PA02-Network I/O

GRS

NAME-Arpit Kumar

ROLL NO- MT25017

GITHUB Repository Link:-https://github.com/Arpit2919/GraduateSystem

1.Implementations:

A1 Two-copy:
Data is copied twice ,kernel to user buffer to kernel socket to buffer that causes higher CPU and memory overhead. Uses send()/recv().

A2 One-copy:
Data is copied only once between user and kernel, reducing memory operations and improving throughput.Uses Sendmsg().Reduces copy operations so better cache locality.

A3 Zero-copy:
Data is copied only once between user and kernel, reducing memory operations and improving throughput.Uses sendmsg() with MSG_ZEROCOPY.Uses kernel-assisted zero-copy (mmap/splice/sendfile).Avoids data copying so lowest CPU work for large transfers.

2.Measurement:

Test Parameters:
Message Size: 256,512,1024,4096 bytes
Threads:1,2,4,8
Versions:A1,A2,A3
Total Experiments:3*4*4=48

Metrics Collected:
Throughput:(bytes * 8)/time
Latency:time/message
CPU Cycles & Cache misses: measured using perf stats -e cycles,cache-misses

All performance counters are collected on the server side.
Reasons:Server handles all packet processing,gives accurate CPU + cache behavior.

3.Results:

Namespace creation:

```
# Create namespaces
sudo ip netns add server_ns
sudo ip netns add client_ns

# Create veth pair
sudo ip link add veth0 type veth peer name veth1

# Assign to namespaces
sudo ip link set veth0 netns server_ns
sudo ip link set veth1 netns client_ns

# Configure IPs
sudo ip netns exec server_ns ip addr add 10.0.0.1/24 dev veth0
sudo ip netns exec client_ns ip addr add 10.0.0.2/24 dev veth1

# Bring up interfaces
sudo ip netns exec server_ns ip link set veth0 up
sudo ip netns exec server_ns ip link set lo up
sudo ip netns exec client_ns ip link set veth1 up
sudo ip netns exec client_ns ip link set lo up
```

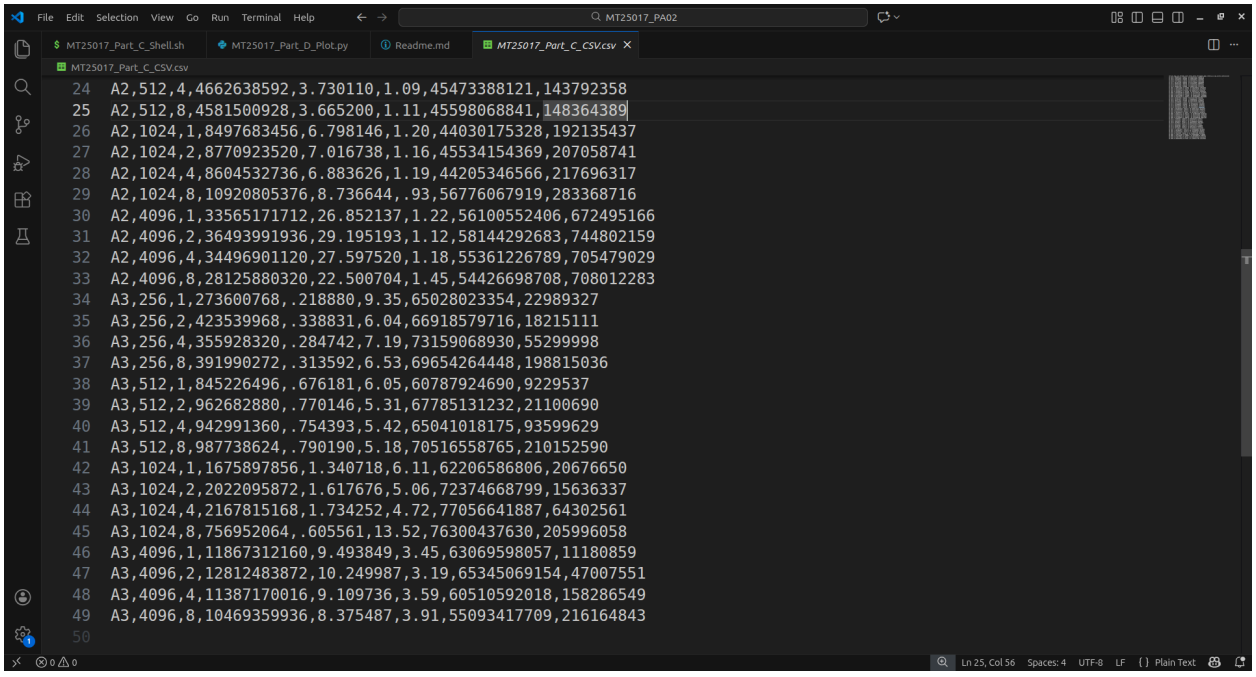
Running shell script:

```
chmod +x MT25017_Part_C_Shell.sh
./MT25017_Part_C_Shell.sh
python3 MT25017_Part_D_Plot.py
```

