CS 342

Project 2

Ecem İlgün (21502157) & Yusuf Gürkan Bor (20900461)

The programs ./server and ./client was executed with parameters:
/shmname = {“/shm”, “/sh”}
/semname = {“/sem”, “/se”}
keyword = {“one”, “two”, “three”, “four”, “five”, “six”, “seven”, “eight”, “nine”, “zero” }
filename = {“text1.txt”, “text10.txt”, “text100.txt”}.
General command structure of our testing was running server process with “./server /shm text.txt /sem” in one terminal and running client processes with “./client /shm k /sem” in another terminal where k was one of the elements of keyword set described above.

- We used 5 different text files:
  text1.txt was a 9 lined text of “ one two three four five six seven eight nine zero\n
  two four six eight zero\n
  three six nine zero\n
  four eight zero\n
  five zero\n
  six zero\n
  seven zero\n
  eight zero\n
  nine zero\n”
  text10.txt was repeating text1.txt’s 9 lines 10 times, therefore having 90 lines.
  text100.txt was repeating text1.txt’s 9 lines 100 times, therefore having 900 lines.
  We aimed to **test the correlation between running time and file size** with that.
- We had 10 different keywords in order to be able to make 10 different simultaneous requests each with another keyword. We used 1, 2, 4, 6, 8, 10 keywords to **test the correlation of running time with the number of keywords**.

- We executed the clients with either concurrent (& between commands) or sequential (&&between commands) fashion to test **the timing and parallelism**.

The experiments took place in a Virtual Machine of Ubuntu Linux (64-bit), version 16.04. The
system’s base memory is 2048 MB and VM has 1 CPU. Hardware acceleration is enabled.

requests	t1	t10	t100
1	0.00666	0.001	0.0044
2	0.002	0.00225	0.068
4	0.00265	0.00433	0.0534
6	0.00755	0.00975	0.0984
8	0.00966	0.0125	0.157666
c	0.008666	0.01625	0.249787

Table 1. Request Amount {1,2,4,6,8,10} versus Textfile size {9 lines, 90 lines, 900 lines}

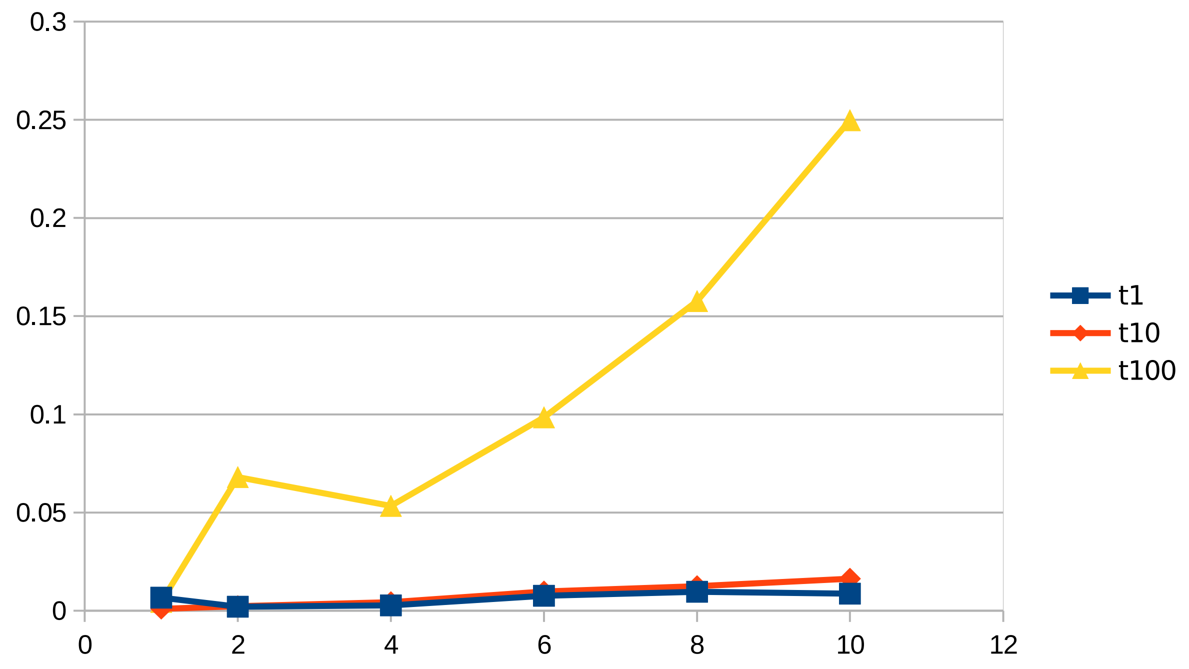

Figure 1. Request Amount {1,2,4,6,8,10} versus Textfile size {9 lines, 90 lines, 900 lines}

requests	sequential	concurrent
1	0.001	0.001
2	0.001	0.00225
4	0.186	0.00433
6	0.214	0.00975
8	0.02633	0.0125
10	0.041	0.01625

Table 2. Request Amount {1,2,4,6,8,10} versus Sequential or Concurrent Execution size

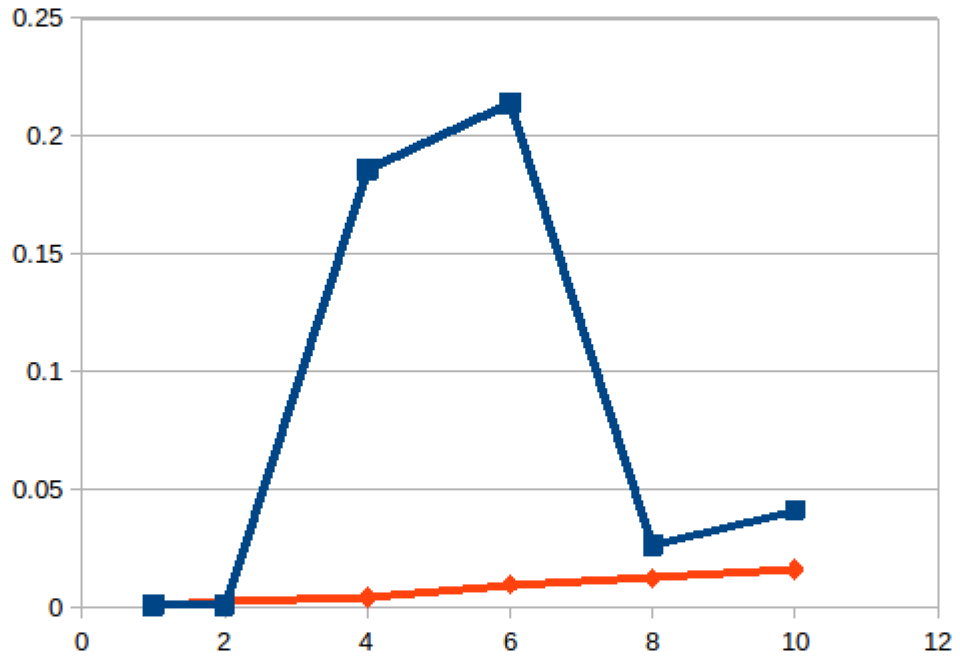

Figure 2. Request Amount {1,2,4,6,8,10} versus Sequential or Concurrent Execution size