Computer Architecture

Instrumentation and Simulation

Debadatta Mishra, CSE, IITK

Recap (Performance Analysis)

- Performance analysis
  - Throughput, Response time
  - Response time vs. throughput
  - Processor performance (IPC, CPI)

Agenda: Instrumentation (Intel PIN), Simulation (champsim, gem5)

Performance Analysis: why simulation?

- Benchmarks on real system is not very common for architecture performance analysis. Why?

Performance Analysis: why simulation?

- Benchmarks on real system is not very common for architecture performance analysis. Why?
- Issues with real systems
  - Opaqueness: no clue on what is underneath, can not change design
  - Noise: difficult to control the environment
  - Reproducibility: Real systems have insanely large # of configurations!

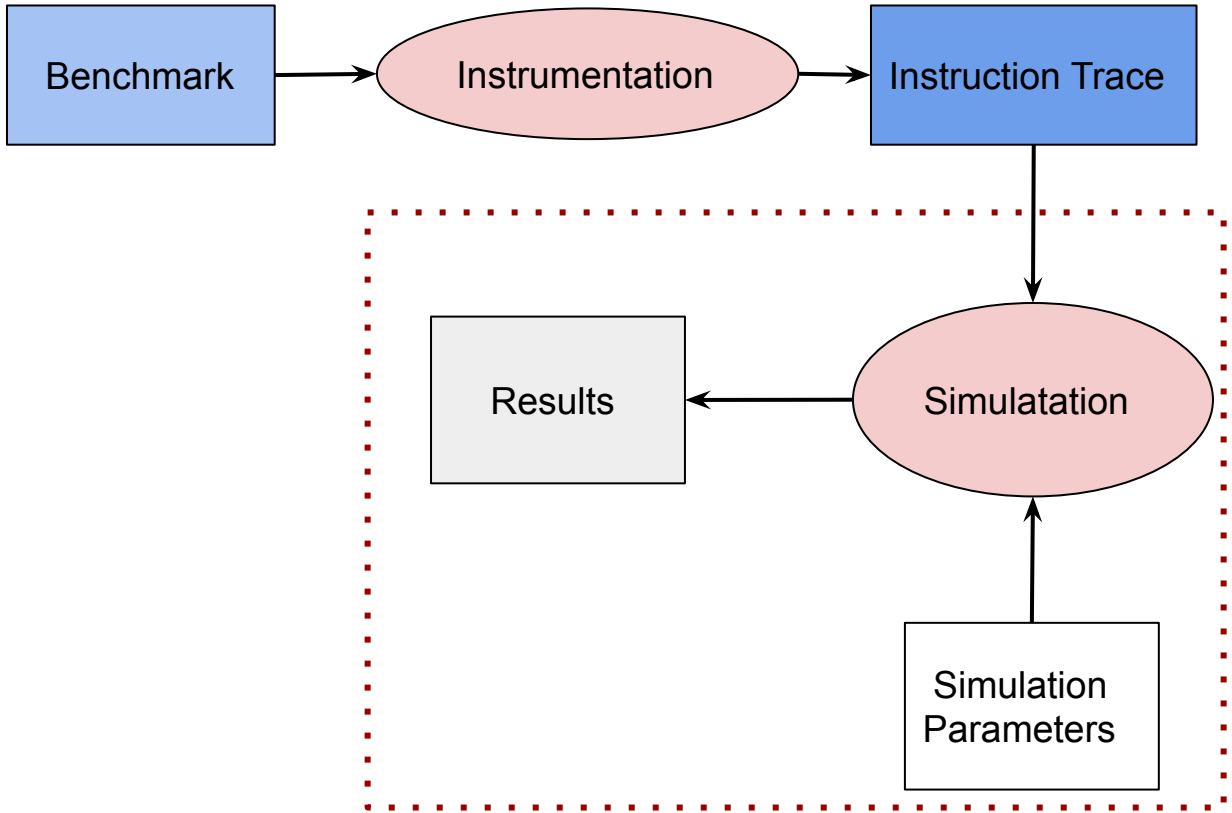
Performance Analysis: why simulation?

