

CS744 - PA4
Parismita Das
22m0815

System:

Cores: 8, Processors 16

I have used core number 0 for single core server and 0-1 for multicore server and 5 to 15 for the load generator.

On server side:

I have kept threads = 50,

Listen Queue = 20000

Queue size = 20000

Load generator:

I have plotted for few cases as mentioned below,

Case 1:

Think time = 0.01

Test Duration = 60

Users = 5000

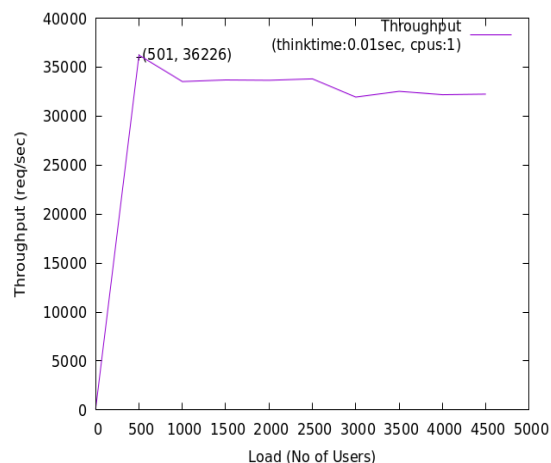
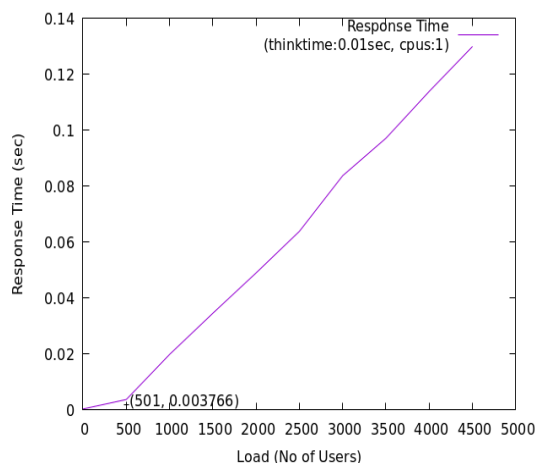
Case 2:

Think time = 0.001

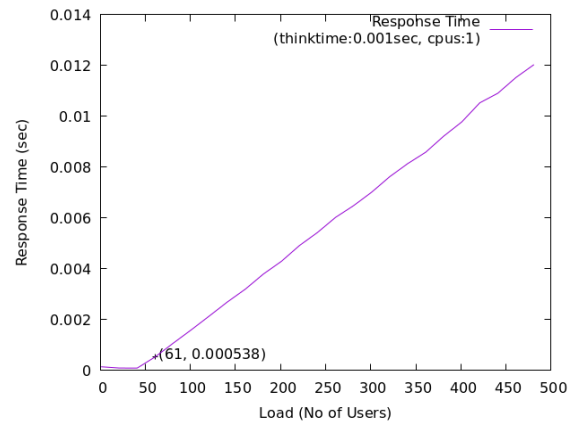
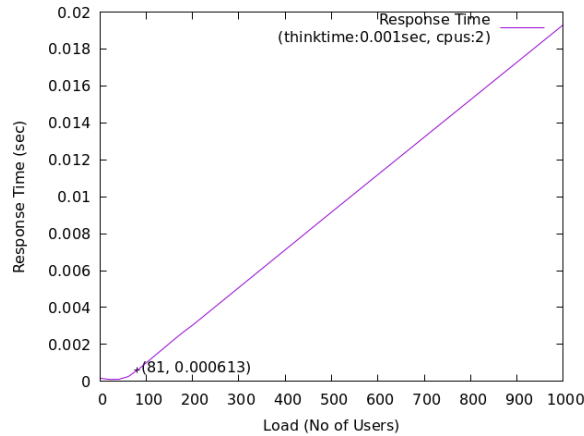
Test Duration = 60

