Co-funded by the
Erasmus+ Programme
of the European Union

DeCAIR

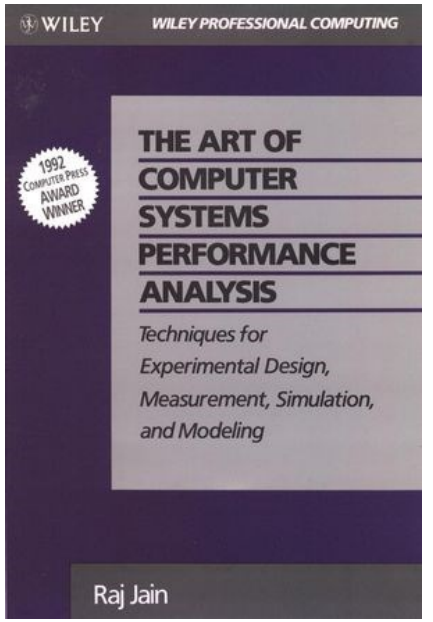
Workloads

Prof. Gheith Abandah

Developing Curricula for Artificial Intelligence and Robotics (DeCAIR)
618535-EPP-1-2020-1-JO-EPPKA2-CBHE-JP

References

- Raj Jain, **The Art of Computer Systems Performance Analysis**, Wiley, 1991.
  - Part I: An Overview of Performance Evaluation
  - Part II: Measurement Techniques and Tools
  - Part III: Probability Theory and Statistics
  - Part IV: Experimental Design and Analysis
  - Part V: Simulation



Outline

- Terminology
- Types of Workloads
  1. Single Operation
  2. Mixture of Operations
  3. Kernels
  4. Synthetic Programs
  5. Application Benchmarks
- Workload Selection
- Workload Characterization
- Monitors

Terminology

- **Test workload**: Any workload used in performance studies; it can be **real** or **synthetic**.
- **Real workload**: Observed on a system being used for normal operations.
- **Synthetic workload**
  - Synthesized to behave like real workload
  - Can be applied repeatedly in a controlled manner
  - May have built-in measurement capabilities
  - Does not have large, real-world, sensitive data files
  - Can be easily modified to scale the workload
  - Can be easily ported to different systems due to its small size

Test Workloads for Computer Systems

1. Single Operation
2. Mixture of Operations
3. Kernels
4. Synthetic Programs
5. Application Benchmarks

1. Single Operation

Example: Addition Instruction

- Processors were the most expensive and most used components of the system.
- Addition was the most frequent instruction.
- One operation can be used for fast and simple evaluation.

2. Mixture of Operations

Example: Instruction Mixes

- Instructions + usage frequency
- **Gibson mix**: Developed by Jack Gibson in 1959 for IBM 704 systems.
- **Performance Metrics**
  - **MIPS** = Millions of instructions per second
  - **MFLOPS** = Millions of floating-point operations per second

1.	Load and Store	31.2
2.	Fixed-Point Add and Subtract	6.1
3.	Compares	3.8
4.	Branches	16.6
5.	Floating Add and Subtract	6.9
6.	Floating Multiply	3.8
7.	Floating Divide	1.5
8.	Fixed-point Multiply	0.6
9.	Fixed-point Divide	0.2
10.	Shifting	4.4
11.	Logical, And, Or, etc.	1.6
12.	Instructions Not Using Registers	5.3
13.	Indexing	18.0
	Total	100.0

3. Instruction Mixes

Disadvantages

1. Complex classes of instructions are not reflected in the mixes.
2. Does not account for instruction time variance due to:
   - Various addressing modes
   - Cache hits and misses
   - Interference from other devices
   - *etc*.

3. Kernels

- Kernels are **short functions that are frequently executed**.
- **Examples**
  - Tree search
  - Sorting
  - Matrix multiplication
- **Advantage**: Exercise the processor and memory
- **Disadvantage**: Do not exercise the I/O devices

4. Synthetic Programs

- Programs to **exercise various aspects** of the computer system.
- To measure I/O performance, analysts developed **Exerciser Loops**.
- The first exerciser loop was by Buchholz (1969) who called it a synthetic program.