- Benchmarks on real system is not very common for architecture performance analysis. Why?
- Issues with real systems
  - Opaqueness: no clue on what is underneath, can not change design
  - Noise: difficult to control the environment
  - Reproducibility: Real systems have insanely large # of configurations!
- Architectural simulators
  - Discrete event simulators, examples: Champsim, Gem5, Multi2sim
  - More detailed simulation ⇒ Accurate modeling of the real system

Architectural Simulation: Taxonomy

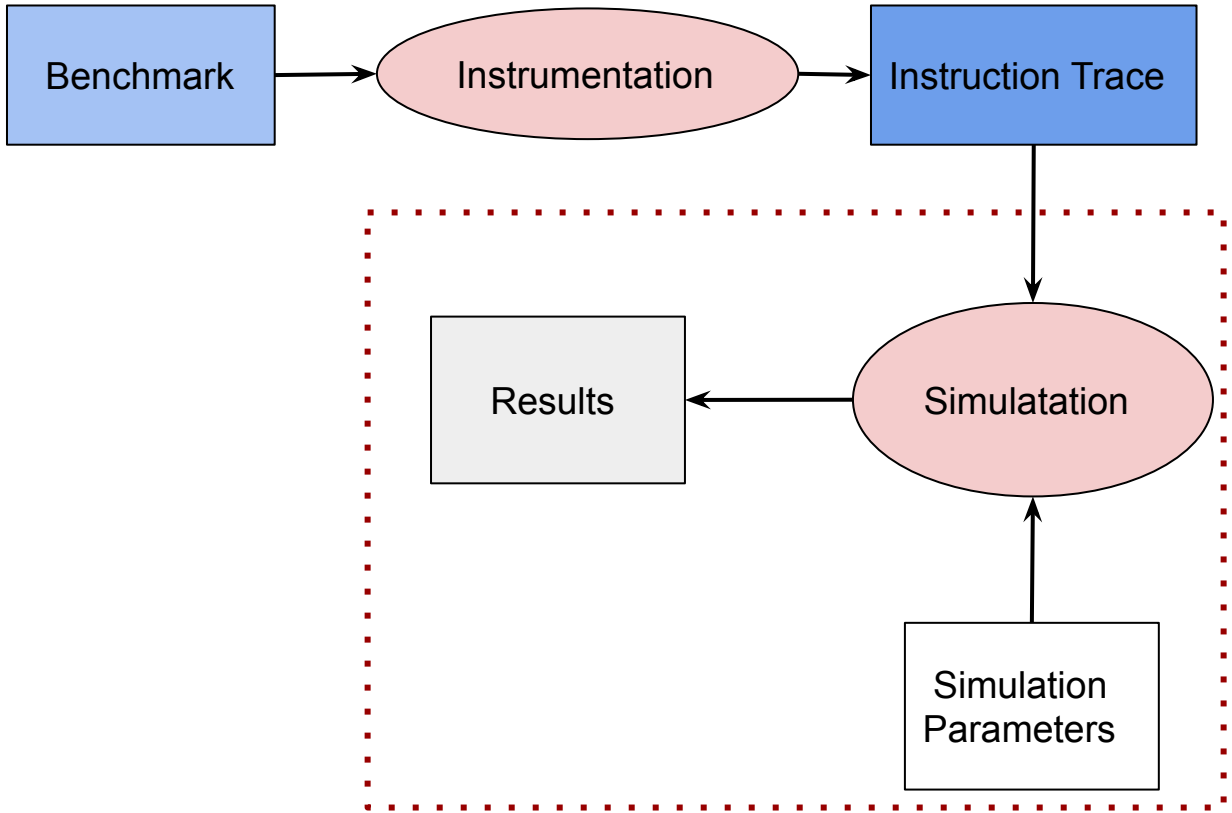
- Functional vs. Timing simulation
  - Functional simulation models instruction semantics and update states
  - Timing simulation additionally models timing at different elements in the micro-architecture
- Trace driven vs. Execution
  - Trace-driven simulators model instructions traced before (can be functional or timing)
  - Execution-driven simulators “execute” the program, simulating each instruction
- Execution-driven
  - Can be a process or the full system (including OS activities)

