COP 4520 Spring 2020

Project Topics

I. Regular Data Structure Topics

Team 1: Concurrent Hash Maps (Lock-free and wait-free)
"A Wait-Free Hash Map"
https://link.springer.com/article/10.1007/s10766-015-0376-3

Team 2: Concurrent Ring Buffers (Lock-free and wait-free)
"A wait-free multi-producer multi-consumer ring buffer"
https://dl.acm.org/citation.cfm?id=2835260.2835264

Team 3: Concurrent Priority Queues (Lock-free and wait-free)
"A Lock-Free Priority Queue Design Based on Multi-Dimensional Linked Lists"
https://ieeexplore.ieee.org/document/7079379

Team 4: Concurrent Vectors (Lock-free and wait-free)
"An Efficient Wait-Free Vector"
https://ieeexplore.ieee.org/document/7073592

Team 5: Lock-free transactional vector" (Lock-free and wait-free)
"Lock-free transactional vector"
No link, please contact Kenneth Lamar (our GTA) to get a copy of his PMAM 2020 paper.

Team 6: Concurrent Dictionaries (Lock-free and wait-free)
"An Efficient Lock-Free Logarithmic Search Data Structure Based on Multi-dimensional List"
https://ieeexplore.ieee.org/document/7536527

Team 7: Concurrent Adjacency Lists (Lock-free and wait-free)
"Lock-Free Transactional Adjacency List"
https://arxiv.org/abs/1903.10036

Team 8: Concurrent Transactional Data Structures (both blocking and non-blocking)
"Lock-free Transactions without Rollbacks for Linked Data Structures"
https://dl.acm.org/citation.cfm?id=2935780
Note: implement the transactional linked-list

Team 9: Concurrent Vectors, ABA prevention (Lock-free and wait-free)
"Scalable Nonblocking Concurrent Objects for Mission Critical Code"
https://dl.acm.org/citation.cfm?id=1639954
Note: implement the lock-free vector with the ABA prevention scheme

Team 10: Concurrent Transactional Data Structures, Optimization strategies (both blocking and non-blocking)
"Check-Wait-Pounce: Increasing Transactional Data Structure Throughput by Delaying Transactions"
https://link.springer.com/chapter/10.1007/978-3-030-22496-7_2
Note: implement the transactional linked list with Check-Wait-Pounce

Team 11: Concurrent Transactional Data Structures (both blocking and non-blocking)
"Wait-free Dynamic Transactions for Linked Data Structures"
https://dl.acm.org/citation.cfm?id=3309491
Note: implement wait-free transactional linked-list

Team 12: Concurrent queues with relaxed semantics (non-blocking and non-linearizable)
"Quantifiability: Concurrent Correctness from First Principles"
https://arxiv.org/abs/1905.06421
Note: implement the quantifiable queue

Team 13: Concurrent merkle trees
No link, please contact Ryan Dozier (ryan.dozier@knights.ucf.edu) for a link to a paper

Team 14: MCAS algorithms
"A Wait-Free Multi-Word Compare-and-Swap Operation"
https://link.springer.com/article/10.1007/s10766-014-0308-7
Note: apply MCAS to implement a linked-list-based set (see textbook Chapter 9)

Team 15: A persistent lock-free queue for non-volatile memory (persistent queues)
"A persistent lock-free queue for non-volatile memory"
https://dl.acm.org/doi/10.1145/3200691.3178490
Note: You do not need NVM to run this queue

Team 16: Concurrent Priority Queues
"A Practical, Scalable, Relaxed Priority Queue."
https://dl.acm.org/doi/10.1145/3337821.3337911

Team 17: Persistent BSTs
"Persistent Non-Blocking Binary Search Trees Supporting Wait-Free Range Queries"
https://dl.acm.org/doi/abs/10.1145/3323165.3323197
Note: You do not need NVM to run this BST

Team 18: Durable/persistent sets
"Efficient lock-free durable sets"
https://dl.acm.org/doi/abs/10.1145/3360554
Note: You do not need NVM to run this set

Team 19: Wait-free queues
“A wait-free queue as fast as fetch-and-add”
https://dl.acm.org/doi/abs/10.1145/2851141.2851168

Team 20: Lock-free Search Trees
“Lock-free Contention Adapting Search Trees”
https://dl.acm.org/doi/abs/10.1145/3210377.3210413

Team 21: Lock-free hash tables
“A scalable lock-free hash table with open addressing”
https://dl.acm.org/doi/abs/10.1145/3016078.2851196

Team 22: Concurrent Double-Ended Queues
“An Unbounded Nonblocking Double-Ended Queue”
https://ieeexplore.ieee.org/document/7573821

Team 24: Lock-free Queues
“BQ: A Lock-Free Queue with Batching”
https://dl.acm.org/doi/abs/10.1145/3210377.3210388

Team 25: Lock-free and wait-free stacks
“FA-Stack: A Fast Array-Based Stack with Wait-Free Progress Guarantee”
https://ieeexplore.ieee.org/abstract/document/8097018

II. Special Topics

Team 23

These projects are more research-oriented and the work on a special topics project will require changes to the listed requirements in Project Assignments 1 and 2. If you are scheduled to work on a special topics project, please plan to discuss the plan of work and the changes needed to the project assignments this with Dr. Dechev.