**GOYA AMEBE-OBARI AWIANKELE**

**16/SCI03/007**

**CSC 410**

The three main performance evaluation techniques are:

Simulation performance modeling

Performance measurement

Analytic performance modeling

The possibilities of achieving an accurate report include:

* Off-chip hardware measurement: Instrumentation using hardware means can also be done by attaching off-chip hardware.
* **On-chip performance monitoring counters**: all state-of-the-art high performance microprocessors including IBM’s POWER 3 and POWER 4 processors, AMD’s Athlon, Compaq’s Alpha, Sun’s UltraSPARC processors incorporate on-chip performance monitoring counters which can be used to understand the performance of these microprocessors while they run complex, real-world workloads. This ability has overcome a serious limitation of simulators, that they often could not execute complex workloads. Now, complex run time systems involving multiple software applications can be evaluated and monitored closely. For an illustration of on-chip performance monitoring, we use the Intel Pentium processors; the microprocessors in the Intel Pentium contain two performance monitoring counters.
* **Logic Analyzers**: a Tektronix TLA 700 logic analyzer is used to analyze 3D graphics workloads on AMD-K6-2 based systems. Detailed logic analyzer traces are limited by restrictions on sizes and are typically used for the most important sections of the program under analysis. Preliminary coarse level analysis can be done by performance monitoring counters and software instrumentation.
* **Software Monitoring**; this is often performed by utilizing architectural features such as a trap instruction or a breakpoint instruction on an actual system, or on a prototype. The VAX processor from Digital had a T-bit that caused an exception after every instruction. One advantage of this is that it is easy to do and one disadvantage is that the instrumentation can slow down the application.

Using the following simulation performance modeling techniques, we can archive an accurate performance report like;

* **Program Profilers**: there are some classes of tools called software profiling tools; they are similar to simulators and performance measurement tools. These tools are used to generate traces, to obtain instruction mix, and a variety of instruction statistics. They can be thought of as software monitoring on a simulator. They input an executable and decode and analyze each instruction in the executable.
* **Execution Driven simulation**; some practitioners refer to simulators that take program executable as input execution driven simulators. These simulators utilize the actual input executable and not a trace. Hence the size of the input is proportional to the static instruction count and not the dynamic instruction count. Miss-predicted branches can be accurately simulated as well. Thus, these simulators solve the two major problems faced by trace-driven simulators.
* **Stochastic Discrete Event Driven Simulation**: It is possible to simulate systems in such a way that the input is derived stochastically rather than as a trace/executable from an actual execution. For instance, one can construct a memory system simulator in which the inputs are assumed to arrive according to a Gaussian distribution. Such models can be written in general purpose languages such as C or using special simulation languages such as SIMSCRIPT. Languages such as SIMSCRIPT have several built-in primitives to allow quick simulation of most kinds of common systems. There are built-in input profiles, resource templates, process templates, queue structures etc. to facilitate easy simulation of common systems. An example of the use of event-driven simulators using SIMSCRIPT may be seen in the performance evaluation of multiple-bus multiprocessor systems in Kurian et. Al