601.420/620 Parallel Computing for Data Science
Homework 6: Ray ResNet

Aravind Kavuturu

November 24th, 2024

1 Does the computation for a single input file (normalization and classification) run in serial or parallel? If serially, how is the dependency enforced?

The computation runs in series, as the output of the normalization step is required for the classification step. Ray enforces this when we send the norm_oid into the remote call of the classification step, which tells Ray that we need to schedule the classification step after the normalization step is completed.

2 Does the computation of different files run in serial or parallel? If parallel, explain why they are independent.

The computation for different files is run in parallel. With different Ray actors, we can handle the computation for different files independently, as each actor will work on their own file. Since the normalization of file A does not depend on the normalization or classification of file B, we can do these files in parallel using different actors.

3 This version has about the same runtime as the version in Activity 22 that does normalization and classification in one actor.

(a) In what configuration would it be fast to do them together?

If you have a small dataset where the parallelism might not be as pronounced, then the overhead of using many actors would outweigh the speedup you gain. Additionally, if the tasks we are performing (normalization and classification) are not computationally expensive, then using only one actor could be better than splitting the work up and potentially leading to delays in the overall computation. In terms of target hardware, we would want to use only one actor when we have limited CPU/GPU resources, as we won't be able to take advantage of the multiple actors.

(b) In what configuration would it be faster to do them separately?

Conversely, if we have a large dataset, we will see a significant speedup with using many actors, since the parallelism will be greater when working on multiple actors at once. Additionally, if the normalization/classification tasks are very expensive computationally, then we will want to split up the work among many actors to ensure that as much of the work is parallelized as possible. In terms of target hardware, we would want to use multiple actors when we have extensive compute resources, such as distributed/cloud compute, efficient GPUs, or even simply multi-core CPUs. Any improvement to the hardware can be directly translated to a speedup in the program if we match the number of actors to our available resources.