Measurement Techniques and Tools

- Types of workload and workload selection
- Workload characterization technique
- Monitors and accounting logs
- Benchmarking
- Case study

-Abdul Waheed

Types of Workloads and Workload Selection

-Abdul Waheed

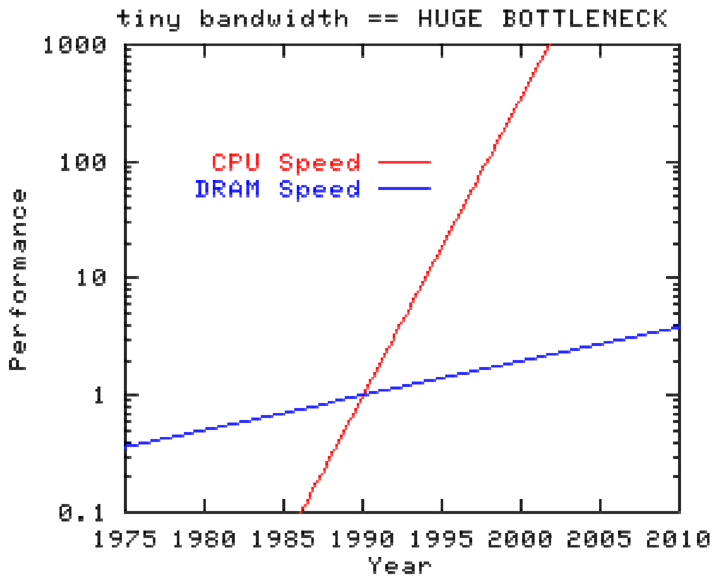
Workloads

- Many workloads are traditionally used to compare systems
  - Terminology and workloads mostly date back to time-sharing systems
- Test workload
  - Used for performance studies
    - Real or synthetic
  - Real workload
    - One that is observed on a system as it is being used in normal operation
    - It cannot be repeated; no suitable for testing
  - Synthetic workload
    - It is developed for performance studies with characteristics similar to real workload
    - It is repeatable
    - No sensitive data involved in testing

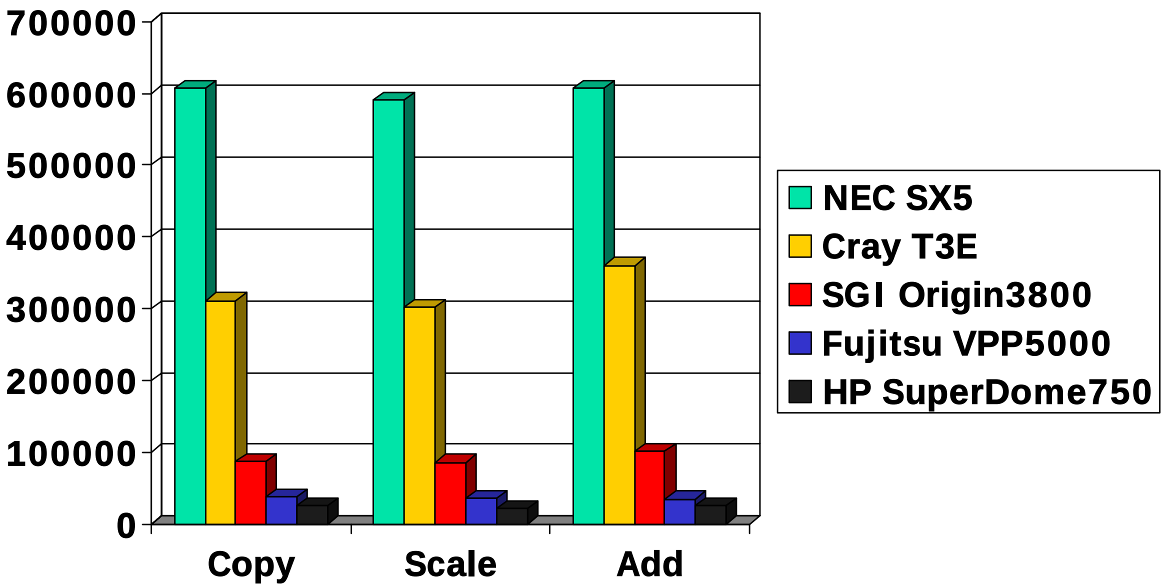
Workloads Examples

- Example: latency evaluation for web based transactions
  - Real workload
    - Transaction request from a browser
    - Latency includes
      - Network latency: communication time and delays due to queuing
      - Server latency: processing time, which is a function of load
    - Hard to repeat latency results as network congestion and server load may vary
  - Synthetic workload
    - Simulated transaction with same characteristics as real
    - Simulated network conditions using e.g., DummyNet tool
    - Results are easy to repeat as both server load and network latencies can be controlled
- Example: memory system performance evaluation
  - Real workload
    - Real application program
    - Order of memory accesses may not be the same over multiple experiments
  - Synthetic workload
    - Memory reference trace
    - Same order of memory references for all experiments

