CS 301
IIT GANDHINAGAR

PARALLEL ZIP

Ajinkya Pawar   Jitender Kumar
Palak Purohit   Sagar Bisen

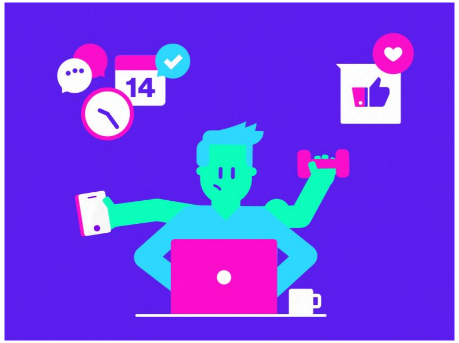
Instructor - Prof. Nipun Batra
Mentor - Rishiraj Adhikari

INDEX

MOTIVATION

- Why zip files parallely?
  - Multi-threaded programs have several advantages
    - Makes a single process faster
    - Threads don't require new address spaces
  - Consequence
    - Improved CPU performance
      - Reduced Turnaround Time

Already stressed out CPU



We try to give some relief

APPROACH

- Compression Algorithms
- Sequential zip
- Parallel Zip
- BETTER PARALLEL ZIP

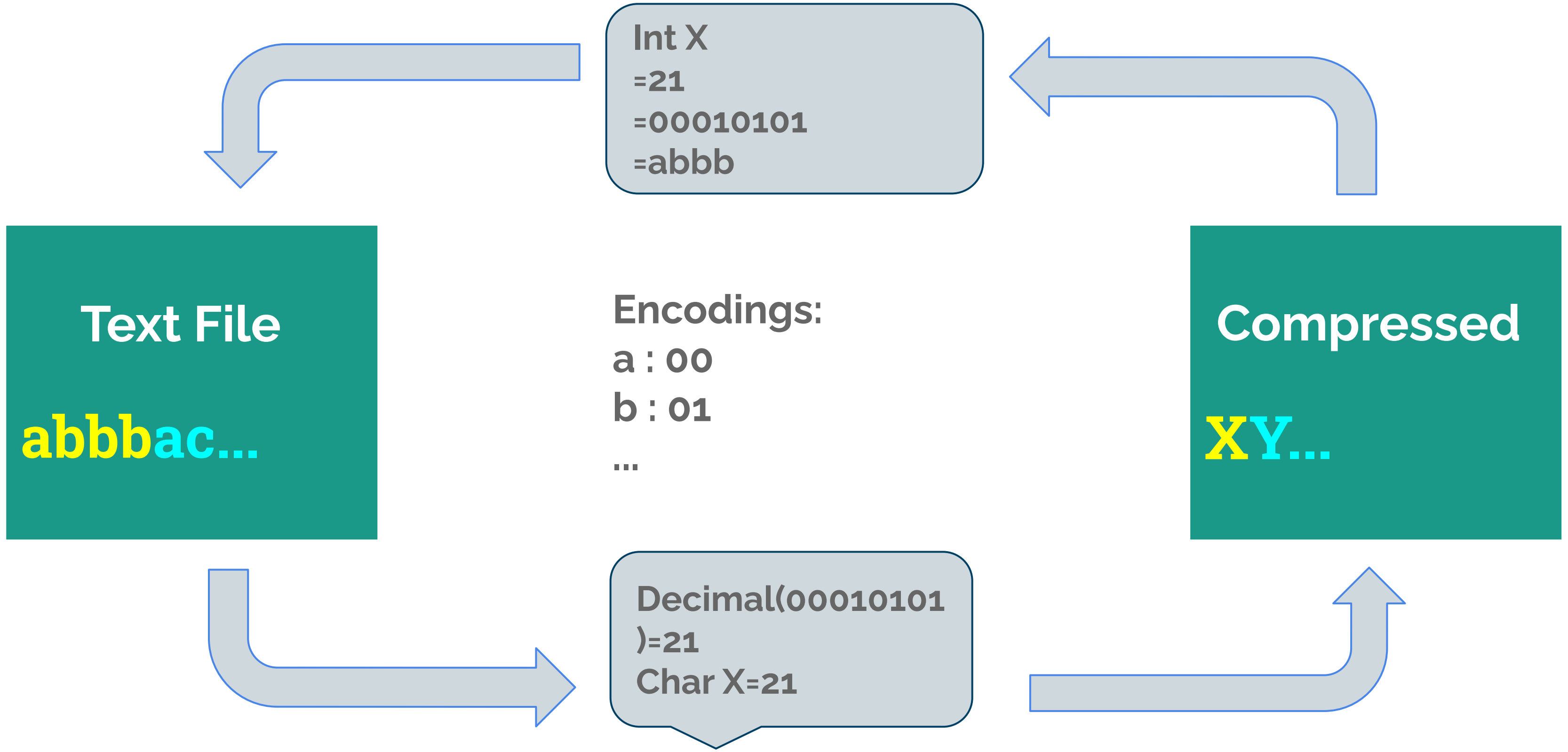
Well, keep an eye on what's attracting your eyes here