Perf stat output:

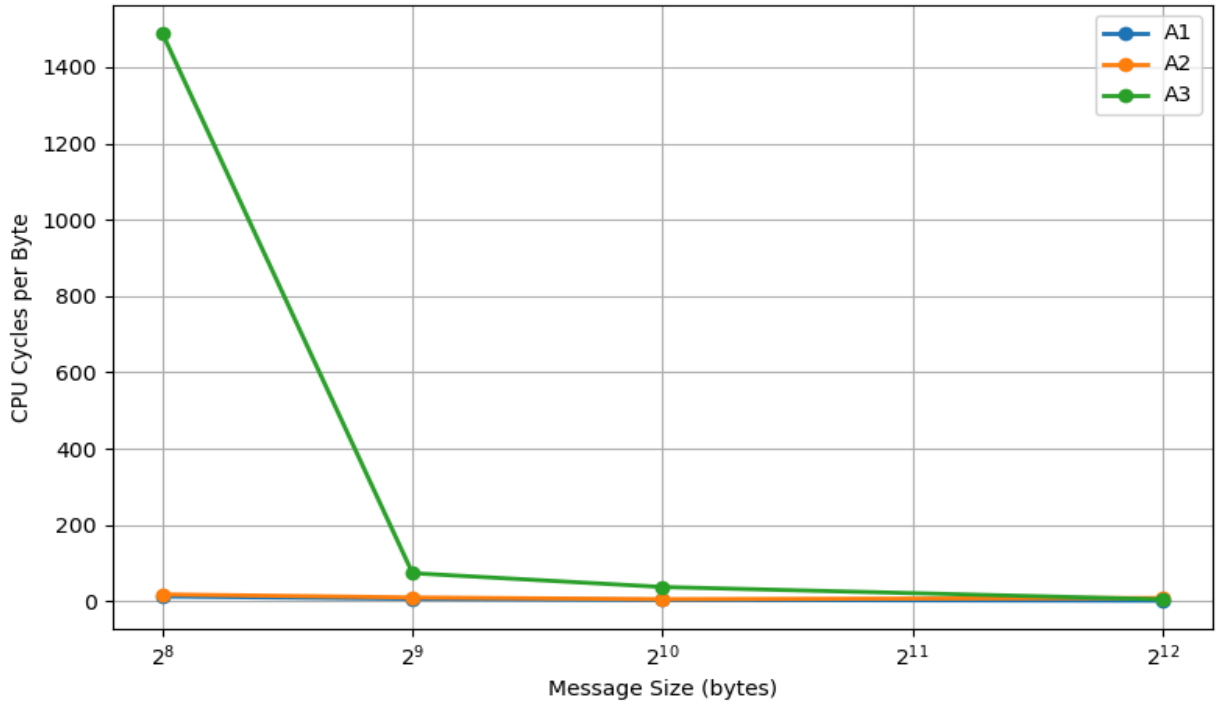
MT25017_Part_C_CSV.csv

```
version,msg_size,threads_count,total_bytes,throughput_gbps,latency_us,cpu_cycles,cache_misses
A1,256,1,3698026752,2.958421,.69,45053131831,106658710
A1,256,2,3793310976,3.034648,.67,45802778554,110948559
A1,256,4,4629304576,3.703443,.55,56339207912,128484052
A1,256,8,4197786880,3.358229,.60,59176360778,140351600
A1,512,1,8611831296,6.889465,.59,54722795094,188306889
A1,512,2,8838422528,7.070738,.57,57534630723,209048371
A1,512,4,8635673600,6.908538,.59,53961824690,209142397
A1,512,8,6880010240,5.504008,.74,46800779195,191415455
A1,1024,1,12697316352,10.157853,.80,43922989716,270244453
A1,1024,2,16620184576,13.296147,.61,56058193882,351706612
A1,1024,4,12043047936,9.634438,.85,55996974313,354640155
A1,1024,8,16746586112,13.397268,.61,58478227176,378646535
A1,4096,1,36948582400,29.558865,1.10,52351049194,751539944
A1,4096,2,35120660480,28.096528,1.16,50715275580,731604217
A1,4096,4,33942425600,27.153940,1.20,49306225387,710108804
A1,4096,8,16923201536,13.538561,2.42,39764280921,608196890
A2,256,1,3610857984,2.888686,.70,64132879703,92039832
A2,256,2,3515878912,2.812703,.72,60508699597,110481373
A2,256,4,3047311616,2.437849,.84,52847266785,101660551
A2,256,8,2657866496,2.126293,.96,46377779776,108347660
A2,512,1,4318181888,3.454545,1.18,40376534597,115816097
A2,512,2,4419900928,3.535920,1.15,44140843816,126488267
A2,512,4,4662638592,3.730110,1.09,45473388121,143792358
A2,512,8,4581500928,3.665200,1.11,45598068841,148364389
A2,1024,1,8497683456,6.798146,1.20,44030175328,192135437
A2,1024,2,8770923520,7.016738,1.16,45534154369,207058741
A2,1024,4,8604532736,6.883626,1.19,44205346566,217696317
```

