Assignment 1 2INCO

Robert Jong-A-Lock 0726356
Tom Buskens 1378120
September 24, 2018

1. Overview

There were 3 main challenges while writing this program:

1. Managing the child processes
2. Synchronizing communication
3. Optimizing job generation

2. Managing the child processes

The exercise dictated that the communication had to be done through 2 message queues. To optimize the program it is desirable to have as less communication as possible between the processes. Therefore the child processes will only send results when they actually found a hash. This reduces communication significantly but it makes it impossible to know when the child have finished computing an job that has no result. It is also important that a child only terminates when there are no jobs left.
Therefore it is chosen to send a killyourself messages after all jobs have been added for each child These messages will result in the child sending a suicide note back and terminating. By counting the suicide notes you make sure that all children are finished and children can wait on the job queue even when it is empty.

3. Synchronizing communication

For the communication 2 message queues are used. When these queues are full the send method can be used to block the process until there is room in the queue. This behavior can be used to not rely on busy waiting which was prohibited by the exercise. But it makes it necessary to make these blocking actions in such a way that a deadlock is not possible. The following synchronization tactic used in the final program.
The farmer has a blocking send on the job queue and a non blocking receive on the result queue. Whenever the farmer adds a job to the job queue the entire result queue is emptied. The workers have a blocking receive and a blocking send.
This works because whenever a worker queues a result it takes a new job from the job queue. Taking this new job will unblock the farmer which can processes the result queue.
Unfortunately this does not work any more when the killyourself messages are being send because after this job, a child does not take a new job. Therefore the farmer uses a blocking receive when these messages are being send. This blocking receive is looped until the suicide note is received belonging to the recent kill yourself. This prevents a deadlock.

4. Optimizing job generation

To optimize the program a few extra things are implemented. Whenever a hash is found no new jobs are queued asking for this hash. Risks of having already found hashes in jobs is spread out by asking as many different hashes as possible instead of filling the job queue with multiple jobs for the same hash.
At last the amount of letters that are generated by the worker is variable. This makes it possible to let the farmer generate more smaller jobs. This is convenient because it’s now possible to check the result queue more often and prevent already found hashes to enter the job queue.
It is still possible to optimize further by calculating a hash once and comparing them against all hashes that have to be found. Instead of calculating the same hash for different hashes that have to be found.