COMP90025 ASSIGNMENT 02B WRITTEN REPORT

A PREPRINT

Xulin Yang
904904
The University of Melbourne
xuliny@student.unimelb.edu.au

September 27, 2020

In this task, I was given k input sequences and computed the sequence alignment between all $total = \frac{k(k-1)}{2}$ pairs. I use MPI+openMP hybrid approach to achieve parallel speedup. Assume we have 12 nodes for simpler explanation.

1 pure openMPI

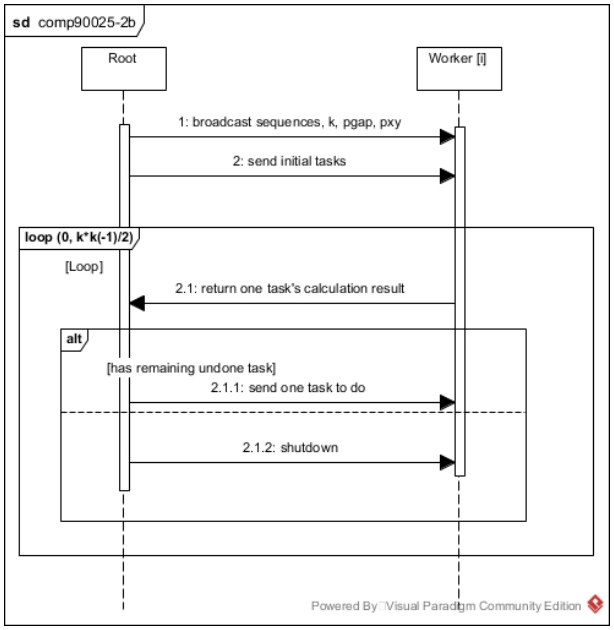

Figure 1: OpenMPI sequence diagram

As shown in the figure 1:

1. In 1st step, root node broadcasts sequences to all workers.[1]
2. Then the root node stores all tasks[2] to be done in a queue.
3. In the 2nd step, iterate all possible nodes, pop a task from the queue and send it to the worker. If the queue is empty in the iteration, a stopping message will be sent instead.

[1] Although one node might not require all sequences to calculate all its tasks, the communication to transport this information is not costly. So broadcast is enough and no need to design complicated information exchange protocol.

[2] one task = index of two sequences to be calculated + its task id

4. In the loop, the root node receives $total$ number of calculation results[3] from worker i. If the queue is not empty[4], keeps sending task to the worker i. Otherwise, send a message to stop the worker i.
5. Finally, aggregates all resulting hashes to one hash and return.

2 pure openMP

The calculation of 2 sequence alignment on each worker is imported from 2A's work.

3 Hybrid openMP + openMPI

To maximize utilization, root node should also be a worker[5]. The other 11 worker nodes use all 16 threads for alignment calculation. The root uses 1 thread for the job mentioned in section 2. 1 thread works as a worker to receive tasks as well as sending back results[6]. And 15 threads reopened[7] from the second thread for alignment calculation.

[3] one calculation result = task hash + task penalty + task id

[4] still some tasks not done

[5] All 12 nodes are workers, the root node is both responsible for dynamic task scheduling mentioned in the section 1 as well as doing calculation mentioned in section 2

[6] "MPI_Init_thread" instead of "MPI_Init" is used to enable openMPI communication between openMP threads.

[7] Using "omp_set_nested(1)" make sure threads can be reopened from one thread in openMP.