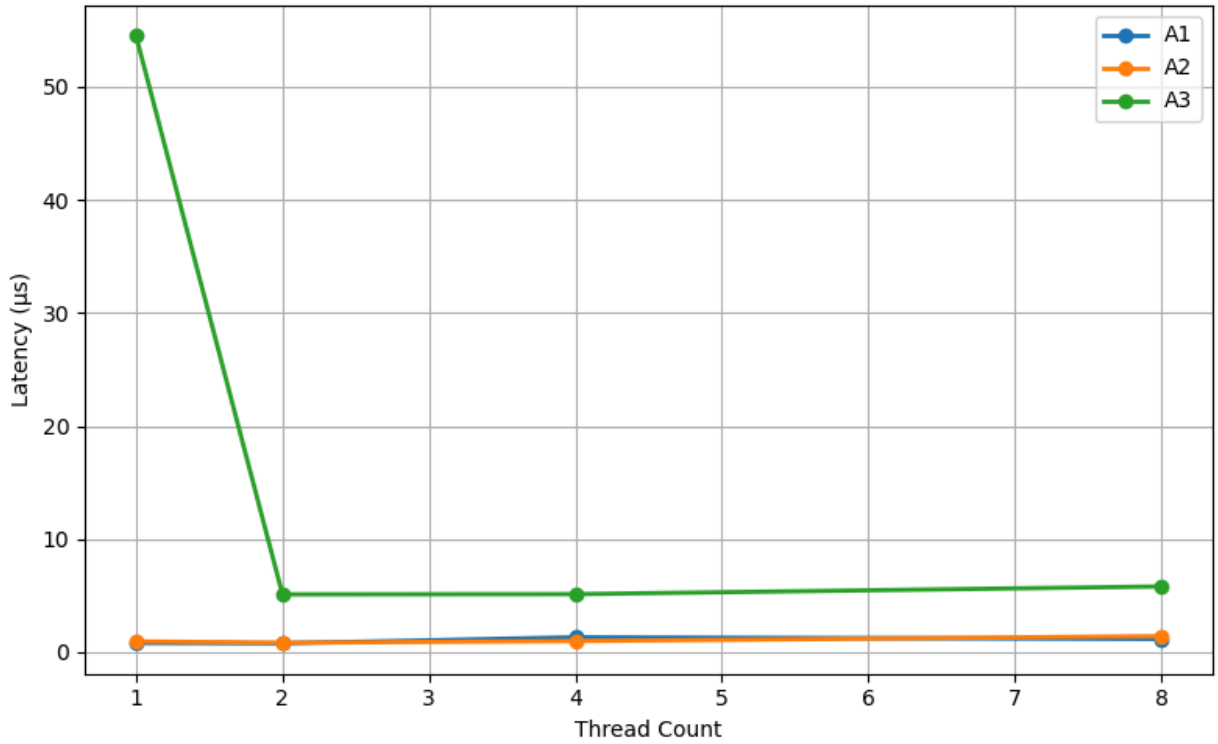


Plots:

