University of St Andrews

CS4204 Coursework 1

Parallel Patterns

Author:
150008022

April 26, 2019

Goal

To implement and evaluate a library for parallelising C programs, using PThreads, locks, and queues.

1 Pipeline

The pipeline pattern is where a series of functions are applied to some input. Often the analogy of a conveyor belt is used, as inputs can flow continuously and functions can be applied simultaneously to inputs that are at differenet stages of the pipeline.

Figure 1 shows how an atomic pipeline was implemented for this submission. An array of functions are submitted on creation of the pipeline, along with the number of worker threads that should run for each stage in the pipeline. The pipeline interface allows for inputs to be added to a queue, where they will wait to be processed, and for outputs to be polled from the outgoing queue.

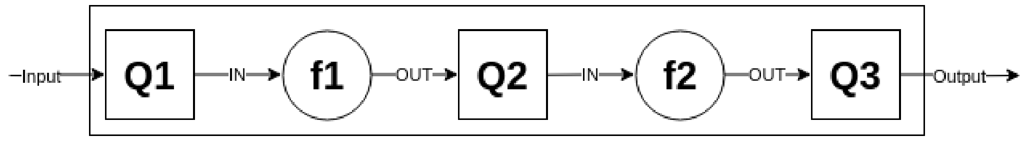

Figure 1: Pipeline abstraction, where Q signifies a blocking queue, and f signifigies a function being applied.

The pipeline was defined to consist of a series of steps which would be managed by a thread. The thread for step i would poll queue Q_i for an input x to process, compute $f(x)$, then add the output to queue Q_{i+1}. Q_0 and Q_{n+1} are the input and output queues made accessible by the pipeline interface, where n is the number of functions in the pipeline.

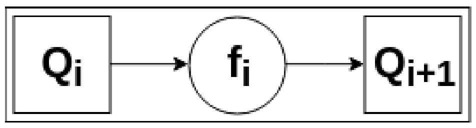

Figure 2: Step of pipeline.

2 Parallel Farm

The parallel farm pattern involves a pool of worker threads that remain idle until assigned a task by a coordinator. Once a worker thread completes its task, it then returns to being idle until a new task is assigned. Parallel farms are useful as they avoid the overhead involved .

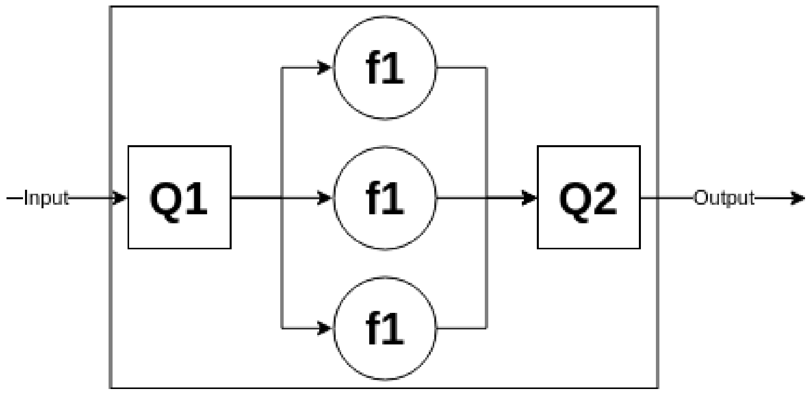

Figure 3: Farm abstraction

Conclusion