Analysis Report for Project 2: Global Distributed Hash Table for Word Counting

COP5611 - Advanced Operating Systems

November 24, 2024

Overview

This report analyzes the implementation and performance of a global distributed hash table for word counting using InfiniBand RDMA technology. The program processes files of varying sizes using 8 parallel processes, storing and querying word occurrences in distributed hash tables.

Implementation Details

Key Features

1. Distributed Hash Table: Each process maintains a local hash table with 256 buckets, each bucket being 4MB in size. 2. RDMA Technology: - Read Operations: Remote data is fetched using RDMA read operations, enabling efficient querying without host intervention. - Memory Registration: Local hash tables are registered to allow remote RDMA reads. 3. Word Hashing: - Words are converted to uppercase for case-insensitivity. - Hash function: Hash(w) = $\sum_{i=1}^{n} \left(W_i \cdot 11^{(i+2)\%2} \right) \%2048$. - Top 3 bits designate the process, and the lower 8 bits specify the bucket.

Performance Evaluation

The program was tested on files of varying sizes (1MB, 4MB, 64MB, 256MB, 512MB), each executed three times. The table below summarizes the average execution times:

Scalability Analysis

Observation 1: Linear Growth in Processing Time The execution time of the program increases almost linearly as the size of the input file grows. This trend is expected because the system is designed to process a proportional amount of data per process, with each process handling a portion of the total

Table 1: Execution Times for Varying File Sizes

File Size	Average Time (s)
1MB	3.665
4MB	4.720
64MB	12.406
256MB	37.630
512MB	71.000

workload. The hash table's fixed bucket size ensures predictable performance, but as the file size increases, the sheer volume of data requires more computation and memory access, leading to longer processing times.

Observation 2: Efficiency of RDMA Technology The use of Remote Direct Memory Access (RDMA) significantly enhances scalability by offloading data transfer tasks from the host CPU. By enabling remote reads directly from another process's memory, RDMA eliminates the need for intermediate buffers or redundant copy operations. This direct access reduces communication overhead and allows the system to maintain high efficiency, even when querying large datasets. The low latency and high throughput of RDMA make it particularly well-suited for distributed systems like this one.

Observation 3: Challenges with Memory and Computation Overhead As file sizes grow, so do the demands on memory and computational resources. Larger datasets increase the likelihood of bucket overflows within the hash table, which can add complexity to record management. Atomic operations, while essential for maintaining consistency in a distributed environment, introduce slight delays when multiple processes access the same buckets. These factors, while manageable, highlight potential areas for optimization in handling large-scale data.

Conclusion

This project demonstrates the successful implementation of a scalable global distributed hash table for word counting using advanced RDMA technology. The use of RDMA substantially improves query performance by minimizing communication overhead and allowing efficient access to remote data. The program's performance exhibits clear scalability trends, with processing times increasing predictably as input file sizes grow. This validates the design's effectiveness in distributing workload across processes and leveraging RDMA to maintain high performance.