Essentially, approach 1 and 2 are doing similar tasks but the difference is that Matrix A is divided up differently. In approach 1, Matrix A is left alone and thus requires no additional memory for the subset blocks created in approach 2. In approach 2, Matrix A is divided up and requires additional memory for each sub-block—memory that could be used instead for processing. Approach A requires more communicating because Matrix A is p times larger than the Matrix A in approach B thus this is p times more communication with each core. Communication requires a large overhead of memory so this process could use just as much as that of dividing up the matrix. It would help to have p get larger only up to a certain point. The point at which it is no longer useful to have a large p is when the memory required for communication becomes substantially large that the benefits of p are no longer substantiated. If some of the cores are faster CPUs than other cores this will not matter because the total time for the job to finish will be determined by the slowest CPU. If all of the CPUs were faster, approach 1 would be best because it would require less memory and only p jobs.