4. Synthetic Programs

- **Advantages**
  - Quickly developed
  - No real data files
  - Easily modified and ported to different systems
  - Have built-in measurement capabilities
- **Disadvantages**
  - Too small
  - Do not make representative memory or disk references
  - Complex interactions such as page faults and disk cache may not be adequately exercised
  - CPU-I/O overlap may not be representative

3. Application Benchmarks

- **Real applications selected and packaged for efficient and standard performance evaluation**.
- **Example Benchmarks**
  - **SPEC**: CPU
  - **TPC**: Servers
  - **Heaven UNIGINE**: GPU
  - **PMLB**: Machine Learning

SPEC Benchmarks

- **SPEC**: Systems Performance Evaluation Cooperative is a non-profit corporation formed by leading computer vendors to develop a standardized set of benchmarks.
- www.spec.org
- **SPEC CPU 2017** designed to provide performance measurements that can be used to compare compute-intensive workloads on different computer systems, contains four benchmark suites:
  - **SPECspeed 2017:** **integer and FP**
  - **SPECrate 2017:** **integer and FP**

SPEC CPU 2017

Calculating SPECspeed Metrics	Calculating SPECrate Metrics
• 1 copy of each benchmark in a suite is run.	• The tester chooses how many concurrent copies to run
• The tester may choose how many OpenMP threads to use.	• OpenMP is disabled.
• For each benchmark, a performance ratio is calculated as: • Time on a reference machine / time on the SUT	• For each benchmark, a performance ratio is calculated as: • Number of copies × (time on a reference machine / time on the SUT)
• Higher scores mean that less time is needed.	• Higher scores mean that more work is done per unit of time.

SPEC CPU 2017

Calculating SPECspeed Metrics	Calculating SPECrate Metrics
Example • The reference machine ran 600.perlbench_s in 1775 seconds. • A particular SUT took 354.3 seconds to run one copy. • SPECspeed = 1775/354.3 = 5.01	**Example** • The reference machine ran 1 copy of 500.perlbench_r in 1592 seconds. • A particular SUT ran 8 copies in 541.5 seconds. • SPECrate = 8×(1592/541.5) = 23.5

TPC Benchmarks

- **Transaction Processing Performance Council**
- http://www.tpc.org/
- Developed many benchmarks:
  - **TPC-A** (Obsolete)
  - **TPC-B** (Obsolete)
  - **TPC-C**
  - **TPC-E**
  - **TPC-H**
  - **TPC-Energy**

TPC-C

- Simulates a complete computing environment where a population of users executes **transactions against a database**.
- TPC-C performance is measured in **new-order transactions per minute**.
- The primary metrics are the transaction rate (**tpmC**), and the associated price per transaction (**$/tpmC**).

TPC-E

- Uses a database to model a **brokerage firm** with customers who generate transactions related to trades, account inquiries, and market research.
- The brokerage firm in turn interacts with financial markets to execute orders on behalf of the customers and updates relevant account information.
- The TPC-E metric is given in **transactions per second** (**tps**).

TPC-H

- Is a **decision support** benchmark. It consists of a suite of business-oriented ad-hoc queries and concurrent data modifications.
- The performance metric reported by TPC-H is called the TPC-H Composite **Query-per-Hour Performance Metric** (**QphH@Size**) for a **specific database size**.
- The TPC-H Price/Performance metric is expressed as **$/QphH@Size**.

TPC-Energy

- Contains the rules and methodology for measuring and reporting an **energy metric** in TPC Benchmarks.

- This includes the energy consumption of system components associated with typical business information technology environments.

