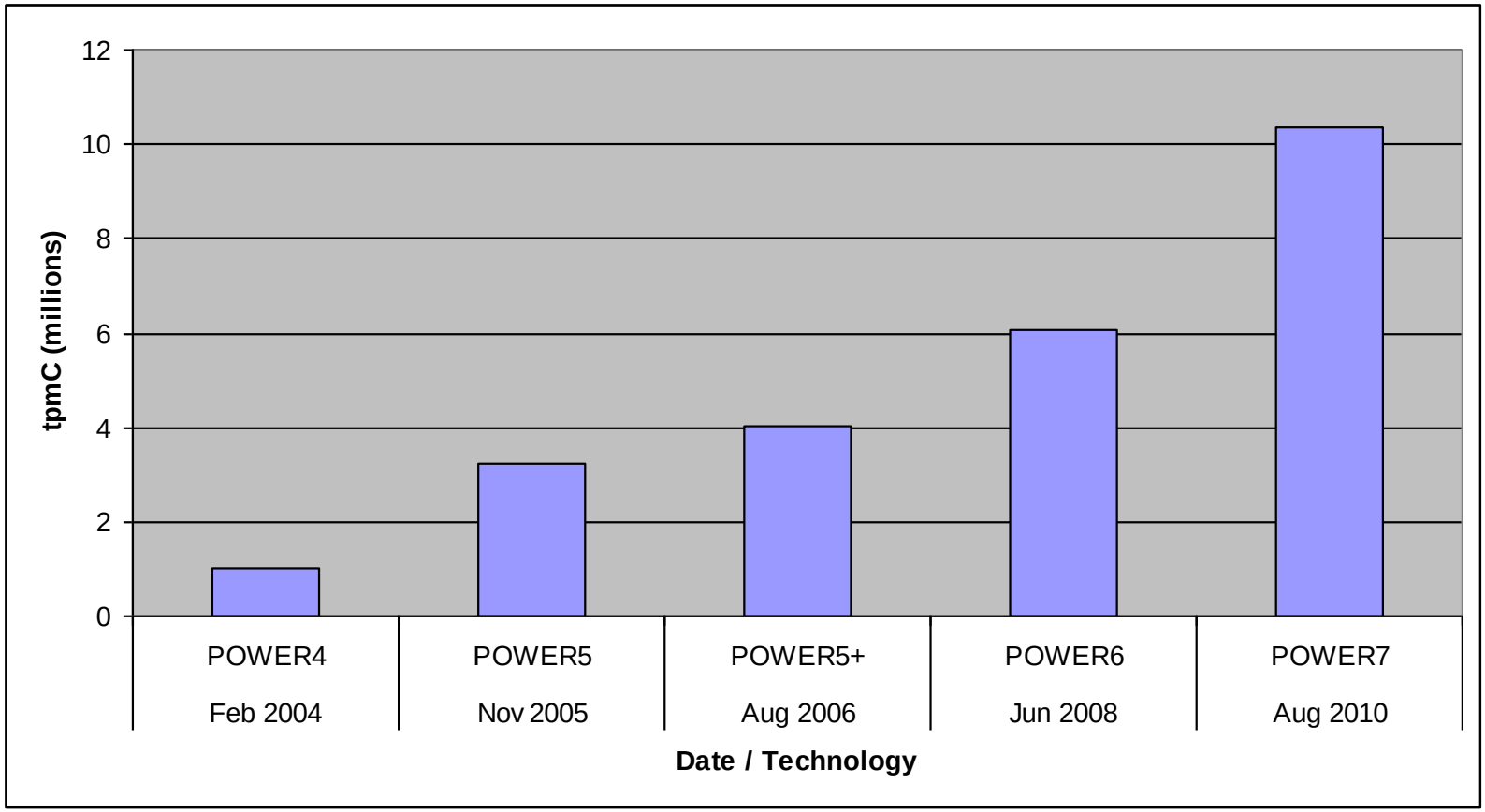
Application to TPC-C and SAP SD

Conclusion

TPC-C

- OLTP benchmark developed by the TPC
  - Released in 1991
  - Results still being published today!
- A very simple model of an order-entry system
  - 11 tables, 13 indexes
  - 5 transactions
    - 3 read-write (new-order, payment, delivery) ➔ 92% of workload
    - 2 read-only (order-status, stock-level) ➔ 8% of workload
- Very scalable
  - Entire schema is based on warehouses
  - Easily partitionable
    - Beneficial for clustered database implementations
    - Beneficial for NUMA systems

TPC-C over the years



Legal Notices

- TPC-C and tpmC are trademarks of the Transaction Processing Performance Council.
- Source: Transaction Processing Performance Council (TPC), as of November 18, 2013.
- Source: http://www.tpc.org/1690, http://www.tpc.org/1641, http://www.tpc.org/1723, http://www.tpc.org/1751, http://www.tpc.org/1777.