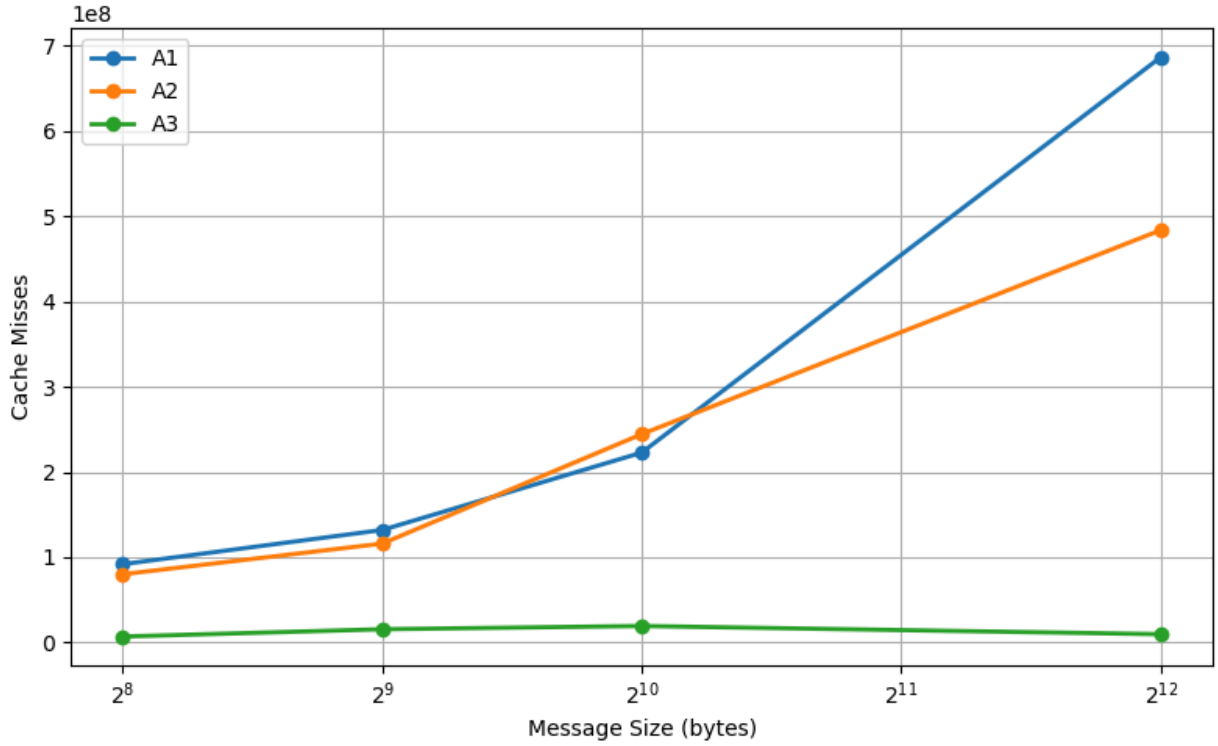
CPU Cycles per Byte



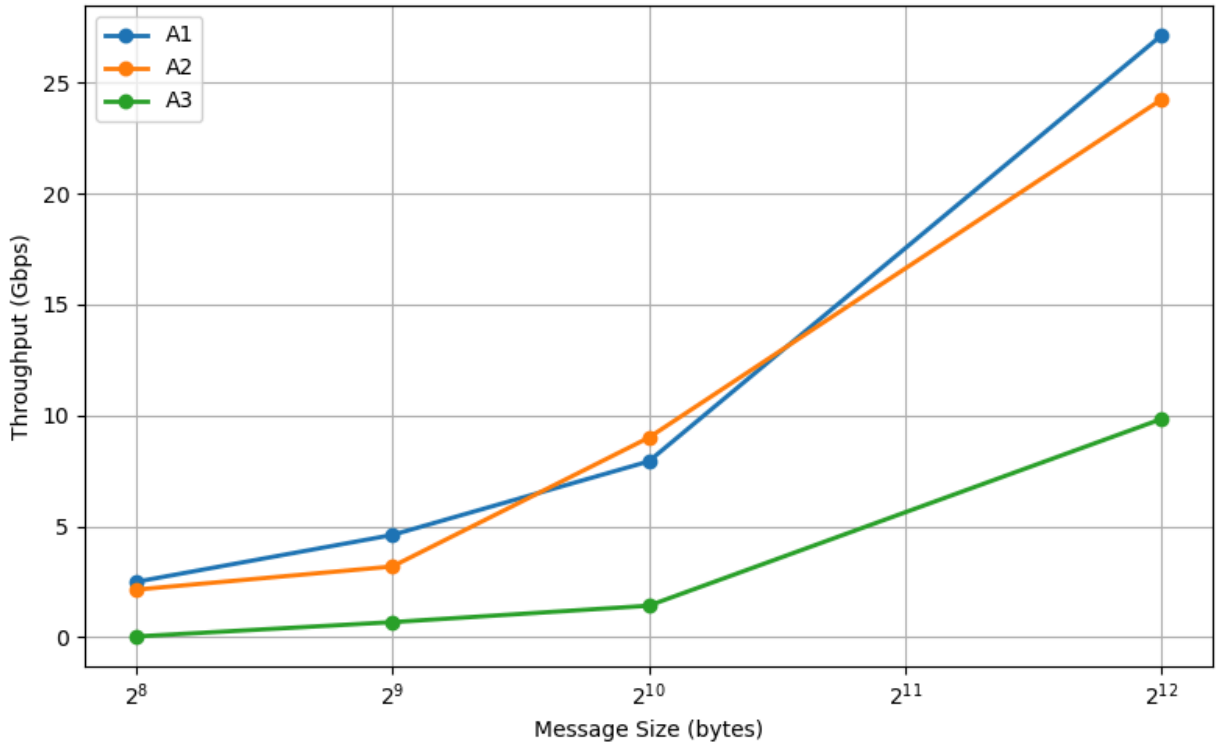
Latency vs Thread Count



Cache Misses vs Message Size



Throughput vs Message Size



Observations

Throughput:

- A2 generally highest throughput
- A1 moderate
- A3 poor for small messages
- Large messages improve all implementations

Latency:

- Small messages → low latency
- More threads → contention → latency increase
- A3 shows higher startup overhead

CPU Cycles:

- A1 highest cycles
- A2 reduced cycles
- A3 lowest per byte

Cache Misses:

- One-copy shows fewer misses
- better cache locality
- reduced memory copying

Analysis Questions (Part E)

Why does zero-copy not always give the best throughput?
Zero copy has setup overhead,page pining,kernel bookkeeping.For small messages overhead is greater than benefit,so throughput drops.

Which cache level shows the most reduction and why?
The last level cache shows the most reduction because it has fewer memory copies,better locality,less DRAM traffic.

How does thread count interact with cache contention?
Most threads share caches that cause invalidations and increase context switches.

When does one-copy outperform two-copy?
From results,starting at 512 bytes ,A2 is consistently greater than A1.

When does zero-copy outperform two-copy?
Observed at 4096 bytes,larger transfer cost.

Unexpected result?
Observed 8 threads is sometimes slower than 4 threads,due to cache thrashing,CPU overused ,scheduler overhead.

AI Usage Declaration:

I hereby declare that the following components of this project were developed using artificial intelligence (Gemini 3 Flash) ,Chatgpt(free version),Claude ai in compliance with the assignment guidelines:

Shell Script Generation: AI was utilized to fully create the automation logic and code for MT25017_Part_C_shell.sh with a focus on capturing and summarizing real-time data.

Plot Generation: AI was used in the development of the full python script MT25017_Part_D_Plot.py that processed the CSV data and produced the combined performance graphs.