Motivation



- Performance bottlenecks
  - Was CPU
  - Now, it is memory
- Comparison of various systems



Types of Workload

- Single instruction (e.g., addition)
- Instruction mixes
- Kernels
- Synthetic programs
- Application benchmarks

Addition Instruction

- Historically, processor performance was most important part of computer system performance
  - Faster processor meant faster computer system
- Number of instructions that computers executed were few
  - Addition was most frequent one
  - Thus, computer with faster addition instruction was more powerful
  - Workload: addition instruction
  - Metric: addition time
- Why performance evaluation based on addition instruction is insufficient for today's computers?
  - CPU is no longer a bottleneck resource in computer systems
  - Pipelining and superscalar techniques enhance CPU performance
  - Bottleneck resource is memory subsystem

Instruction Mixes

- Number of instructions grew with processor complexity
  - Addition instruction was not enough
  - Instead, use relative frequencies of several instructions on real systems as weighting factors to get an average instruction time
- An instruction mix is a specification of various instructions coupled with their usage frequency
  - Compute single number metric: CPI or average instruction time
  - Inverse of average instruction time is also used: MIPS or MFLOPS
  - Use these measure to compare various CPUs
  - Several instruction mixes are used in computer industry
- Gibson mix
  - Developed by Jack Gibson in 1959 for use with IBM 704 systems
  - Processor speed was determined by measuring memory cycle time, addition time, or an average of addition and multiplication times
  - Gibson mix extended averaging to 13 different classes of instructions
  - Weights are based on frequency of operations on IBM 704 and IBM 650 systems

Gibson Instruction Mix

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	1.6
12.	Instructions not using registers	5.3
13.	Indexing	18.0
		100.0

Drawbacks of Using Instruction Mixes

- Computer architecture has become too complex
  - Classes of instructions that are not reflected in a mix
    - Supersclar, pipelining, cache pre-fetching, address translation (TLB), speculative execution, etc.
  - Execution time is highly variable due to addressing modes, cache hit rates, pipeline efficiency, and interference from other applications
- An instruction mix is not a real program
  - A mix will have a fixed contribution from each type of operation
  - Real program/data may use sparse data structures
    - Most of multiplies with 0 will be much faster
    - Some conditional branches are based on data
- Measure only the speed of a processor as a single number
  - May not reflect the system performance, which is limited by bottleneck resource
  - Useful when processor is the bottleneck resource

Kernels

- Limitations of instruction mixes due to architectural complexity lead to the use of kernels
  - A kernel is a generalization of instruction mix
  - Kernel literally means nucleus
- A set of instructions that constitutes a higher level function
  - Most frequent of such functions can be used as workload
  - Aka a **kernel**
  - A **processing kernels** solely exercises processor without using I/O devices
- It is possible to identify a kernel in several applications
  - A set of common operations e.g., sorting, matrix inversion, differential equation solution, etc.
  - Different processors can be compared using kernels
  - Kernels are not selected based on any measurements
    - They simply become popular due to use by many researchers

Examples of Kernels

- Scalar product of two vectors (BLAS1)
- Matrix vector multiply (BLAS2)
- Matrix inversion
- Sorting
- CFD kernels in NAS benchmarks
    - Numerical solution of partial differential equations
    - Examples
        - FT
        - LU

Disadvantage: kernels typically do not exercise I/O subsystem

Synthetic Programs

- Synthetic programs were developed to overcome limitations of kernels
  - Exerciser loops for I/O devices
    - Specified number of I/O requests to determine device performance
  - Often written in high-level languages for portability
- First exerciser loop written by Buchholz (in 1969) was called a synthetic program
  - Other exercise loops are used to measure OS services
  - Examples: process creating, memory allocation, etc.
- Example: a synthetic program to evaluate I/O performance
  - Make a number of disk read and write requests
  - Determine latency of read or write
  - Control parameters
    - Number of iterations
    - Size of blocks moved to and from disk
    - Buffered vs. unbuffered
  - Metrics
    - Throughput: ops/sec
    - Latency: seek time, read latency, and write latency

