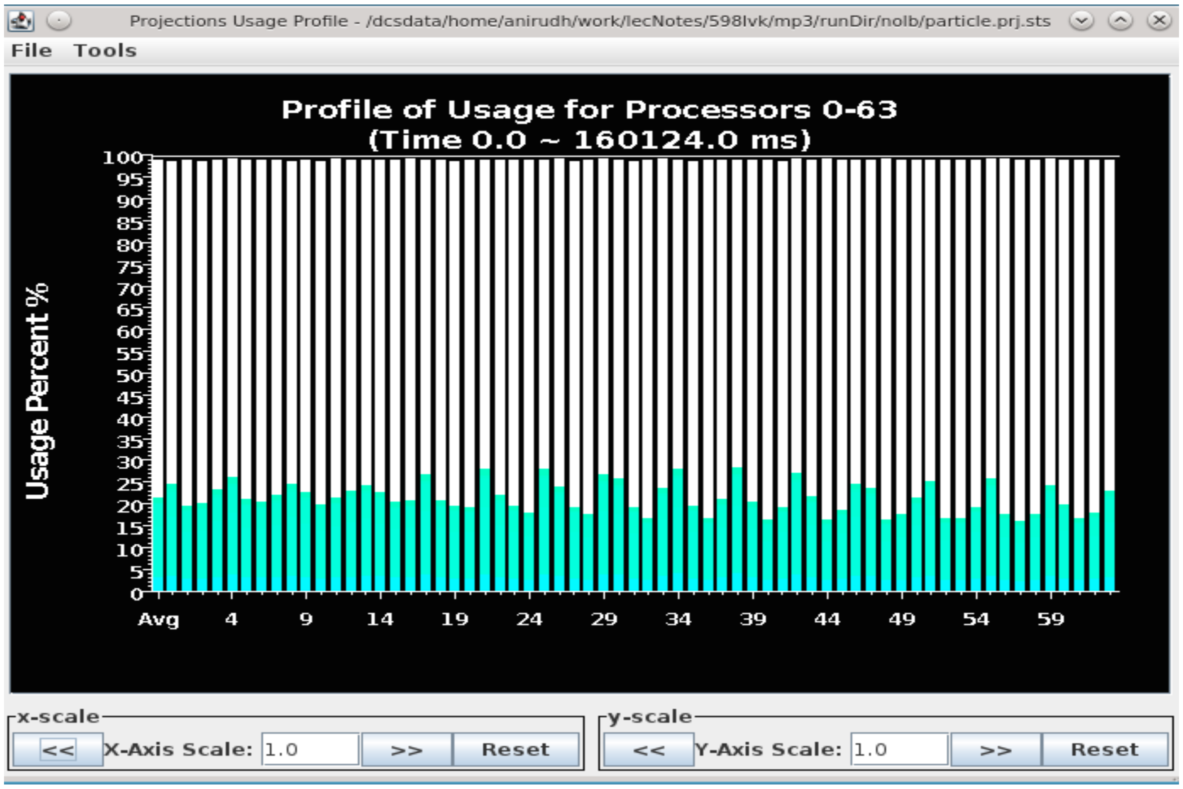
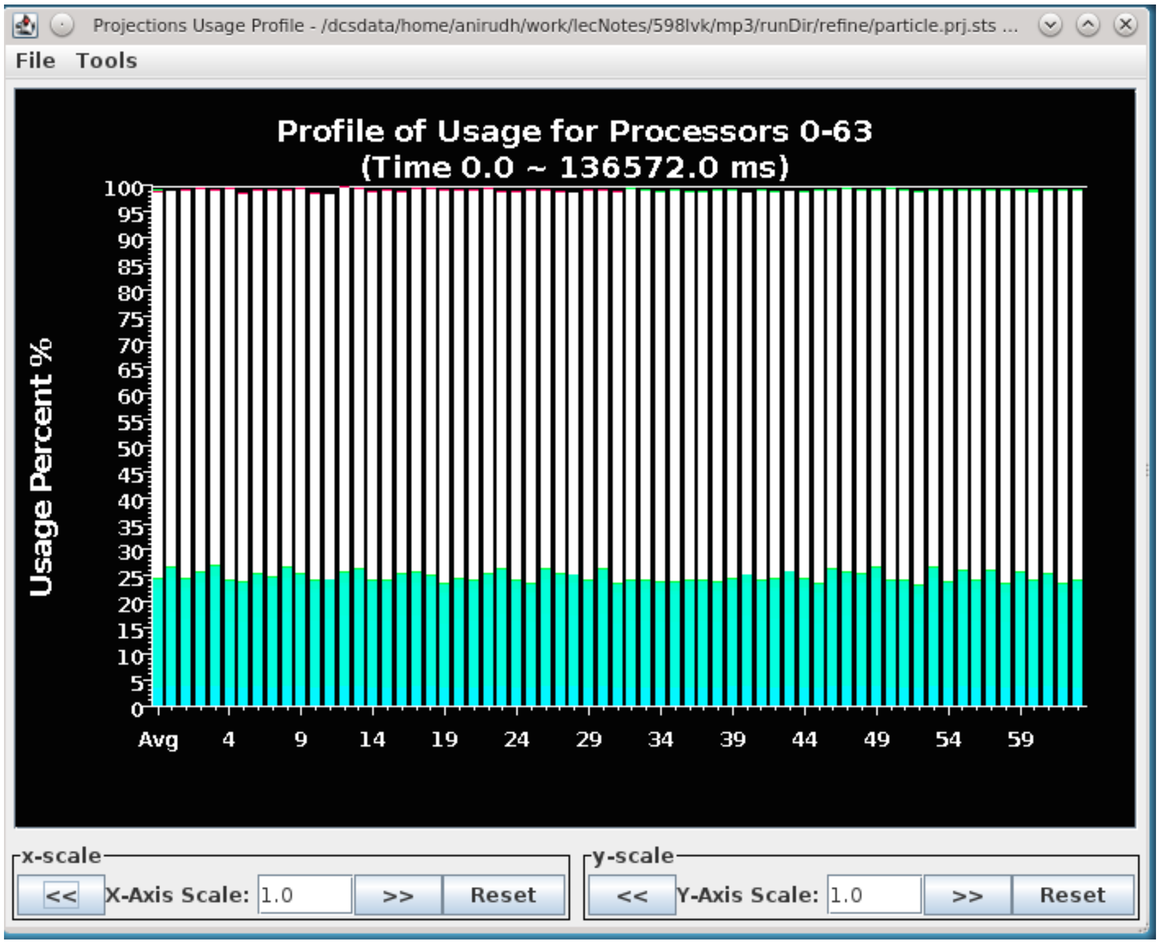
Report: Effect of load balancing stratergies on the particle code.