How do we scale TPC-C?

Analyse

- Linear Extrapolation

Observe/Analyse

- Reduced-Scale Environment

Observe/Analyse

- Full-Scale Environment
- Achieve Performance Target

TPC-C Experience: POWER4

Analyse - Linear Extrapolation

- Found numerous problems
  - Largest object would be too big to fit in DB2 tablespace
  - Some B-Tree indexes would be 5 or 6 levels
    - Overhead of maintaining B-Tree indexes
- Solved with additional DB2 capabilities
  - Improved UNION ALL VIEW support
    - Allowed updatable views over many smaller/tables indexes
    - Views had to be updatable due to TPC-C transparency requirements
  - Later releases had customer-friendly enhancements
    - Increased tablespace size limits
    - Increased table size limits
    - Range-partitioned tables and indexes
  - Range Clustered Tables
    - New table type that provides O(1) access for primary key lookups
    - Minimizes number of indexes that need to be maintained (13 down to 2)

TPC-C Experience: POWER4

Analyse/Observe - Reduced-Scale Environments

64% & 75% target database sizes

- Identified bottlenecks in I/O subsystem
  - Physical disk / controller infrastructure
    - Changed to different storage technology
  - AIO subsystem
    - Made changes in AIX
  - DB2 page cleaner subsystem
    - Invented “alternate” page cleaning mechanism
- Identified bottlenecks in software
  - DB2 transaction manager
  - DB2 lock manager
    - Adjusted critical sections to improve scalability
- Identified bottleneck from poor page caching
  - Doubled system memory

TPC-C Experience: POWER4

Analyse/Observe - Full-Scale Environment

90% target database size

- Final configuration
  - Didn't quite hit our performance / database size target
- Found and fixed many small issues in AIX and DB2
  - Fine-tuning to alternate page cleaning algorithm
  - Fine-tuning to AIX and DB2 NUMA exploitation capabilities

TPC-C Experience: POWER7

Analyse - Linear Extrapolation

- Found one major problem
  - Capability of single-threaded logger

Analyse/Observe - Reduced-Scale Environment

- Started with an 8-node cluster (1/12 of final size)
  - Found and fixed numerous small issues

Analyse/Observe - Full-Scale Environment

- Found and fixed issues specific to clustered environments
  - Excessive communication
    - Rollbacks sent to all nodes even if they didn't participate in transaction
    - Non-local transactions sent to all nodes even though we had proper subset routing data

SAP SD

- OLTP benchmark developed by SAP
  - Released in 1993
  - Tests the Sales & Distribution (SD) components of SAP R/3
- Runs on top of an actual SAP R/3 installation
  - Has a very complex schema and transactions
  - Workload consists of multi-step business transactions that have multiple application/database round-trips and database transactions each
- Very scalable
  - Most of the schema is based on company
  - Easily partitionable
    - Beneficial for clustered database implementations
    - Beneficial for NUMA systems

SAP SD Experience: POWER7+

Analyze - Linear Extrapolation

- Bottleneck: Atomic generation of log sequence numbers (LSNs)
  - Throughput / response time impacted by NUMA latency
- Solution
  - Redesign for throughput & response time
    - Parallelized the process of generating log sequence numbers (LSNs)
    - Implemented by new threads (“db2lragen”)
  - Improved throughput
    - No longer a serial process
  - Improved response time
    - Affinitization of producers/consumers reduced NUMA latency impact

Analyse/Observe - Full-Scale Environment

- This solution worked well and readily scaled past the known limits of the prevoius mechanism
- Were able to expose additional problems that otherwise would have been masked

Conclusions

- Constantly evaluate against key metrics
  - Throughput, Response Time
- Understand your environment
  - Be aware of potential bottlenecks and scaling points
- Project and Test
  - Validate the assumptions against reality where possible
- Repeat!