John Wineman
CS 390C Concurrency & Parrallelism
Scribe notes for 2/16/2012

Announcements:
- No class next Tuesday (2/21/2012)

Amdal's Law
- Used in parallel computing to predict the theoretical maximum speed up of a program using multiple processors.
- $\text{Speedup}_{\text{enhanced}(f,S)} = 1/((1\text{-}f) + (f/S))$
  - f –fraction of the computation that can be sped up
  - S – Increased speed of computation f
- $\text{Speedup}_{\text{parallel}(f,N)} = 1/((1\text{-}f) + (f/N))$
  - f – fraction of computation that can be parallelized
  - N – number of cores

Examples:
- A program runs in 100 seconds and a multiply operation consumes 80% of this time, how much do we need to increase the speed of the multiply operator to make the program run 4x faster?
  - $1/((1\text{-}.80)+(.80/S)) = 4$
  - $1/(.2+(.8/S)) = 4$
  - $1 = 4(.2 + (.8/S))$
  - $1 = .8 + 4(.8/S)$
  - $.2 = 4(.8/S)$
  - $.05 = .8/S$
  - $.05S = .8$
  - $S = 16$
  - **The multiplication operation must be 16x faster for the entire program to run 4x faster.**
- A new processor is 20x faster on search queries than an existing one. Queries account for 70% of the time spent on a computation, what speedup is gained by the new processor?
  - $1/((1\text{-}.7)+(.7/20))$
  - $1/(.3+.035)$
  - $1/.335$
  - 2.985
  - **The new processor is 2.985x faster than the old one.**
- 90% of a calculation can be parallelized. What is the maximum speedup on 5 processors? 10 processors? 1000 processors?
  - $1/((1\text{-}.9) + (.9/5))$
  - $1/(.1+.18)$
  - $1/.28$
  - **3.571x faster on 5 cores**
  - **5.263x faster on 10 cores**

- **9.910x faster on 1000 cores**
- What if 99% of the calculation can be parallelized?
  - 1/((1-.99)+(.99/5))
  - 1/(.01+.198)
  - 1/.208
  - **4.807x faster on 5**
  - **9.174x faster on 10**
  - **90.991x faster on 1000**

Amdahl's law illustrates that the maximum speedup of any program is dependent on the slowest part of the program. You'll notice that even if 99% of a program can be parallelized, you only see the speed up your expecting at 1000 cores. This is because even though 1% of the program isn't parallelized it still slows down the 99% down dramatically when ran in parallel.

Trade off **SHOULD NOT** be more parallelism at the expense of sequential optimization.

Reading – Amdahl's Law and Multicores

- Assume performance of multicores to be sqrt(n) performance
  - 16 multicore = 4 cores in parallel.
- Symmetric work
  - $\text{Speedup}_{\text{symmetric(f,n,r)}} = \dfrac{1}{\left(\dfrac{1-f}{perf(r)} + \dfrac{f*r}{perf(r)*n}\right)}$
  - **Results**
    - Amdahl's law applies to multicores because the best speedups require f near 1.
    - Using more BCEs per core, r >1, CAN be optimal. Even when performance only grows by sqrt(r).
    - Moving to denser chips increases likelihood that cores will be nonminimal. Even f = .99 achieves optimal speedup at chip size n = 16.
  - **Implications**
    - Researchers should target increasing f architectural support, compiler optimization, and programming model improvements
    - Researchers should look to increase core performance even at high cost
    - As Moore's law leads to larger multicore chips, researchers should look for ways to design more powerful cores
- Asymmetric Work
  - $\text{Speedup}_{\text{asymmetric}}(f,n,r) = \dfrac{1}{\left(\dfrac{1-f}{perf(r)} + \dfrac{f}{perf(r)+n-r}\right)}$

- **Results**
  - Asymmetric multicores can offer potential speedups that are much greater than symmetric multicores
  - Denser multicore chips improve both speedup benefit of going asymmetric and optimal performance for overall single large core
- **Implications**
  - Researchers should continue to investigate asymmetric multicore scheduling and the overhead that Amdahl's law fails to capture.

- Conclusion – Asymmetric work on multicore processors is more beneficial than symmetric work. See Slide 32 on Lecture-6.pdf for graph examples.