In this document we are going to discuss the analysis of the gprof and valgrind report of our original code and then are going to discuss how we have modified our code using the analysis of our original code.

In our original code while profiling, we were seeing that while having gprof analysis we were having a lot of percentage of our runtime in timedwork function in which we were actually calling classify and hence we have to optimize it i.e. we have to basically decrease the percent of runtime taken by this function which we have concluded from the gprof analysis. Apart from that we can also try to decrease the runtime of our algorithm i.e. the time taken by our code to run and for that we can decrease the time complexity which searching the ranges and this can be reduced from linear to binary and then to ternary search which will then decrease our total running time although thus will have no effect in our parallelization or distributiveness of the program.

Now from the valgrind report we can conclude that in our original code we were having a lot of L cache references, D cache references and LL cache references and also there are a lot of D misses and the miss percentage is actually 5.9% (6% while reading and 1.5% while writing) and we have to basically decrease it somewhat and after the optimizations in our code we were able to cut it short i.e. the references of LL caches actually decreased and also the d misses were decreased by a significant number. Also, we have removed some of the bad segmentation errors from the code which we were able to get from the memory leak part of valgrind.