QUIZ 4 answers Aditya Shankar (5454360)

Q1. No, a GPU may not always be better than a CPU for parallel computing. Despite the larger number of cores in a GPU, the latency per-instruction on a single GPU is much more than that on a CPU. In simple terms, this means that the GPU can work well when there are a large number of small tasks that need to be parallelized, i.e., SIMD – (Single Instruction Multiple Data) applications.

However, since the computational power of a single CPU core is much better than that of a single GPU core, so CPUs are more appropriate for use when the degree of parallelization is small, say < 10 parallel tasks. However, if this number becomes very big (1000+), then for more throughput a GPU is better.

Q2. Efficiently parallelizing the multigrid method is not trivial because there are stages when some processors are idle:

Consider the example of a 2-D grid with dimension $(2^m + 1)$x$(2^m + 1)$ elements. If there are (2^k)x(2^k) processors in total, then the number of elements per processor is 2^{m-k}. So, at the highest-level *m,* there are initially 2^{m-k} elements per processor. This reduces as the level decreases. From levels *m=m* up to *m = k,* each processor has at least 1 element. However, from levels *k-1* to 1, the number of processors exceeds the total number of elements, so there are some idle processors. This reduces the extent of parallelization.

In contrast, the Jacobi method has fixed amount of computation at every stage, so the problem of some processors becoming idle did not exist- i.e., the extent of parallelization remained the same throughout.

Q3. If we have *N* points in total and *P* processors, then the number of points per-processor would be *N/P = k.* Let is consider the data locality ratio, α, when moving from 2-D to 3-D case:

2-D: Here, the points along the edge of the square for each partition are communicated. The side of each square, and the number of sides is 4, so the total communication is $4(k^{1/2})$.

The data locality ratio is now: α = Computation time/Communication time = $k/(4(k^{\frac{1}{2}}))$

3-D: Here the points on each surface of the cube are communicated. The number of points on each face is: $k^{2/3}$, and the number of faces is 6, so the total communication is $6(k^{2/3})$.

Let's consider the case where *N/P = k* = 10.

In 2-D: $\alpha = 10/(4(10^{\frac{1}{2}}))$= 0.79

In 3-D:

α = $k/(6(k^{\frac{2}{3}}))$ = 0.3590

We see that 3-D has lower data locality ratio than 2D, so 3D parallelization is worse than 2D. Even though we have taken a specific value for N/P, the comparative performance of 3D and 2D still holds based on these values.