SOME THEORY

Two encoding schemes

- Huffman Encoding
- Run length encoding(RLE)

COMPRESSION - HUFFMAN CODING



RUN LENGTH ENCODING (RLE)

$$\begin{tikzcd}
 & \text{hhllo} \\
\text{hhllo} \arrow[ur] \arrow[dr] & \\
 & \text{2h2l1o}
\end{tikzcd}$$

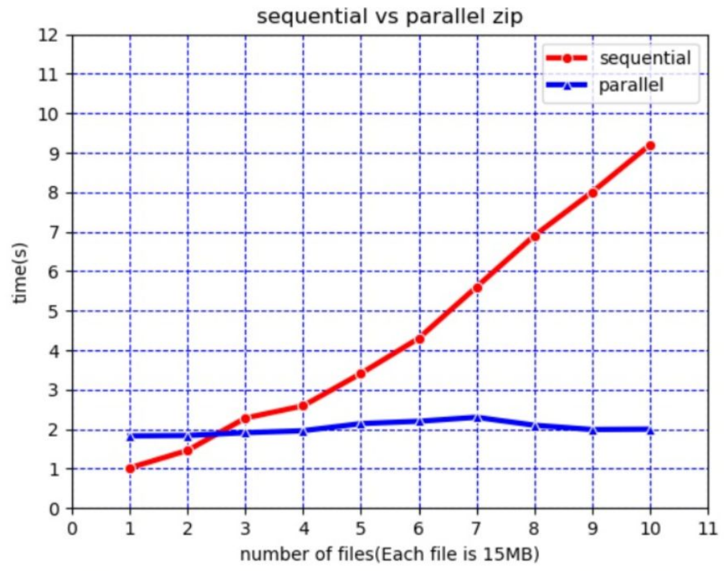
- For run length >2, encoding makes sense

$$\begin{tikzcd}
 & \text{wwwwbbbb} \\
\text{wwwwbbbb} \arrow[ur] \arrow[dr] & \\
 & \text{4w4b}
\end{tikzcd}$$

INITIAL MODELS

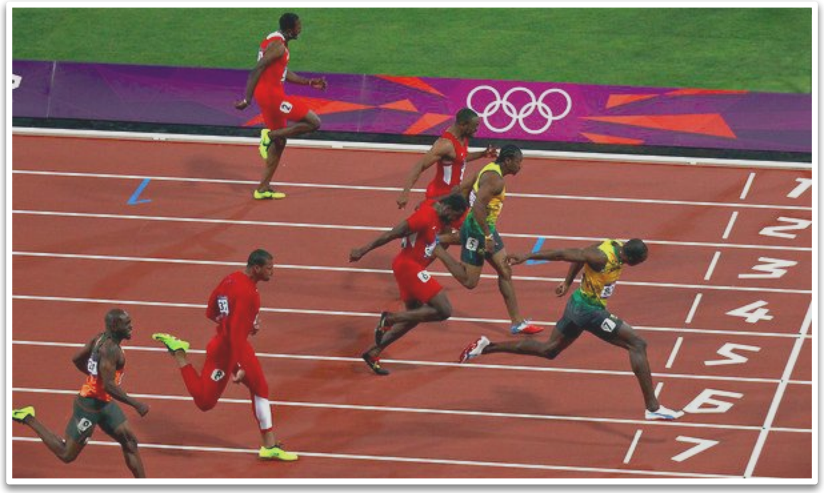
- SEQUENTIAL ZIP
  - Pseudo code
    - For i in range (no. of files)
      zip (ith file)

