Truly scalable data structures

Based on relaxations, Distributed Queues (DQs) outperform (and scale) similar lock-free data structures by load-balancing contention among different cores. Out-of-order bounds are controlled by configurable load-balancing strategies. k-FIFO queues go one step further and provide even configurable out-of-order behavior at the expense of performance. (Figure 1)

Scalloc: A scalable allocator

Thread-local buffers for fast local access combined with globally available DQs to reclaim objects form an allocator that performs and scales well even in workloads where objects are shared among threads. Furthermore DQs also enable work-stealing on objects and buffers. (Figure 2)

Results

Suppose you have a FIFO queue and an enqueue sequence of

\[ \text{enqueue}(a) \rightarrow \text{enqueue}(b) \rightarrow \text{enqueue}(c) \]

Then in a relaxed setting, dequeuing elements could allow reorderings, resulting in sequences, such as

\[ \text{dequeue}(a) \rightarrow \text{dequeue}(b) \rightarrow \text{dequeue}(c) \quad \text{of} \quad \text{dequeue}(b) \rightarrow \text{dequeue}(a) \rightarrow \text{dequeue}(c) \]

The resulting freedom to choose some element can be used to avoid synchronization to some extend, effectively increasing the possible parallelism of such concurrent objects.