Example: Synthetic Workload Generator

```
      DIMENSION Record(500)
!Control parameters:
      Num_Computes=500            !Repeat count for computation
      Num_Reads=35                !Number of records read
      Num_Writes=40               !Number of records written
      Num_Iterations=1000         !Repeat count for experiment
!Open files:
      OPEN(UNIT=1,NAME='In.dat',TYPE='Old',
     1FORM='Unformatted',ACCESS='Direct')
      OPEN(UNIT=2,NAME='Out.dat',TYPE='New',
     1FORM='unformatted',ACCESS='Direct')
      CALL Get_Time(CPU1,Elapsed1)     !Record starting time

      DO 500 Iteration=1,Num_Iterations

!Perform a number of read I/Os
      DO 100 i=1,Num_Reads
      READ(1'i),Record
100   CONTINUE
!Do computation
      DO 200 j=1,Num_Computes
      DO 200 i=1,500
200   Record(i)=1 + i + i*i + i*i*i
!Perform a number of write I/Os
      DO 300 i=1,Num_Writes
      WRITE(2'i),Record
300   CONTINUE
500   CONTINUE
      CALL Get_Time(CPU2,Elapsed2)     !Get ending time
!Close files:
      CLOSE(UNIT=1)
      CLOSE(UNIT=2)
      CPU_Time=(CPU2-CPU1)/Num_Iterations
      Elapsed_Time=(Elapsed2-Elapsed1)/Num_Iterations
      TYPE *,'CPU time per iteration is ',CPU_Time
      TYPE *,'Elasped time per iteration is ',Elapsed_Time
      STOP
      END
```

Pros and Cons of Synthetic Programs

- Advantages
  - Overcomes limitation of kernels
  - Exercises I/O and OS services
  - Can be developed quickly and given to vendors
  - Not necessary to use real data files that may contain sensitive information
  - Programs can easily be modified and ported
  - Once developed, measurement process is automated on
- Disadvantages
  - Too small
  - Do not make representative memory or disk references
  - Page faults and disk cache may not be properly exercised
  - CPU and I/O operations overlap may not be representative
  - May not represent multi-user environments

Application Benchmarks

- Benchmark programs are suitable for a particular application
  - Scientific applications
    - Linear algebra
    - Computational Fluid Dynamics (CFD)
  - Banking and air line reservations
    - Databases
    - Transaction processing systems
  - WWW
- Exercise almost all sorts of system resources
  - CPU
  - Operating system
  - Cache and memory subsystem
  - I/O devices
  - Network
- Benchmarking = measurement based comparison among systems
  - Workload in such studies is referred to as benchmarks

Popular Benchmarks

- Sieve
  - An algorithm to find all prime numbers below a given number n
    - Consider all number from 1, ..., n
    - Strike out all multiples of k = 2, 3, ..., $n^{1/2}$
    - Remaining number are prime numbers
  - A high-level language program can be written and run on multiple systems to compare their performance
  - Performance depends on
    - Cache/memory overhead and data structure used to lay out the dataset
- Ackermann's function
  - It is a recursive function
  - Assesses the efficiency of procedure-calling mechanisms in high-level languages
  - Determines execution time per call, # of instructions executed per call, and amount of stack space required per call

Popular Benchmarks (Cont'd)

- Whetstone
  - Used at British Central Computer Agency
  - The kernel consists of 11 modules to match the frequency of operations of 949 ALGOL programs
  - Represents small engineering/scientific applications
    - FP workload
  - Exercises processor
    - Array addressing, fixed and FP arithmetic, subroutine calls
  - Results are measures as KWIPS (Kilo Whetstone Inst/Sec)
- Dhrystone
  - Developed in 1984 at Siemens
  - Lots of procedure calls
  - Kernel represents system programming environments
    - Integer workload
    - Does not exercise FP unit or I/O devices
  - Results presented as DRIPS (Dhrystone Inst/Sec)

Popular Benchmarks (Cont'd)

