PERFORMANCE PROFILING

Minsoo Ryu

Real-Time Computing and Communications Lab.
Hanyang University

msryu@hanyang.ac.kr

OUTLINE

- ❒ HISTORY
- ❒ UNDERSTANDING PROFILING
- ❒ UNDERSTANDING PERFORMANCE
- ❒ UNDERSTANDING PERFORMANCE PROFILING
- ❒ PERFORMANCE PROFILING TYPES
  - ▪ PERFORMANCE COUNTERS
  - ▪ EVENT BASED PROFILING
  - ▪ STATISTICAL PROFILING
  - ▪ INSTRUMENTATION PROFILING
  - ▪ HYPERVISOR/SIMULATOR
- ❒ PERFORMANCE PROFILING TOOLS

HISTORY

HISTORY

❒ 1970

- Usually based on timer interrupts which recorded the program status word (PSW) at set timer-intervals to detect "hot spots" in executing code
  - Instruction-set simulators permitted full trace and other performance-monitoring features in 1974

❒ 1980

- Profiler-driven program analysis on Unix dates back to at least 1979, when unix systems included a basic tool
  - Prof which listed each function and how much of program execution time it used
  - Gprof extended the concept to a complete call graph analysis in 1982

❒ 1990

- The ATOM(analysis tools with OM) platform converts a program into its own profiler
  - It inserts code into the program to be analyzed: instrumentation profiling
    - ✓ That inserted code outputs analysis data, at compile time

UNDERSTANDING PROFILING

UNDERSTANDING PROFILING

❐ What is measured

- Time
- Control flow
  - Loop counts, Function calls
- Aliasing facts
- Cache stats
- Allocation information
  - Track allocation sites for objects
- Hardware stats
- Granularity of what is measured
  - Instructions, basic blocks, line of code, function, modules

UNDERSTANDING PROFILING

❒ How are measurements taken

- Instrumentation
  - – The code is modified to take the measurements
  - – When?
    - ✓ At compile time: static time
    - ✓ At runtime: dynamic time
- Interruption
  - – An outside event triggers inspection and measurement
  - – Who?
    - ✓ Hardware
    - ✓ Timer
    - ✓ Another thread

UNDERSTANDING PROFILING

❒ When are measurements taken

- All the time
  - – Expensive
- Sampling
  - – Cheaper
  - – When?
    - ✓ N_th function call
    - ✓ N_th basic block
    - ✓ Timer
    - ✓ Some property of the hardware

UNDERSTANDING PERFORMANCE

UNDERSTANDING PERFORMANCE

- We need visibility into what the machine is doing
  - To understand low-level performance
  - We mostly only control data and code
- We need to know how it interacts with HW
  - But to make an existing algorithm/data-structure work better
- Sometimes it is enough to know time
  - – Algorithm-level fix, data-structure-level fix
- Part of performance measurements
  - – CPU
  - – Memory
  - – Network
  - – Disk
  - – SQL DB
  - – User Input

UNDERSTANDING PERFORMANCE

❐ Aspects of performance

- **Availability**
  - Availability of a system is typically measured as a factor of its reliability
    - ✓ As reliability increases, so does availability
- **Response time**
  - Response time is the total amount of time it takes to respond to a request for service
  - The response time is the sum of three numbers
    - ✓ Service time + wait time + transmission time
- **Throughput**
  - Throughput is the rate of production or the rate at which something can be processed

UNDERSTANDING PERFORMANCE

❒ Aspects of performance

- **Latency**
  - Latency is a time delay between the cause and the effect of some physical change in the system being observed.
  - Latency is a result of the limited velocity with which any physical interaction can take place.
    - ✓ This velocity is always lower or equal to speed of light
- **Scalability**
  - Scalability is the ability of a system or process to handle a growing amount of work in a capable manner or its ability to be enlarged to accommodate that growth

UNDERSTANDING PERFORMANCE PROFILING

UNDERSTANDING PERFORMANCE PROFILING

❒ What is a hotspot

- Where in an application or system there is a significant amount of activity
  - Significant: activity that occurs infrequently probably does not have much impact on system performance
  - Activity: time spent or other internal processor event