- PARALLEL ZIP
  - Pseudo code
    - Pthread_t threads[no. of files];
    - For i in range (no. of files)
      Pthread_create
      (ith thread, zip , ith file)



STRAGGLERS PROBLEM

- What are Stragglers ?
- Time
  - Sequential
    - File1 + File2 + …
  - Parallel
    - Max (core 1, core 2, …)+ thread overhead.
- Can sequential be faster than parallel !?



SOLUTION - PRODUCER & CONSUMERS

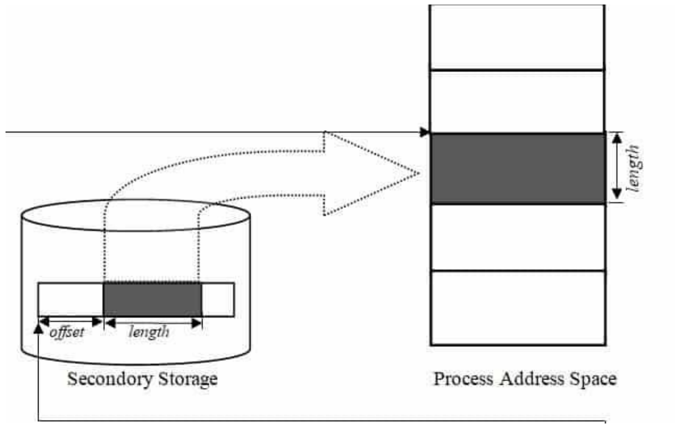
- Single producer
- Multiple consumers

- Producer:
  - Adds to the buffer
  - Each buffer object has
    - File number
    - Page number
    - Memory mapping
- Consumers:
  - Eat from the buffer
  - Each zipped object has
    - Compressed data
    - Data size

Read/Write System Calls vs MMAP

- Traditional Read
  - Process can't access disc
  - Entire data read and copied from target file into I/O buffer followed by process buffers

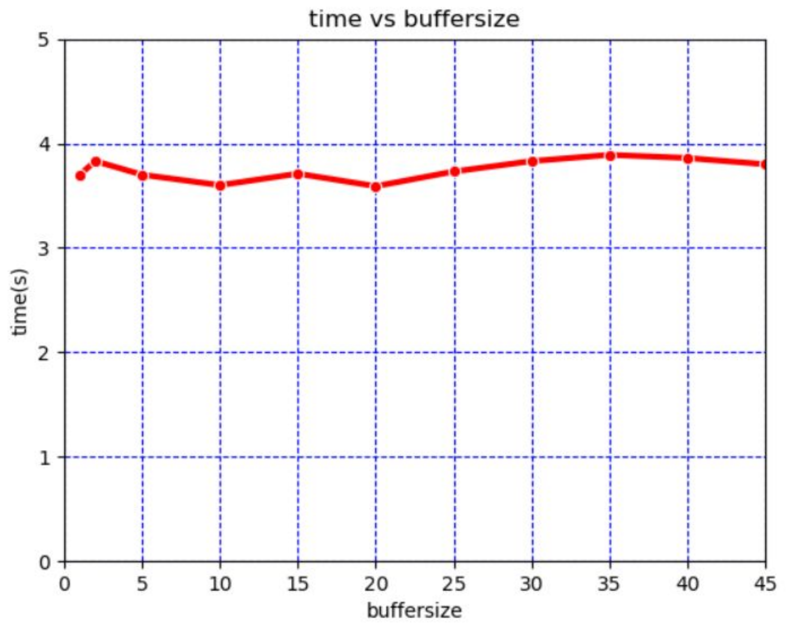
- MMAP
  - Loading relevant file sections to the RAM
  - File is loaded as pages into RAM

