5 Exercise #5: Non-Blocking Synchronization

This exercise is about non-blocking synchronization. You will implement certain non-blocking data structures, compare the performance of a non-blocking queue to its locked variants and learn about problems and benefits of non-blocking synchronization.

5.1 Non-Blocking Queue

Read Michael and Scott's paper[1] about their blocking and non-blocking queue algorithms. Implement their non-blocking algorithm and use double-word compare-and-swap operations on pointer and counter pairs to make the ABA problem very unlikely.
To make the queue compatible with the provided `SimpleQueue`, do not copy the contents of the next data value (line D12 in the paper), but just dequeue the first item, then check if it is the dummy item and when it is: enqueue it again and start the dequeue operation from the beginning.
Can you think of any problems when the queued items are allocated from the heap and are frequently allocated and deallocated together with other data? Have you ideas how to solve this problem?

5.2 Wait-Free Buffers

Design and implement a bounded buffer and a list-based (unbounded) buffer for one producer and one consumer thread. Both threads should be able to perform their operation concurrently. Your implementation should be wait-free in the sense that the number of instructions each operation takes should be bounded by a constant number.
The bounded-buffer should indicate whether the buffer is full, or respectively empty, and perform no operation on the buffer.
Explain how your buffers work and argue why they work correctly in the case of one concurrent producer and consumer. In what situations can you use this kind of buffer?

5.3 Test and Measure

Test your implementations and compare the performance of your non-blocking queue with the blocking variants that you already have. Measure high contention cases as well as the uncontended case. Are there any benefits with regard to performance? If not, can you think of any other benefits?

5.4 Conclusion

What conclusions do you personally draw with regards to non-blocking synchronization vs. lock-based synchronization? Do you see use for it in runtime and operating systems?

5.5 Submit

Submit your solutions by creating the directory /proj/i4cs/students/`your_login`/assignment5/ All files in this folder will be collected after the submission deadline. The file `comments.txt` will be created in this directory and contains comments from the tutors. Please create a file `group.txt` with your and your partner's login if you do your assignments in a group of 2 people.

Remarks:

- GCC will only produce a 16 byte compare-and-swap instruction when the -mcx16 compiler flag is given. Use `__sync_bool_compare_and_swap()` with the `__int128` data type on 64 bit systems.
- The memory reference which is target to the double-word compare-and-swap must be 16 byte aligned.

Submit until: 2015-01-27

[1] Maged M. Michael and Michael L. Scott "Simple, fast, and practical non-blocking and blocking concurrent queue algorithms"