- Examples of other events: cache misses, branch mispredictions, floating-point instructions retired, partial register stalls, and so on

UNDERSTANDING PERFORMANCE PROFILING

❒ In theory, the " algorithm " and " data structure " can be predicted through knowledge. but, performance profiling requires precise measurements for the possible prediction results

- Even though " the same problem " , substantially significant differences occur
- The hotspot can be found through precise measurements

❒ Performance profiling programming

- Knowledge of all the layers involved
- Experience in knowing when and how performance can be a problem
- Skill in detecting and zooming in on the problems
- A good dose of common sense

PERFORMANCE PROFILING TYPES

PERFORMANCE PROFILING TYPES

- ❒ PERFORMANCE COUNTERS
- ❒ EVENT BASED PROFILING
- ❒ STATISTICAL PROFILING
- ❒ INSTRUMENTATION PROFILING
- ❒ HYPERVISOR/SIMULATOR

PREFORMANCE COUNTERS

- ❒ Performance counters
  - We need to count the events
  - Program a counter to count a specific event
  - HW provides special programmable counters
  - Software can read and write the counters

- ❒ What's wrong with performance counters
  - Should be solved the course-granularity problem
  - Extend performance counters with a programmable limit value
    - Cause an interrupt when the counter overflows

EVENT BASED PROFILING

- ❒ Event based profiling is triggered by any one of the software-based events or any performance monitor event that occurs on the processor
- ❒ **Event-based profiling: advantages**
  - The routine addresses are visible when interrupts are disabled
  - The ability to vary the profiling event
  - The ability to vary the sampling frequency
- ❒ **Event-based profiling: disadvantages**
  - Limited data collection
    - Per run
    - Types of data
  - In-exact
    - Sample point and event cause don't match

STATISTICAL PROFILING

- ❒ A method of profiling an application by taking samples of the program address at regular intervals while the application is executing

- ❒ Cycle accurate profiling methods
  - ▪ These samples are then associated with either a specific function or a specific memory range
  - ▪ A simple statistical analysis is performed to determine the areas where an applications spends the largest portions of its cycles
  - ▪ Statistical Profiling is a very quick and handy way to get a first look at which functions are consuming the largest proportions of cycles

STATISTICAL PROFILING

❒ **The statistical profiling: advantages**

- No installation required
- Wide coverage
- Low overhead

❒ **The statistical profiling: disadvantages**

- Approximate precision
- Limited report

INSTRUMENTATION PROFLING

- ❒ A method of profiling how to get the information by inserting an additional command to the code
  - ▪ The performance impact depends on the amount of detail in the data type
    - To measure the execution frequency of each line VS execution times of the function
  - ▪ Method
    - Programmer to insert code by hand
    - Automatic code insertion tool
    - How to insert the code with the help of the compiler
    - The final non-binary conversion code
    - Modify the binaries to run immediately before
    - Operate in protected mode while running

INSTRUMENTATION PROFLING

❑ The instrumentation profiling: advantages

- Perfect accuracy □
  - Where you visit immediately before and after your
  - Can calculate how much time you spent at each site
  - How many times you visited each site

❑ The instrumentation profiling: disadvantages

- Low granularity □
  - Too coarse
- High overhead □
- High touch □
  - I have to build all those area, which expands the space in each site you visit

HYPERVISOR/SIMULATOR

❐ Hypervisor/Simulator

- Hypervisor: data are collected by running the unmodified program under a hypervisor
- Simulator: data collected interactively and selectively by running the unmodified program under an instruction set simulator

PERFORMANCE PROFILING TOOLS

PERFORMANCE PROFILING TOOLS

In order to do	You can use
Performance counters	**Perf, Oprofile**
Event based profiling	**Ruby, JIT**
Statistical profiling	**Oprofile, Vtune**
Instrumentation profiling	**Gprof, Valgrind**
Hypervisor/Simulator	**SIMMON, OLIVER**
Other tools exist for Network, Disk IO…	

수고하셨습니다.