All experiments are done with the following configuration. No of particles per chare = 100000, array size = 16x16.

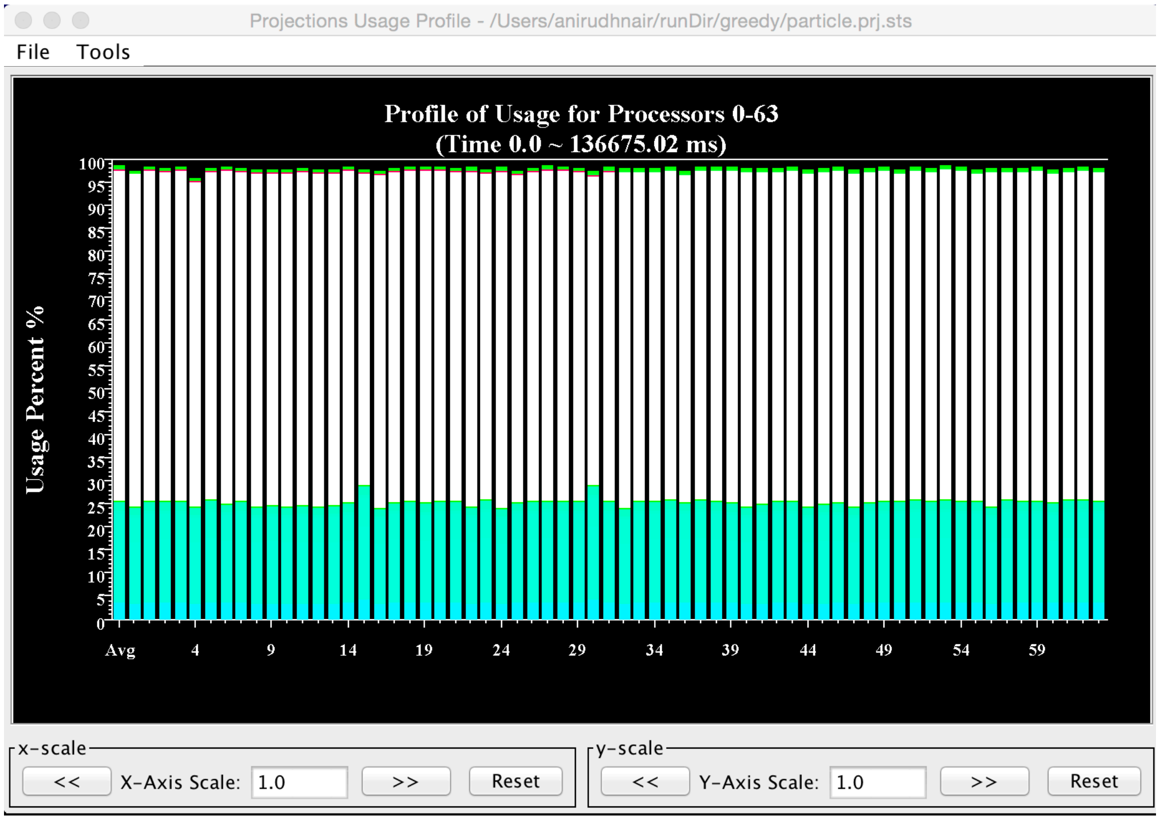
Without loadbalancing: Total time: 159 secs. Avg Idle time: 124.6 secs


Projections Usage Profile - /dcsdata/home/anirudh/work/lecNotes/598lvk/mp3/runDir/nolb/particle.prj.sts
File Tools
Profile of Usage for Processors 0-63
(Time 0.0 ~ 160124.0 ms)
Usage Percent %
Avg 4 9 14 19 24 29 34 39 44 49 54 59
x-scale
<< X-Axis Scale: 1.0 >> Reset
y-scale
<< Y-Axis Scale: 1.0 >> Reset

Refine LB: Total time: 136.36 secs, Avg Idle time: 101 secs
The refineLB scheme does a good job at load balancing. The performance improved by 14.23%. The increase in performance is a direct result of the decrese in the idle time. The idle time decresed by 18.9%.

