CSE 30321 – Computer Architecture I – Fall 2009
Lab 06 – General Feedback

- Generally, most people did the first several parts of this lab completely right. If there were any problems, they had to do with the last part.
- Even if mistakes were made in the last part, most people understood that you really needed to think about the communication overhead … and that performance got better as problem sizes got larger (as the linear-time cost to communicate between cores can be amortized over the O((N/k) log (N/k)) time of the `mergesort` on each core.)
- Because the differences between run times were somewhat small, a small deviation in your answer could lead to the wrong answer. However, this mistake was generally just a 3-point deduction as the thought process of most was correct.
- That said, below is an explanation of how the solutions were calculated. I will look at the 256-element example.

Solution:

- In general, running time of Core_Mergesort is $\log_2(N)$ (this is given in the problem)
- For power of 2 input sizes, N+1 levels of recursion are needed
  - E.g. $\log_2(N) + 1$
- In the body of the core_mergesort() function, there is a split() function call and a merge() function call which take 2 and 15 CCs respectively
  - Thus, the body takes 17 CCs
- If there are N elements, then there are 17 x N CCs
- Thus, the total time is equal to:
  - $(\log_2(N) + 1) \times 17 \times N$

For the **single core** machine, if N is = 256, then the total time is:

- $(\log_2(256) + 1) \times 17 \times 256$
- 9 x 17 x 256
- 39,168 CCs

For the **dual core** machine, 128 elements can be sorted on each core *in parallel*

- Therefore N is 128
- The sort time is:
  - $(\log_2(128) + 1) \times 17 \times 128$
  - (7 + 1) x 17 x 128
  - 17,408 CCs
- The time for the sublists to be sent back to the master routine is:
  - 70 CCs / element x 256 elements
  - 70 x 256 = 17,920 CCs
- Thus, the total time is: 17,408 + 17,920 = 35,328 CCs

The **quad core** is calculated performance is like the dual core, but now N is equal to 64.

- Thus, the time to calculate is 7,616 and the time to merge is 30,720 for 38336 CCs total time

Overall, the dual core requires the lowest number of overall CCs

Take away: More cores work best if problems are big (or independent). Otherwise, overhead can dominate.