Makefile: AI was used in the development of the Makefile that compiles and generates executable files for all the files in the assignment folder.

Debugging: With AI support, implementation problems pertaining to memory/buffer safety, socket handling, multi-threaded server behaviour, and precise performance measurement using automation and perf scripts were fixed.

Client Code:AI was utilized to create the full code of client logic and code of file such as MT25017_PartA1_Client.c,MT25017_PartA2_Client.c,MT25017_PartA3.c.

Server Code:AI was utilized to create the full code and logic of servers file names as following
MT25017_PartA1_Server.c,MT25017_PartA2_Server.c,MT25017_PartA3_Server.c.

Prompts Summary:

- i have to do this assignment, i want to know first that whats measurement i have to record is it only server side or its both side and what will the csv file contain, analyse this pdf completely
- is this is correct, check it i think it is wrong
- i have this code for PartA1_Client, is this correct or i should replace it with your code

- while testing both terminal should be open in the project folder or globally
- while doing testing following these command on server side getting this usage output, why is it happening
- what is recommended according to the assignment guidelines generate code according to that
- why my server does not get shutdown, it is still running
- how to test part two is working fine or not
- fix these bugs and give me updated code for part2 server
- so now the code compiles successfully, now how i test the second part
- my server terminal is not shutting down from part1
- now give me the code for part3 server and client
- give me complete final full A3 file pasted clean
- give me the client code for a3 or it will be same as a1 and a2
- how should i check the third part
- invalid netns name error while running server, how to fix
- what is part b
- now generate a shell file for the assignment
- how to execute shell script
- shell script is working for a1 but it is stuck for a2, why
- check my a1 server code and tell me if it is correct or not
- give me updated code for a3 server
- check my a3 server code and tell me if it is correct or not
- generate a script for plotting
- check if my plot code is correct and fix mistakes
- update the plot code
- check my shell script for performance measurement, is it correct or not
- which side is perf measuring performance in this script
- give me updated shell script
- terminal output is messy, correct the script
- why does the script run twice automatically
- commands to run plot script
- modify plot script so it only generates graphs and not pdf
- what are the column names in the csv generated by the shell script
- rename columns cycles to cpu_cycles and threads to threads_count and update whole script
- is perf used in this script or not
- make python plot code compatible with updated csv
- which side measurement is captured using perf
- update python plot code according to new column names
- does this python code generate required four graphs correctly

- is there any way to store bytes in csv file using shell script
- update shell script to store bytes and update plot code accordingly
- assignment says values must be hardcoded in python, is my code following that
- write code that copies csv values into arrays then plots using matplotlib
- are the generated graphs correct
- hardcode my csv data into python file
- double check the python code with my csv
- every time i run script data changes, why
- answer part e analysis questions based on my data
- generate a readme file for this assignment
- generate report file according to assignment guidelines
- include screenshots, plots, ai usage declaration and github url in report
- explain two-copy, one-copy, zero-copy in one line each
- what images to include when running shell script
- give commands to run shell and plot in text format
- Asked how to compile and run A1, A2, A3 servers and clients
  • Asked how to test Part A3 inside Linux network namespaces
  • Asked how to execute shell scripts and fix script permission issues
  • Requested generation and debugging of the automated benchmark shell script
  • Asked why server/client processes were stuck or hanging during tests
  • Asked how to collect performance metrics using `perf stat`
  • Requested fixes for parsing throughput, latency, cycles, and cache misses into CSV
  • Asked to rename CSV columns and update scripts accordingly
  • Asked how to store additional metrics (e.g., total bytes) into CSV
  • Requested generation of Python plotting scripts using matplotlib
  • Asked to modify plotting code to remove PDF output and generate only PNG graphs
  • Asked to hardcode CSV data into Python arrays as required by assignment guidelines
  • Requested verification of correctness of throughput/latency/cache plots
  • Asked which side (server/client) the `perf` measurements apply to
  • Requested README file generation
  • Requested full report template and structure
  • Requested answers to analysis questions based on collected CSV results
  • Requested rephrasing of AI declaration and debugging statements
  • Asked for short conceptual explanations of two-copy, one-copy, and zero-copy mechanisms
  • Asked for commands to run scripts and generate plots