Providing Actionable Insight

Team:

Problem

- Fixing often requires a lot of knowledge
  - Microarchitectural issues you have to know about it and how to fix
  - Knowledge about what the UMD, KMD drivers are accomplishing with accelerators
- ROI is not obvious from fixing the problem
  - What if scenarios don't end up presenting the entire gains achievable
- Detailed modeling to catch problems accurately
- There are often other bottlenecks that prevent you from getting the full performance gains
  - Larger bottlenecks and you might get no gains
- Tools built for observation
  - Don't often describe the problem and how to fix
- Many tools are good at different things…but we cannot get holistic view of what is wrong
  - Have to use several tools
- Users don't know the ramification their coding decisions
  - Example is precision. Make sure in the code that you know you need that precision

Potential Solutions

- Can we use data analytics and inference to possible solutions?
- Standardized methodologies of outputting and utilizing data between tools
  - Tools can take that an automatically optimize for it
  - Example: AutoFDO, BOLT, HWPGO from Google, Facebook, Intel
- Can tools automatically optimize based on context
  - Utilize O1, O2, O3 based on the loops, code space maybe profiles
- Showing coders the ramifications of their actions is useful
  - Precision of floating point was one example given
- Tools still need to marry observations, documentation on issues and potential fixes
- Live dangerously options
  - Very dangerous optimizations but then give documentation on what optimizations were present
  - Michael Lam's work with floating point to automatically use lower precision FP
- Using the right compute under the right context
  - Latency might be CPU, throughput might be GPU, may vary how an accelerator is being utilized

Actionable Items

- Performance toolsets will standardize if you show them multiple different capabilities will benefit
  - Example = VTune, LinuxPerf with AutoFDO
- Need to know more about the impact of variable optimization
  - How does this impact tools that may expect only one optimization level in a function
- Need for tools to determine what the best device a code should run on (CPU, GPU, ASIC, others…)
- Exploring new methods for diagnosing specific problems
  - ML, Inference, etc.
- How to present potentially dangerous compiler optimizations
  - Need methods to show the expected benefit
- Determining potential solution to problematic operations.