

Real-Time Benchmarking : Methodology and Applications

Introduction

- **What is a benchmark ?**
- **Need for benchmarks**
- **Benchmarking methodology - low-level, high-level and domain specific**

System benchmarking

- **Claims - 1, 2, 3**
- **Low-level benchmarks - Rhealstone + Superconducting Supercollider**
- **High-level Application benchmarks - Hartstone benchmarks**
- **Real-time measures**

Domain(Problem) Specific Benchmarks

- **Honeywell benchmarks for complex real-time apps - low-level, kernel-level and domain-specific(application)**
- **Mitre benchmarks for Real-time embedded DSP applications - low-level, kernel-level and domain-specific(application)**

- **DARPA RASSP SAR DSP benchmarks**

SYSTEM = HARDWARE + SYSTEM SOFTWARE + APPLICATION + NETWORK

What is a benchmark ?

- **A benchmark is a program(codelet or “juggernaut”) that exercises the system based on different input parameters and required outputs.**
- **Models the physical system, in the absence of a real physical system for study and experimentation.**

Need for Benchmarks

- **Validate and verify the functioning of the system**
- **Compare multiple systems - for performance and timing**
- **In the absence of a real physical system, provide a model of a physical system with workload to physical system mappings. This allows workloads to be run on a “virtual” system.**

Benchmarking Methodology

- **Use a benchmark hierarchy - Low-level, Kernel-level, Compact applications**
- **Low-level - Clock, Interrupt Latency**

- **Kernel-level (Computation and Communication building blocks) - 1D FFT, Distributed Corner turn, QR decomposition**
- **Applications - RT_STAP, RT_ARM, RASSP SAR**

System Benchmarking

- **Claim 1: “The RTOS is the brain of the Real-time System”**
- **Claim 2: “If the RTOS is the brain of the Real-time system then, benchmarks are its vital indicators”**
- **Claim 3: “Benchmarks are the basis of modern Real-time Systems research”**

Low-level Benchmarks

Parkbench

- **Clock accuracy, Clock resolution**

Rhealstone

- **Task switch time(context switch) - Average time to switch between two tasks of equal priority**
- **Preemption time - Average time for a high priority task to preempt a lower priority task**
- **Interrupt Latency - Average delay between the CPU's receipt of an interrupt request and the execution of the first instruction**
- **Semaphore-shuffle time - The delay in the OS before the task acquires a semaphore that is in the possession of another task**
- **Deadlock break time - This is the average time to break a deadlock caused when a high priority task preempts a low-priority task that is holding a resource the high-priority task needs**
- **Intertask message latency - This is the delay in the OS when a non-zero length data message is sent from one task to another**
- **Datagram throughput time - Number of Bytes/sec sent between two tasks using system communication primitives**
- **(App specific) Rhealstone number = (Weighted by relative frequency) mean of the above**

SSC Benchmark

- **Create/Delete Task - Time to create and delete a task**
- **Ping Suspend/Resume - A low priority task resumes a suspended high priority task**
- **Suspend/Resume task - A high priority task suspends and resumes a suspended lower priority task**
- **Ping Semaphore - Two tasks of same priority ping a semaphore**
- **Getting/Release Semaphore - Take and give a semaphore**
- **Queue fill, Drain, Fill Urgent**
- **Allocating/Deallocating memory**

Real-time tasks

- **Periodic (harmonic and nonharmonic)**
- **Interactive loads (non-realtime loads) - Aperiodic + Periodic**
- **Schedulability Test (Liu and Layland) - “n” independent periodic tasks will always be schedulable(not miss their deadlines) if system utilization is less than 69.3 %**

HARTSTONE Benchmark Experiments

- **PH Series - Periodic Tasks, Harmonic frequencies**
- **PN Series - Periodic Tasks, Nonharmonic frequencies**
- **AH Series - Aperiodic tasks**
- **SH Series - Periodic tasks with Synchronization**
- **SA Series - Periodic tasks with aperiodic processing and synchronization**

Hartstone Distributed Benchmark

- **Extends to distributed systems**
- **Same workload model as the HUB**
- **Deadlines for tasks and messages**
- **Deadline of the receiving task determines the deadline of the receiving message**
- **Breakpoint - Increase length of message, number of messages until a message ACK is missed before a deadline of a sending task**

- **Provides figures of merit for the complex end-to-end scheduling and timing behavior of the system**

Real-time Measures

- **Rhealstone**
- **Process Dispatch Latency**
- **Tri-dimensional Measure**
- **Real/Stone - Obsolete**

HONEYWELL Benchmarks for Complex Real-time Applications(C3I) (Adaptive Real-time)

- **Level 1 - Basic hardware and software (message passing performance of an interconnection switch) - local clock, global clock, scheduling clock and real-time communications**
- **Level 2 - Common building of military applications - signal transforms**
- **Level 3 - Evaluate application sub-systems**
Pipelined data parallel stages, input from sensor banks, Central controller to monitor and control real-time performance(frame rates, latency, resource reservations, resource utilization, allocation, adaptation, enactment)

MITRE Benchmarks for Embedded High Performance Computing (Adaptive Real-time)

- **Low-level benchmarks - Bintime, Time-driven regularity**
- **Kernel-level benchmarks - 1D FFT, Distributed Corner Turn, QR Decomposition**
- **Application benchmarks - RT_2DFFT, RT_STAP**

DARPA RASSP SAR Benchmark

- **Radar front-end benchmark (strip map imaging)**
- **Pipelined data parallel stages**
- **Focus on frame processing and frame latency**
- **Interesting - Effects of back-flow events**
- **Steps are :**

Load Kernels

FIR Filter

Apply Taylor weights + FFT (Range compression)

Corner turn + Apply Azimuth Kernel + FFT (Azimuth compression)