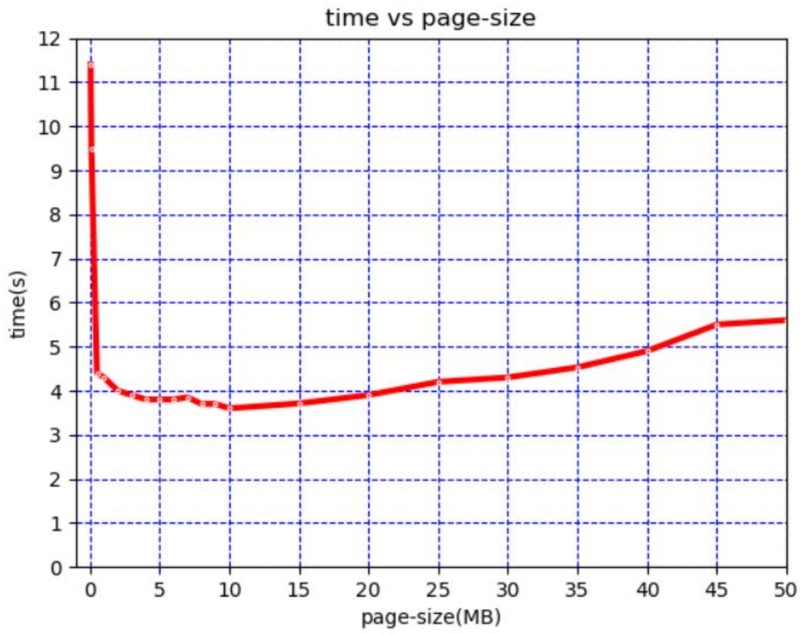


RESULTS

- ❖ Varying **BUFFER SIZE**
- ❖ Varying **PAGE SIZE**

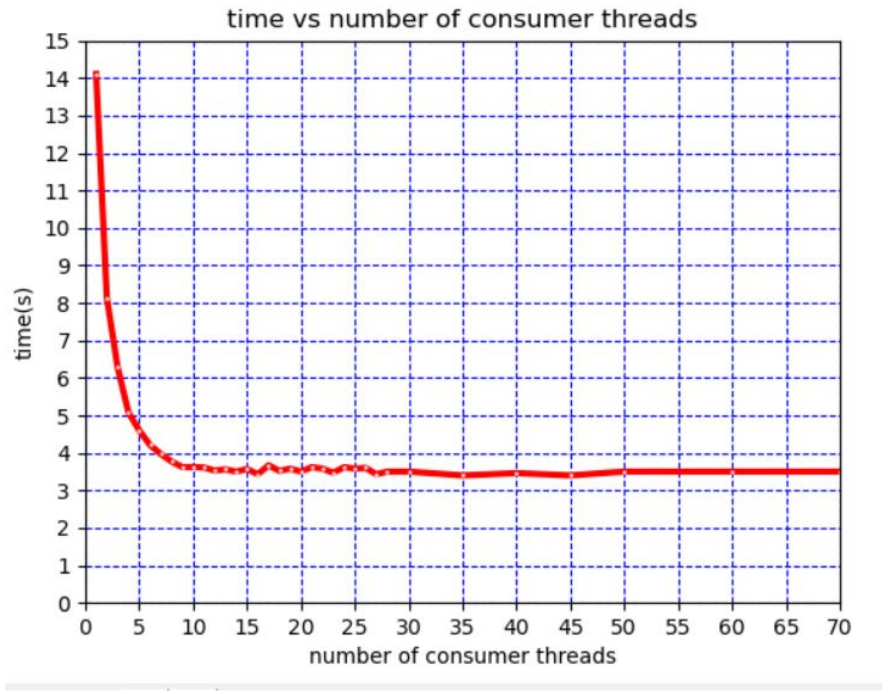
- Workload - 3 Files (100 MB, 300 MB, 400 MB)

