Analysis of Lock-Free Scalable Vector Performance in High-Concurrency Environments

ByeongKyu Park

February 26, 2024

1 Introduction

This document presents an analysis of a Lock-Free Sorted Vector (LFSV) designed for high-concurrency environments. Unlike traditional approaches that may rely on lock-based synchronization mechanisms, the LFSV leverages atomic operations to ensure thread safety and high performance. A key aspect of our implementation is the **Garbage Remover** component, which provides an efficient memory management strategy by deferring the deletion of vector elements.

2 Implementation

The LFSV is implemented with a focus on the Garbage Remover component for efficient memory management in a concurrent setting:

- **Garbage Remover:** Safely deletes vectors after a delay, preventing use-after-free errors and reducing dynamic allocation overhead, thus minimizing lock contention.

This approach allows for safe updates and access by multiple threads without serializing access, leveraging atomic operations for consistency.

3 Performance Evaluation

Performance was evaluated under different thread counts, with the following updated results observed:

4 Analysis

The updated results indicate a trend of improved performance with an increase in threads, up to a certain point. This highlights the efficiency of the LFSV's lock-free design and the effectiveness of the Garbage Remover in managing concurrent operations.

Number of Threads	Operations per Thread	Execution Time (seconds)
1	21000	0.498376
2	10500	0.439024
3	7000	0.44203
4	5250	0.391371
5	4200	0.379913
8	2625	0.290448
16	1313	0.333443

Table 1: Updated LFSV Performance Results

5 Conclusion

The LFSV, with its innovative use of the Garbage Remover for memory management, demonstrates significant potential for enhancing performance in concurrent applications. This lock-free data structure improves scalability and efficiency, making it an excellent choice for modern software development in high-concurrency environments. Future work will explore further optimizations and the impact of various hardware configurations on performance.