- Linpack
  - Developed by Jack Dongarra at Argonne National Lab in 1983
  - Programs to solve dense systems of linear equations
    - High percentage of FP add and multiply instructions
    - Most of the time is consumed in as set of subroutines called Basic Linear Algebra Subprograms (BLAS)
  - Compares systems based on execution rates (MFLOPS)
  - Popular for comparing FP performance
- Livermore loops
  - Workload consists of a set of 24 separate tests
  - Tests dominated by large vectorizable scientific computations
  - Abstracted from real scientific (supercomputing) applications
  - Applications represent FP workloads
  - Measure performance in MFLOPS

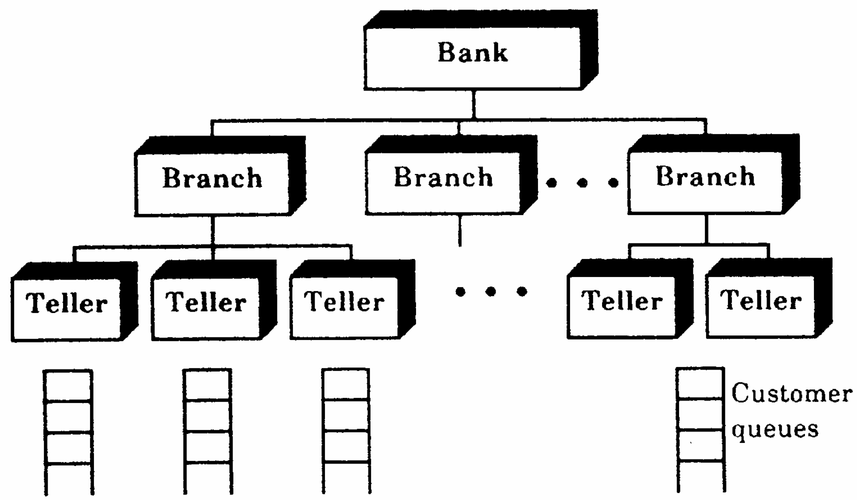
Popular Benchmarks (Cont'd)

- NAS parallel benchmarks
  - Benchmark is based on CFD kernels that solve PDEs
  - It is the first paper-and-pencil benchmark
    - It simply specifies the problem
    - Anyone is allowed to use specifications to write their own code
    - Lots of room to optimize code for specific architectural features
    - Flexibility to use advanced compiler techniques for vectorization or parallelization
  - It is often used to compare the performance of parallel and vector supercomputing systems
    - Scientific FP code
    - Almost all kernels are memory-intesive
  - Performance is measured in terms of MFLOPs as well as execution time on one or more processors
    - Several implementations are also available now

Popular Benchmarks (Cont'd)

- Debit-Credit Benchmark
  - Application benchmark for comparing transaction processing systems
  - Benchmark represents a distributed banking network
    - Consists of several branch offices, each with tellers
    - Customers wait in queues for next teller
  - Select suitable parameters for a study: number of branches, tellers, and account holders
  - Systems compared with price-performance ratios
  - TPC (Transaction Processing Performance Council) benchmark is a variant of debit-credit benchmark

```
Begin-Transaction
    Read message from the terminal   (100 bytes)
    Rewrite account                  (100 bytes, random)
    Write history                    (50 bytes, sequential)
    Rewrite teller                   (100 bytes, random)
    Rewrite branch                   (100 bytes, random)
    Write message to the terminal    (200 bytes)
Commit-Transaction
```



Popular Benchmarks (Cont'd)

- SPEC benchmark suite
  - Standardized set of benchmarks developed by System Performance Evaluation Cooperative (SPEC)
  - SPEC benchmark suites: 1992, 1995, 1998, etc.
  - Based on programs contributed by scientists and engineers
    - GCC Gnu C compiler
    - Espresso Electronic design automation
    - Spice Electronic design automation
    - Doduc Monte Carlo simulation on nuclear reactor
    - NASA7 FP kernels using matrix ops submitted by NASA
    - LI Time to solve 8-queens problem by LISP interpreter
    - Eqntott Translates boolean equation to a truth table
    - Matrix300 Linpack operations on 300x300 matrices
    - Fpppp Quantum chemistry benchmark
    - TomcatvVectorized mesh generation program
  - These benchmarks exercise CPU, FPU, and memory subsystems
  - Metric: SPECmark is determined as geometric mean of relative throughput (SPECthruput) as measured wrt a VAX-11/780

SPEC2000 CPU Performance Comparisons