Outline

- Terminology
- Types of Workloads
    1. Single Operation
    2. Mixture of Operations
    3. Kernels
    4. Synthetic Programs
    5. Application Benchmarks
- Workload Selection
- Workload Characterization
- Monitors

Workload Selection

Considerations

1. Services exercised
2. Level of detail
3. Loading level
4. Representativeness
5. Timeliness
6. Repeatability

1. Services Exercised

- The **workload selected depends upon the system under test** (SUT).
- **Examples**
  - CPU: Instructions
  - System: Transactions
  - Two systems identical except for CPU
    - Comparing Systems: Use transactions
    - Comparing CPUs: Use instructions
- For SUT providing multiple services, exercise as **complete a set of services** as possible.

2. Level of Detail

- **Most frequent request**
  - Valid if one service is much more frequent than others
- **Frequency of request types**
  - Example: Instruction mixes
- **Time-stamped sequence of requests**
  - May be too detailed
  - Not convenient for analytical modeling
- **Average resource demand**
  - Used for analytical modeling
- **Distribution of resource demands**
  - Used if the distribution has large variance and impacts the performance

3. Loading Level

- A workload may exercise a system in:
  - **Best case**: Full capacity
  - **Worst case**: Beyond its capacity
  - **Typical case**: At the load level observed in real workload
- For **procurement purposes** $\Rightarrow$ Use typical case
- For **design purposes** $\Rightarrow$ Use all cases

Other Considerations

4. **Representativeness**: The test and real workloads should have same characteristics, *e.g.*, elapsed time and resource demands.
5. **Timeliness**: The test workload should be current.
   - Users are a moving target.
   - Users tend to optimize the demand, *e.g.*, given fast multiplication $\Rightarrow$ Higher frequency of multiplication instructions.
   - Important to monitor user behavior on an ongoing basis.
6. **Repeatability**: The test workload can be easily used in successive evaluation experiments with repeatable results.

Outline

- Terminology
- Types of Workloads
  1. Single Operation
  2. Mixture of Operations
  3. Kernels
  4. Synthetic Programs
  5. Application Benchmarks
- Workload Selection
- Workload Characterization
- Monitors

Workload Characterization

- **Workload features**: Measured quantities of service requests or resource demands, *e.g.*, transaction types, instructions, packet sizes, source-destinations of a packet, and page reference pattern.
- **Do not use parameters that depend upon the system**, *e.g.*, the elapsed time and CPU time.
- **Features of service requests**
  - Arrival time
  - Type of request or the resource demanded
  - Quantity of the resource demanded, *e.g.*, pages of memory

Outline

- Terminology
- Types of Workloads
  1. Single Operation
  2. Mixture of Operations
  3. Kernels
  4. Synthetic Programs
  5. Application Benchmarks
- Workload Selection
- Workload Characterization
- Monitors

Monitors

- A **monitor** is a tool used to observe the activities on a system.
- Useful for:
  - Characterizing workloads
  - Software optimization
  - Finding bottlenecks
- **Event** is a change in the system state.
- **Trace** is a log of events usually including event times.
- Monitors often have overheads.

Monitor Types

- **Trigger type**
  - Event driven: Good for infrequent events
  - Timer driven (sampling): Good for frequent events
- **Results displaying**
  - Online
  - Batch
- **Implementation**
  - Software, hardware, firmware, and hybrid

HW and SW Monitors

- **Hardware Monitors**
  - Timers
  - Counters
  - Probes
  - Logic analyzers
- **Software Monitors**: Activation mechanism:
  - Trap instruction
  - Trace mode; interrupt after every instruction
  - Timer interrupt (resolution problem)

Comparison

Criterion	Hardware monitor	Software monitor
Domain	HW events	Application and OS events
Input rate	High	Low
Time resolution	High	Low
Overhead	None	Varies

Summary

- Terminology
- Types of Workloads
  1. Single Operation
  2. Mixture of Operations
  3. Kernels
  4. Synthetic Programs
  5. Application Benchmarks
- Workload Selection
- Workload Characterization
- Monitors