Trace based simulation



- Instrumentation is a one-time activity for a given benchmark
- Simulation is repeated for different configs and h/w designs

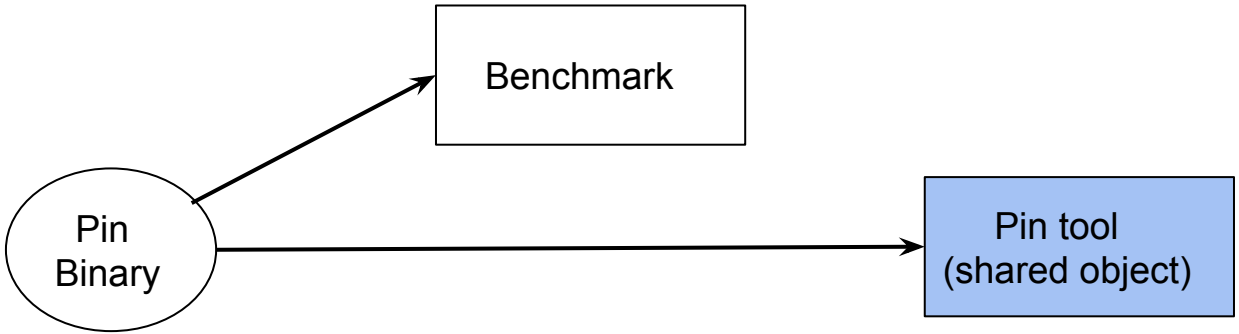
Trace based simulation



For this course:

- Instrumentation using Intel Pin
- Simulation using Champsim

Instrumentation using Intel Pin



Perform dynamic instrumentation of the benchmark binary like a JIT compiler.

Specify the points of instrumentation and handlers for those points.

Instrumentation using Pin

- Support for instrumentation at different granularity
  - Example: instruction, basic block, function (routine)

Instrumentation using Pin

- Support for instrumentation at different granularity
  - Example: instruction, basic block, function (routine)
- Handler invocation
  - Before (`IPOINT_BEFORE`), After (`IPOINT_AFTER`), and branch taken (`IPOINT_TAKEN_BRANCH`)

Instrumentation using Pin

- Support for instrumentation at different granularity
  - Example: instruction, basic block, function (routine)
- Handler invocation
  - Before (`IPOINT_BEFORE`), After (`IPOINT_AFTER`), and branch taken (`IPOINT_TAKEN_BRANCH`)
- Usage examples
  - Count all instructions in a binary
  - Count instructions executed in a function
  - Trace all memory load/store instructions

Simulation using Champsim

- Champsim is a trace-based simulator
- Features: x86 traces, OoO CPU, cache/memory
- Gives a specialized pintool for tracing
- Supports warmup and steady state
- Example:
  - Trace a simple benchmark compiled with different compiler optimization levels
  - Observe the IPC – better for lower optimization level (hmm?)
  - When comparing IPC, same trace to be used!

Execution Simulation using Gem5

- Support for simulating execution of a binary using system call emulation mode (SE mode)
- Supports full system simulation including OS activities (FS mode)
- Detailed parameters for CPU, Cache, Memory
  - More detailed simulation ⟶ increased simulation time
- CPU simulation modes
  - AtomicSimpleCPU
  - TimingSimpleCPU
  - O3CPU
- Supports many architectures (X86, ARM, risc-v)