Users = 500

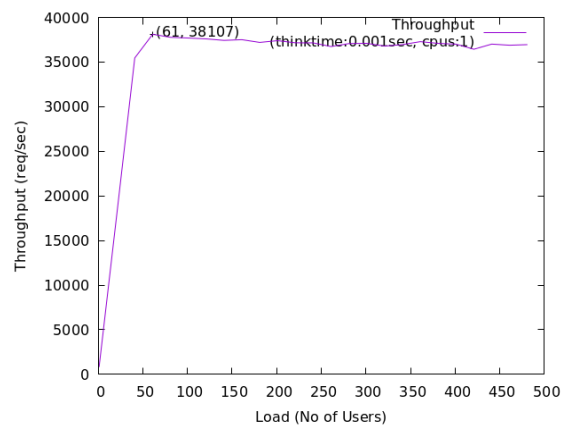
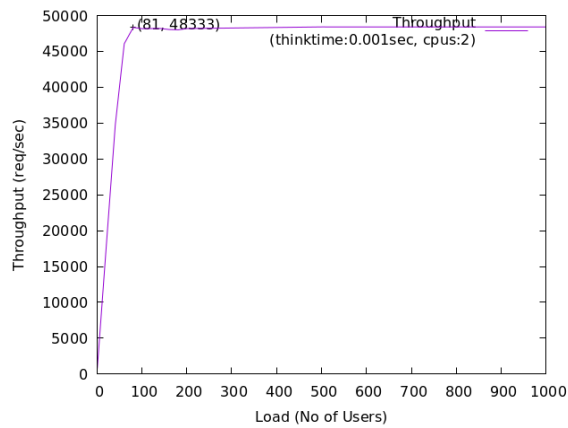
Here are my plots:



Response time and throughput for Case 1 respectively.



Response time plot for 2 processors and 1 processor respectively. For Case 2.



Throughput for double and single processor respectively for Case 2.

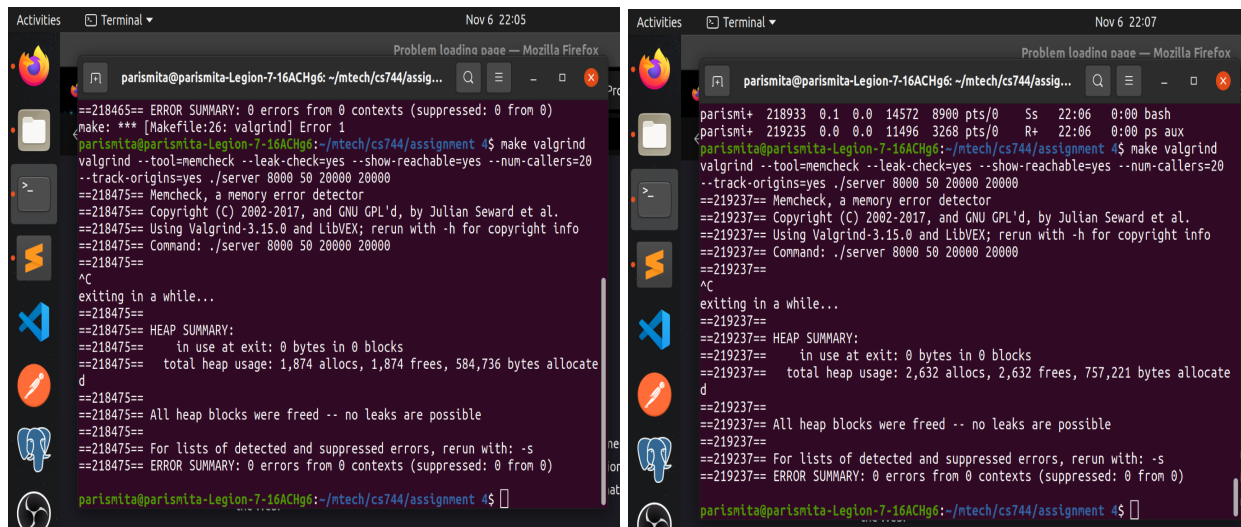
Observation: Closed loop testing.

