Implementing linearizable range queries for non-blocking data structures

Callum Robertson
Dr. Vincent Gramoli
School of Information Technologies
FACULTY OF ENGINEERING & INFORMATION TECHNOLOGIES

INTRODUCTION

In-Memory Databases

- In-memory databases remove the need for disk-based storage in normal read and write operations, resulting in faster and more predictable performance results.
- When disk storage is removed from database operations, the performance bottleneck imposed by concurrency control becomes more apparent.
- As the use of locks and mutual exclusion tend to serialize operations, current concurrency control mechanisms do not scale well when there are many threads of execution. As a result, recent research has been focused on developing less restrictive alternatives to locking mechanisms.

Non-Blocking Data Structures

- The use of locks involves a trade-off between coarse-grained locking, which can significantly reduce opportunities for parallelism, and fine-grained locking, which increases locking overhead.
- Non-blocking data structures provide concurrent access to resources without the need for locks and mutual exclusion, allowing for highly concurrent systems without a large performance overhead.
- A non-blocking algorithm is more complicated to develop than an equivalent system using locks, but provides guaranteed forward progression. This ensures that a system-wide deadlock cannot occur.

The Skip List

- A skip list is a data structure that operates in similar manner to a linked list, but with index nodes that allow searches to skip over blocks of elements, resulting in logarithmic query times.
- Numerous research efforts have been aimed at implementing non-blocking implementations of the skip list, as non-blocking skip lists are simpler to implement than other logarithmic data structures such as B-trees.

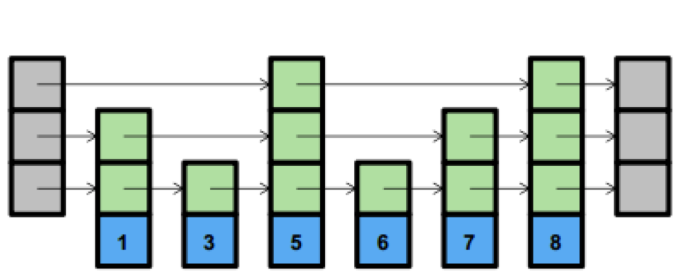

Figure 1. A Skip List

Range Queries

- A range query is a common read operation in database systems which retrieves all records with a value between a given lower and upper boundary.
- In order to provide a consistent view of the data structure, range query algorithms need to provide a guarantee of atomicity that is difficult to achieve without the use of mutual exclusion.

THE ROTATING SKIP LIST

- The Non Hot-Spot Skip List [1] proposed by Crain, Gramoli and Raynal is one of the latest research efforts to provide a scalable non blocking skip list.
- This data structure avoids contention hotspots by delegating index-level modifications to a background thread, so that all modifications made during the insert and delete operations performed by normal threads only affect the lowest levels of the structure.
- Ian Dick's optimized C implementation, The Rotating Skip List [2], improved the performance of this data structure through the use of memory locality and cache friendliness.
- The Rotating Skip List is a state-of-the-art non-blocking skip list which scales extremely well under high contention, but is of limited use in database systems due to the lack of range query operations in its interface.

THE CONTRIBUTION

- The contribution is to showcase the performance benefits of contention-friendliness by extending the interface of the Rotating Skip List with a non-blocking range query implementation that performs well under high contention.
- Sets of previous values were added to each nodes in the data structure, which allow the range query operation to collect a snapshot of the structure as it was at a particular time without blocking concurrent write operations.
- Write operations and range queries do not need to compete for the same resources as they will access different versions of the same node, which allows these operations to have a wait-free parallel execution.
- The performance overhead introduced to write operations is minimal, as values are only recorded when there is concurrent range query that may need to access them.

