Principles, Tricks and Quirks

VLDB ADMS Workshop Panel on In-Memory Database Processing

Ken Ross, Columbia University
Thesis

• Need to understand (and occasionally discover) principles that govern performance

• Come up with tricks that exploit those principles

• Understand quirks that affect performance (often in an architecture dependent way)
Example: Memory Hierarchy

• Principle: Locality improves cache performance

• Trick: CSB+ tree nodes have one pointer

• Quirk: Systems support multiple outstanding requests (larger nodes with prefetching)
Example: Branches

• Principle: hard-to-predict branches are expensive

• Trick: reorganize loop to avoid branches

  for i = 1 to n {
    answer[current] = i;  current += p(r[i]);
  }

• Quirk: compilers can help or get in the way
Example: Multicore

- Principle: cooperative work by many threads can utilize shared memory & cache efficiently
- Trick: selectively replicating data items can avoid contention
- Quirk: Atomic instructions and locks have different performance profiles on different machines
Many more examples

- Flash
- PCRAM
- GPUs
- FPGAs
- etc.
Principles

• Very general (usually not architecture dependent)

• Too high level to be directly actionable
Tricks

• Require innovation

• "Trick" should not be a pejorative term

• Often can be architecture independent, but magnitude of impact can vary
Quirks

- Often architecture specific
- Risk of solution being too narrow (no generalizability)
- Common, interesting, annoying
How to Keep Innovating

- Play around with the latest hardware
- Simulate future hardware
- Influence future hardware design