PA5: The Server Moved Out
Christopher McGregor - 726009537

1 Design

This PA focuses on separating the client and server when communicating. Because of this, we now need to implement communication over a network – in our case we've gone with TCP. I will briefly describe the needed changes to implement this.

NOTE: PA5 was demoed in lab with Feras

1.1 TCPreqchannel.h

This outlines the bulk of the work for this PA as this is where we define our needed TCP class for communication. TCP communication now only requires one file descriptor for the socket. New to this PA, we now need a constructor which takes in a host name and port, another constructor who only takes in a file descriptor, and a getter function for the file descriptor. From FIFO in PA4, we still need a cread and cwrite function.

1.2 TCPreqchannel.cpp

This is the brains of this PA. The regular constructor first decides if we are the server or client – this is decided by whether or not a hostname is supplied. If no hostname is given, we start the server on the specified port. If a hostname is given, we attempt to connect. Other functions are much like FIFO implementation except adapted for TCP.

1.3 client.cpp

Minimal changes are needed here. Mainly, we need to update command line arguments to take in a hostname and port, and we no longer care about histogram threads. Other than that, we simply need to make sure we're using our TCP class rather than FIFO.

1.4 server.cpp

Minimal changes are needed here too. Notable changes are: we no longer need to process new channels, we need to support a port number from the command line arguments, and our main loop is now used for accepting connections.

2 Runtime analysis

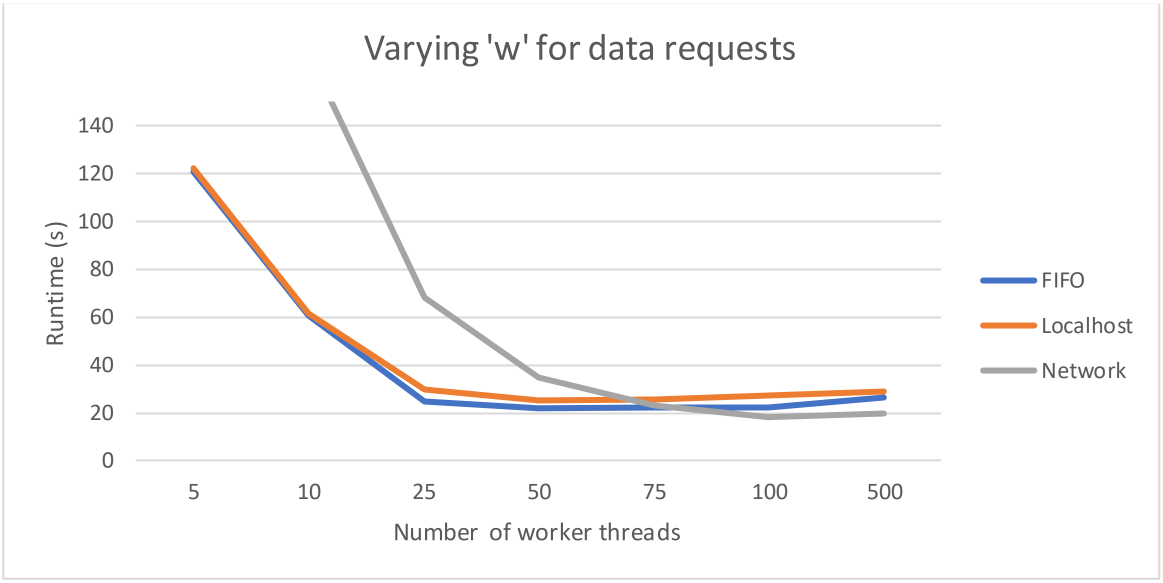
We plot run times against running in TCP with local host, TCP across a network, and using FIFO from PA4.

2.1 Data requests

We use the following parameters across all tests:
-n 15000 -p 15 -b 50 -h 5

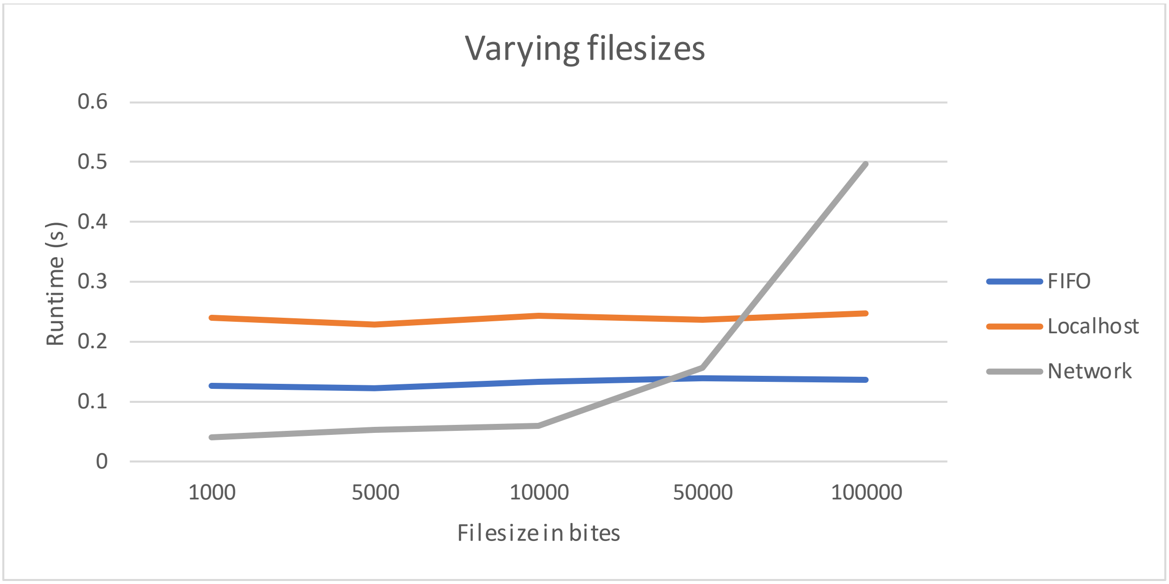
We notice in general that in order of decreasing run time we have FIFO as the fastest followed by requests over a network and localhost respectively. This is expected as each test sends a packet 'further' than the last. In that, for FIFO we run everything within one process. When using local host, packets stay within the same network. And when sending requests across a network, the packet travels through the internet. It is worth noting that FIFO and localhost runtimes are very close since like mentioned above, the packet does not need to travel across the internet.

Comparing to PA4, we see the point of diminishing return differ at about w = 50 for FIFO, but around 100 for TCP over a network. Additionally, we see a limit to connections with the base ulimit at around w = 1500.



2.2 File Requests

We test transferring various file sizes across FIFO, TCP localhost and TCP over a network. We notice that FIFO and requests across localhost have about constant runtime despite the file size – and FIFO is faster than localhost. However, we see that requests over a network scale with filesize.


Varying filesizes
Runtime (s)
0.6
0.5
0.4
0.3
0.2
0.1
0
1000
5000
10000
50000
100000
Filesize in bites
FIFO
Localhost
Network