MAPREDUCE WITH MULTI-THREADING

PREPARED BY:

HAMZA MEHMOOD (19I-0458)

ABIA JAVED (21I-0311)

ISRAA BATOOL (21I-0344)

OBJECTIVE & OVERVIEW

Objective:

- Implement a multithreaded MapReduce framework in C++ using pthreads.
- Count word frequencies from multiple input strings.

Overview:

Mapper Phase: Splits input strings into key-value pairs (word, count).

Shuffle Phase: Groups intermediate pairs by key (word).

Reducer Phase: Aggregates word counts and prints final results

KEY STRUCTURES & RESOURCES

KeyValuePair Structure:

- Fields:
  - word: Stores the word (max 50 characters).
  - counts: Array holding counts.
  - countIndex: Tracks count positions.

Shared Resources:

- keyValuePairs: Stores intermediate key-value pairs.
- pairCount: Tracks unique words.
- mutexLock: Ensures thread safety.

MULTITHREADING EXECUTION

Input Handling:

- Accepts multiple strings until the user types exit.
- Creates Mapper threads for each string using pthread_create.

Thread Management:

- Synchronization: Mutex ensures thread-safe access to shared resources.
- pthread_join: Ensures threads complete before proceeding to next phase.

PERFORMANCE & RESULTS

Performance:

- Parallel execution with pthreads enhances speed and scalability.
- Mutex prevents race conditions during shared resource access.

Results:

- Intermediate and final outputs successfully tested for multiple input cases.

CONCLUSION

- Multithreaded MapReduce efficiently counts word frequencies.
- Handles large datasets with improved execution speed.
- Potential optimizations: Advanced data structures like hash maps.

THANKYOU