Scribing on Nov 4th
Yazhou Zu

Newsflash: The intel Core M processors.
The “Broadwell” microarchitecture. Aimed for tablet and mobile platforms
Intel's tick tock development cycle. “Broadwell” is Intel's 14nm technology node of its predecessor microarchitecture, codename “Haswell”. In other word, it is the “tick” phase.

Paper discussion: multicore CPU v.s. GPU performance on throughput computing workloads:

Argument that supports unfair comparison:
1. The optimization effort is different on different architecture. On i7 it may be unfairly optimized. (Argument: In high performance computing, the kind of optimization described in this paper is very common.)
2. The performance analysis on these products are valid, but ignoring power efficiency analysis might make it unfair.
3. The comparison didn't include memory rewrite operations which benefit GPUs a lot.
4. The delta precision operation of Monte Carlo might be unfair because it is disabled on some cards.

Additional question:
Is the maximum performance achievable going to be 7.5x as stated in the paper?
R: If data is brought into the texture cache and accessed more frequently the GPU performance will definitely be better.

Key insight:
It answers the question of “What performance improvement should we actually expect by porting throughput applications to GPU?” and “Why does the performance gap exist, is it purely due to the difference in the number of Processing Elements?”

Summary of the paper:
Microarchitecture difference:
Number of Processing Elements between CPU and GPU: 4 v.s. 30. (the main source of performance difference)
Frequency between CPU and GPU: 3.2:1.3 (2.5x helping CPU)
Memory BandWidth: 4.7x in favour of GPU
Caches: i7 has advantage
Special Cache: GPU has advantage
Synchronization: CPU has advantage
Flops: scalar 4.5x GPU
    SP: 3x-9x GPU (Fused Multiply Add)
    DP: 1.5x GPU

Design goal:
CPU: fast responsiveness, many types of applications, not latency tolerant, hard to switch thread context, large caches, large core area, fewer cores
GPU: throughput computing, graphics workload, latency tolerant, easy hardware support for multithreading, special caches, each pE is small, more cores

Hardware Recommendation for throughput computing (parallelism, amount of work completed in time matters rather than the latency):
1. High memory BW, 3D stack cache, compressed cache
2. Large caches, if working set can fit in on-die storage
3. Cache coherency support
4. High compute ability， more SMID width, more PEs
5. Gather-scatter support
6. Specialized logic
7. texture sample
8. multiple mem backs, cache ports
9. shuffle logic

Gather/scatter good for vector load/store, sparse matrix computaion applications.
Shuffle, combine values in a "funny" way.
Reduction, aggregate partial results into a final one.
VOXEL, a VOlume Element, a combination of volume and pixel, not necessarily a picture.