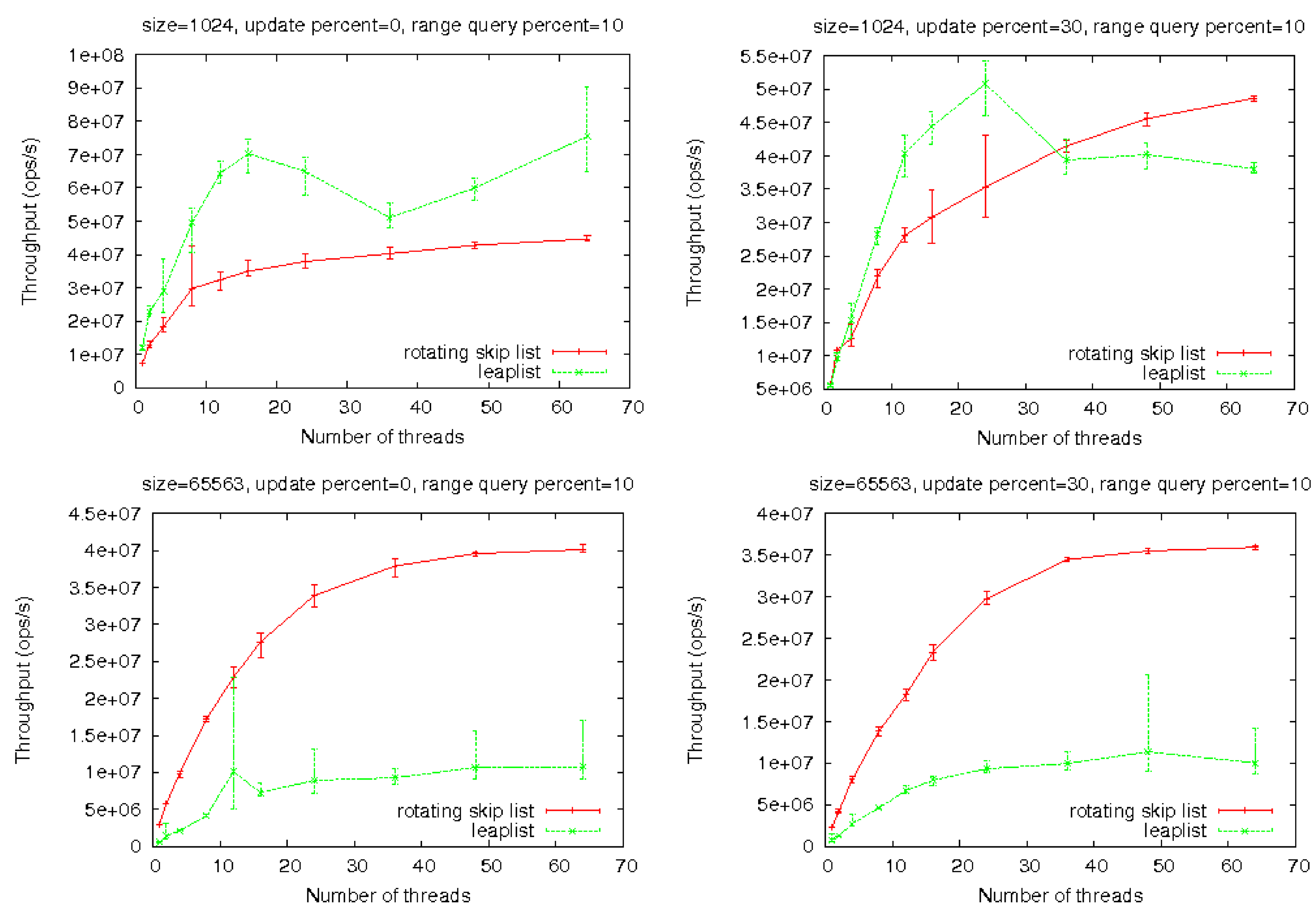

Figure 2. Performance experiments for the Rotating Skip List and LeapList

THE EXPERIMENT

- The modified Rotating Skip List, implemented in C, was benchmarked against the LeapList [3] data structure using a micro benchmark suite.
- Transaction throughput was documented in various experiments, where higher throughput equals higher performance
- Different range query and update frequencies, and data structure sizes, were manipulated to see how well the two lists perform under different levels of contention.

THE RESULTS

- As the percentage of update operations increases, we see the effects of contention on the two lists (Figure 2).
- The Rotating Skip List scales better with the number of threads than the LeapList when many threads performing concurrent update and range query operations.

FUTURE WORK

- The background thread's traversal of the data structure, and the traversals of range query operations, can be used to cache information that will minimize the work done by future operations.
- In order for the Rotating Skip List to be useful in the context of database systems, its interface will also need to be extended with a linearizable range update operation.

REFERENCES

1. T. Crain, V. Gramoli, M. Raynal. "No Hot Spot Non-blocking Skip List", IEEE 33rd International Conference on Distributed Computing Systems (ICDCS), pp.196-205, 2013
2. I. Dick. "The Rotating Skip List: A New Non-Blocking Skip List", Honours thesis, University of Sydney, 2013
3. H. Avni, N. Shavit, A. Suissa, "Leaplist: Lessons Learned in Designing TM-Supported Range Queries", PODC, p. 299-308, 2013