For a single core, the throughput is going to be around 38000 req/sec. And response time is 0.538 msec when I have my think time as 0.001 sec and the saturation point is 61 users. Hence my system capacity for single core is 38000 req/sec and service demand of bottleneck which is the CPU in this case, as CPU is 100% utilized, is 26 microsec. The turnaround time is coming to be 1.538 msec. The optimal user value is around 57 users and my plot shows it as 61 users which is nearby as I load in units of 20 req/sec. Hence proves that my results are accurate.

Similarly for 2 cores, throughput is saturating at 48,000 req/sec. Hence capacity is 48000 req/sec and response time at that point is 0.613msec for 81 users. The CPU was utilized 100% hence its the bottleneck. The TAT is around 1.6msec as think-time is 0.001sec. The optimal values for users come around 72 users which is similar to my results.

The bottleneck is the CPU for all the above cases, I have checked it using top.

Here's my report on valgrind memory leak.



```
parismita@parismita-Legion-7-16ACHg6: ~/ntech/cs744/assig...  
==218465== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)  
make: *** [Makefile:26: valgrind] Error 1  
parismita@parismita-Legion-7-16ACHg6: ~/ntech/cs744/assignment 4$ make valgrind  
valgrind --tool=memcheck --leak-check=yes --show-reachable=yes --num-callers=20  
--track-origins=yes ./server 8000 50 20000 20000  
==218475== Memcheck, a memory error detector  
==218475== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.  
==218475== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info  
==218475== Command: ./server 8000 50 20000 20000  
^C  
exiting in a while...  
==218475==  
==218475== HEAP SUMMARY:  
==218475==    in use at exit: 0 bytes in 0 blocks  
==218475== total heap usage: 1,874 allocs, 1,874 frees, 584,736 bytes allocate  
d  
==218475==  
==218475== All heap blocks were freed -- no leaks are possible  
==218475==  
==218475== For lists of detected and suppressed errors, rerun with: -s  
==218475== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)  
parismita@parismita-Legion-7-16ACHg6: ~/ntech/cs744/assignment 4$  
  
parismita+ 218933 0.1 0.0 14572 8900 pts/0 Ss 22:06 0:00 bash  
parismita+ 219235 0.0 0.0 11496 3268 pts/0 R+ 22:06 0:00 ps aux  
parismita@parismita-Legion-7-16ACHg6: ~/ntech/cs744/assignment 4$ make valgrind  
valgrind --tool=memcheck --leak-check=yes --show-reachable=yes --num-callers=20  
--track-origins=yes ./server 8000 50 20000 20000  
==219237== Memcheck, a memory error detector  
==219237== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.  
==219237== Using Valgrind-3.15.0 and LibVEX; rerun with -h for copyright info  
==219237== Command: ./server 8000 50 20000 20000  
^C  
exiting in a while...  
==219237==  
==219237== HEAP SUMMARY:  
==219237==    in use at exit: 0 bytes in 0 blocks  
==219237== total heap usage: 2,632 allocs, 2,632 frees, 757,221 bytes allocate  
d  
==219237==  
==219237== All heap blocks were freed -- no leaks are possible  
==219237==  
==219237== For lists of detected and suppressed errors, rerun with: -s  
==219237== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)  
parismita@parismita-Legion-7-16ACHg6: ~/ntech/cs744/assignment 4$
```

To avoid possible leaks on ctrl+c, I have used a signal handler which sets a flag to exit the thread and the server is closed after all threads are exited.

Additional Code to server:

- 1) Introduction to flag
- 2) Signal handler
- 3) Closing socket fd where it was not closed
- 4) Freeing heap variables in http_server.cpp