Abstract

The trend in modern hardware development has shifted away from enhancing serial hardware performance, towards multi-core processing. This trend forces programmers and algorithm designers to think about ways in which to make their code and applications parallel in nature. Concurrent algorithms tackle the problem of sharing data and keeping that data coherent while multiple actors simultaneously attempt to change or access the data. For concurrent data structures and algorithms to perform optimally on modern processors they must generally have two properties. First, they must use the processor’s cache memory efficiently for both data and instruction. Today’s processors are very sensitive to memory access patterns and contention induced by concurrency in cache coherence protocols. As such, algorithm designers must take special care to accommodate these particulars. Second, the algorithms must make reading the data structure as cheap as possible. These reads are generally known as invisible reads and other threads cannot see their presence in a data-structure. As such, correctly recycling of memory becomes an enormous issue.

Recycling memory in concurrent data-structures is known as the “memory reclamation problem” and is considered the one of the largest open problems in the concurrency field. Since concurrent threads cannot know which thread is looking at a piece of memory they must communicate view some other means. These algorithms must be designed carefully so that they preserve the original performance and the theoretical progress of the data-structure. This talk gives a brief introduction into the memory reclamation problem, some existing algorithms/solutions, and finally a new solution HazardScan.