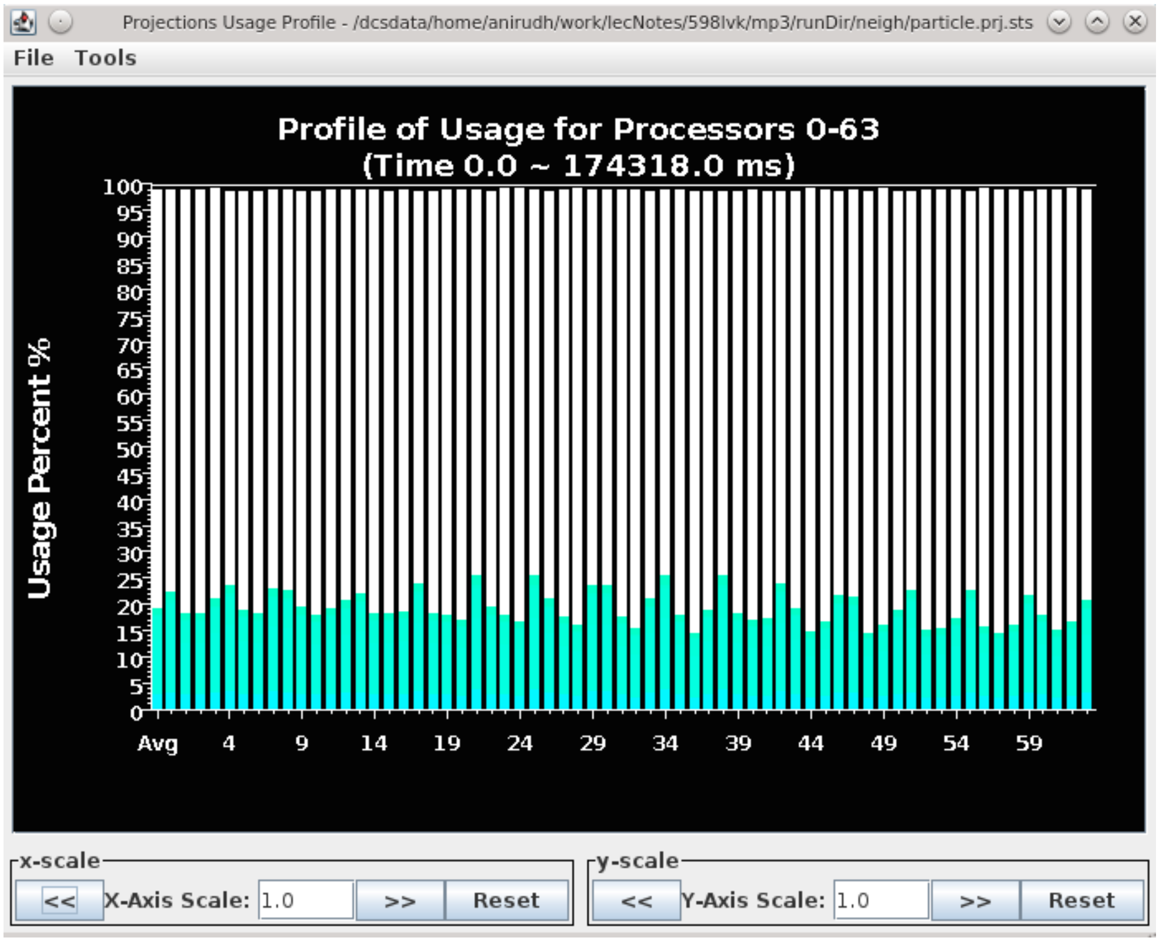


GreedyLB: Total time: 272secs
GreedyLB always assigns the heaviest object to the least loaded processor. Hence, there is a lot of migration of objects in the system. This is the possible reason for the bad performance of the GreedyLB. From the LBdebug logs around 44 sec were used for migrating the objects. That is a huge overhead inorder to increse the utilization. Also, taub's system utilization seemed to be high during the time of the greedyLB job execution. This could also have contributed to the incerase in time.



NeighborLB: Total time: 174 secs

Since NeighborLB is a distribute technique, it take a while before it converges to a good loadbalancing solution. In this case we have 100 iterations and around 10 loadbalancing points. This scheme does a poor job at loadbalancing for this run. Since distributed techniques are primiarily used at higher processor count the benefits are not seen for here. The performance is decreased by 9.43% when compared to the run without any loadbalancing. The two take away points are 1) use neighborLB when the application if expected to iterate over a long range 2) use neighborLB for higher core count.



Increasing the LB period from 10 iterations to 20 iterations yielded interesting results for refine. Though the expectation was that the execution time would decrease slightly, the empirical results show an increase in execution time. The refineLB run with load balancing every 20 iterations is 277 secs. This increase again could be due to taub's increased utilization resulting in network contention.