Word Count Problem & Cross Correlation Calculation

MPI Implementation

Filipe Pires | 85122 | filipesnetopires@ua.pt
João Alegria | 85048 | joao.p@ua.pt

University of Aveiro, DETI

June 7, 2020

Multi-Thread to MPI Mapping

Given that a multi-thread solution was already implemented, the team's efforts were focused on mapping the it to the MPI environment. The required mapping was:

- The previous central entity(monitor) was translated to a dispatcher that is responsible for the same functionalities and is implemented in a predefined process created by the MPI platform.
- The dispatcher is responsible for keeping track of files to process, provide data chunk when requested, stored partial results and print final results.
- The previous thread workers are now implemented as process supported by the MPI platform.
- There logic keeps the same, but now they wait for a data chunk to process, sending the obtained results to the dispatcher.

Word Count Entity Interaction

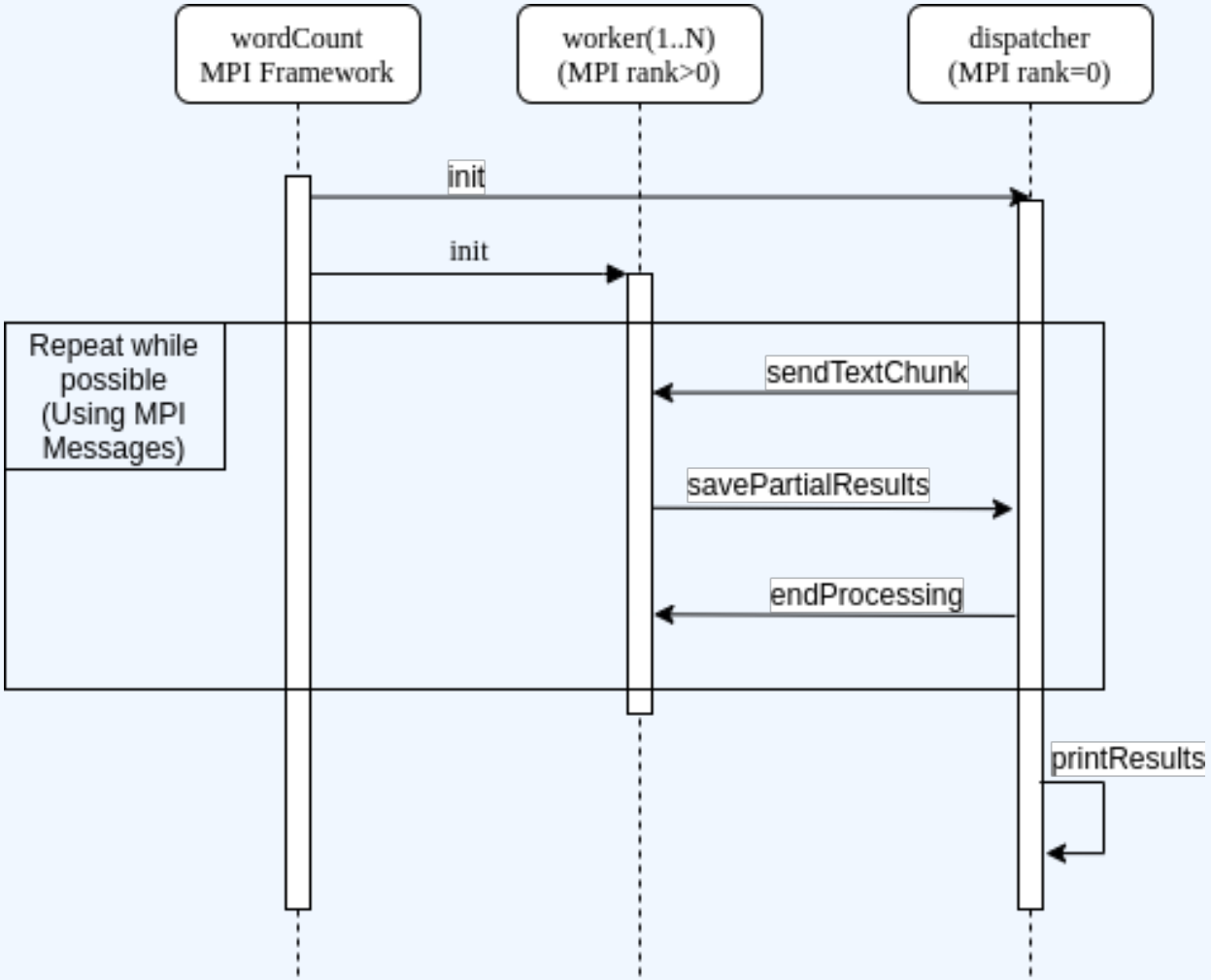

Figure: Entity Interactions

Cross Correlation Entity Interaction

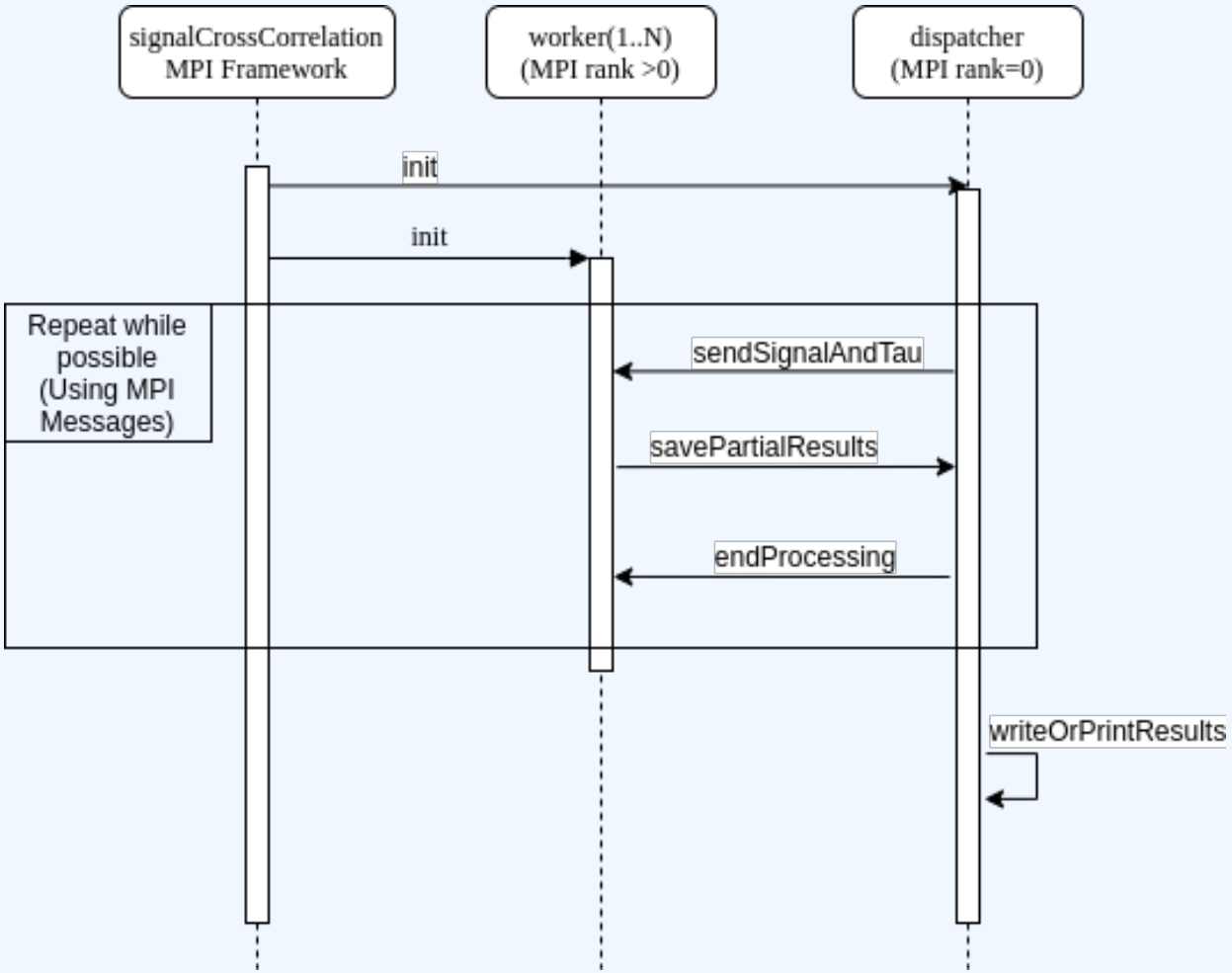

Figure: Entity Interactions

Results

As expected, the more worker processes the user deploys, the faster it delivers the results.
By analyzing the table below, it is not difficult to see an inverse relation between the number of threads and the execution times.

		Average Execution Time (s)							
		Problem 1				Problem 2			
		text1	text2	text3	text4	sigVal1	sigVal2	sigVal3	sigVal4
# of Threads	1	0.004	0.038	0.010	0.026	0.006	0.132	2.408	36.985
	2	0.005	0.020	0.008	0.015	0.006	0.067	1.764	25.453
	4	0.002	0.019	0.006	0.013	0.005	0.0657	1.334	21.481

Table: average execution time of running the programs 10 times for each input file and for each different number of worker threads.

Conclusions

The results of implementing a multi-process version using the MPI library for the different problems proved to be a good option to the multi-thread approach. By carefully orchestrating the worker processes, we were able to speed up our solutions up to 2 times. In fact this achievement can be greater if there are more CPU cores and therefore supports a higher number of processes. A conclusion we made during tests was that if the number of processes used exceed the number of possible threads in the system, the application can suffer in performance, possibly due to the action of exchanging different processes in and out the CPU, which can be costly.
For future work, one possible improvement would be do adopt different computation approaches, such as using multidimensional techniques for the cross-correlation calculation.