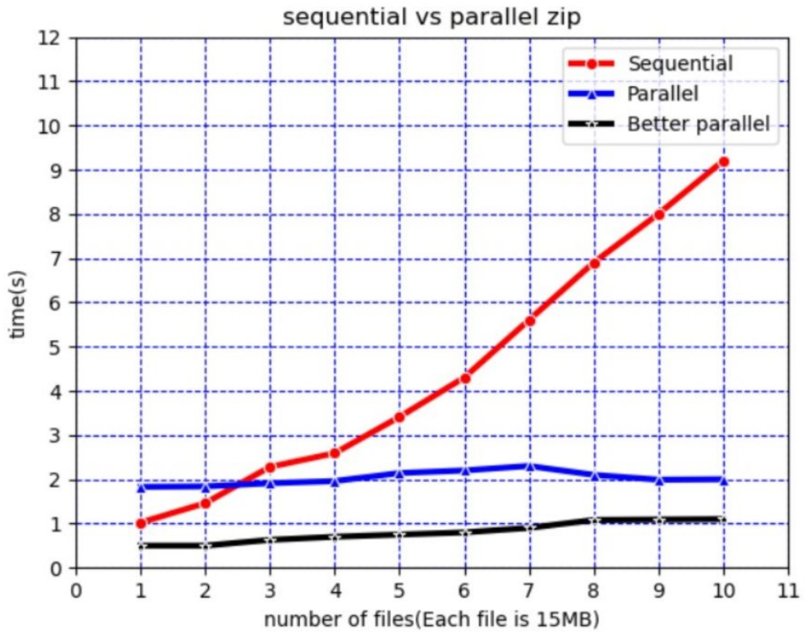




RESULTS

- Varying NUMBER OF CONSUMER THREADS



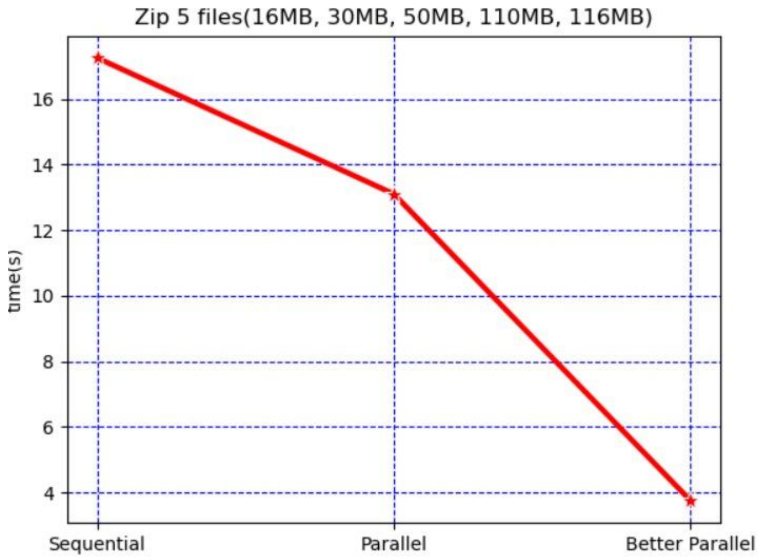


RESULTS

```
1  [|||||||||||||||||||||||||                         38.1%]
2  [|||||||||||||||||||||||||                         37.9%]
3  [|||||||||||||||||||||||||                         40.3%]
4  [|||||||||||||||||||||||||                         40.1%]
Mem[|||||||||||||||||||||||||||||||||||||||||6.10G/7.69G]
Swp[                                           270M/18.4G]
```

CPU Utilization of all 4 cores on running code

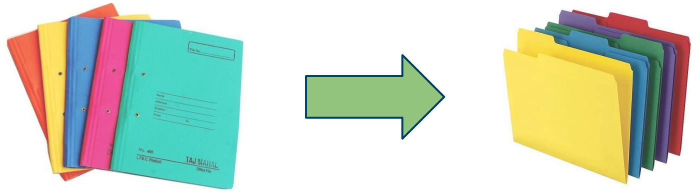
- ❖ Performance ∝ No. of cores



- ❖ Performance: Sequential<Parallel<<Better Parallel

LIMITATIONS AND SCOPE

- Scope



- Limitations
  - MMAP wastes space, wastage=(int*page_size - file_size)
  - Parallel Unzip

THANK YOU