HW2 writeup

1. concurrent execution is about servals computations are executing within the same time frame,and potentially interacting with each other.For parallel execution, many calculations are carried out simultaneously.Parallelism requires multiple pprocessing units, concurrency can be used a lot in single process unit, however, it has also been implied in multiple processing with respect to speed. For example, ife have a single processing unit, it cannot execute multiple threads at same time literally. It can only make cpu time to bunch of different partitions, and giving them to different threads. When one thread is executing, other is suspended.We call this concurrent. If a system has more than one cpu. A cpu can execute one thread ,when the other cpu is executing another thread. These two threads can execute simultaneously in every exact time point. For this way, we call it parallel.

2.

a) For the data points are generating at rate of 1000 points per second, each point will be generated in a interval of $10^6 ns$.For the question, we can know that the switching between user-space and kernel-space takes $100\mu s$ to complete on the platform, and it takes 10ns to copy one data point value from kernel memory to user memory. Therefore, the total period of time in get a data point from kernel to user space would $100\mu s + 10\text{ns} = 10^5 + 10\ ns$. It will less than $10^6 ns$. Therefore, there are no points lost due to overhead delay in this case.

b)For this case, 100000 points will release per second, it means that every point will release between every $10^4\ ns,$ And after one point shows up , the mode will switch and beginning to transfer point from kernel to user space. Because the time spends on switching mode is 10^5.Therefore, it will cause 9 points lost. And because the time of copying data points from kernel to user is $10\text{ns}.$ However, it is less than the time of next point showing up. There, it still need to wait next point shows up. After next point shows up, it will begin a next cycle. Therefore, the total fraction of points are lost due to the overhead is $\frac{9}{10}$.

c) For the buffer which size equals 1000 points, it means that it will switch mode and copy data points from kernel to user space after fill the buffer. Therefore, for the first 1000 points, it would not lose it and will wait those points fill the buffer. Then it again to switch mode and transfer point from kernel to user, it will cost $10^5 + 10 \times 1000\ ns = 10^5 + 10^4\ ns$, Points show up in this period time cannot read by the buffer and those points will miss. After that,

buffer would begin to store points again. And it will go on till all points has released. Therefore, the whole fraction of points which are lost is $\frac{9}{1000}$

d) For improving the performance, we can let the size of buffer become larger, therefore, it can transfer much more points for one mode switch. It will reduce the time spends on mode switching. Because the time of mode switching is 10^5 and it is longer than the time period of the time between two points release. It can reduce the fraction of points which lost by the overhead delay.