1.

The speedup is high for static scheduler(as by default the thread knows which index it is going to operate on) and dynamic scheduler with highest granularity as this would decrease the amount of time spent in getting the next indices.

2. For lesser values of n, there is not much speedup because not much of the code is parallelized as compared to the code when the value of n as more number of iterations occur parallelly. And hence there’s more speedup with higher n values.

Similarly, when the value of n is low speedup is far more for the lesser granularity and higher intensity. And when the value of n is high the speedup achieved is more for both the granularity values of 1 and 1000 but only when the value of intensity is equal to 1000, because this makes sure that most of the time is spent on the functions calculation rather than the overhead.

3.

The speedup for prefix sum is around 1.8 because it still has a larger part of sequential code in calculating the offsets or the sums of partial sums.

4.

The speedup for merge sort is good and it is mostly because of the bottom-up merge sort being executed in parallel.