CS586 – Distributed Computing
Fall semester - Ac. Year 2012-13
Instructor: Panagiota Fatourou

Theory Projects

Deadline for the survey: January 21, 2013

Projects

Project 1 (Universal Constructions)

1. J. Anderson and M. Moir, "Universal Constructions f or Large Objects", *IEEE Transactions on Parallel and Distributed Systems,* 10(12):1317-1332, December 1999 (contact the instructor) .
2. P. Fatourou and N. Kallimanis, "The Red-Blue Adapti ve Universal Construction", *Proceedings of the International Symposium on Distributed Computing (DISC),* 2009 (contact the instructor).

Project 2 (Concurrent Data Structures I)

1. M. Michael. "High Performance Dynamic Lock-Free Has h Tables and List-Based Sets", In *Proceedings of the 14th annual ACM Symposium on Parallel algorithms and architectures (SPAA)*, pages 73-82, August 2002 (http://www.research.ibm.com/people/m/michael/spaa-2002.pdf).
2. M. Fomitchev and E. Ruppert. "Lock-Free Linked List s and Skip Lists", In *Proceedings of the 23rd annual ACM symposium on Principles of distributed computing (PODC)*, pages 50-59, July 2004 (http://www.cse.yorku.ca/~ruppert/papers/lfll.pdf).

Project 3 (P2P Systems I)

1. Aspnes and Shah, "Skip Graphs", SODA 2003.
2. Aspnes, Kirch and Krishnamurthy, "Load Balancing an d Locality in Range-Queriable Data Structures", PODC 2004.

Project 4 (Grid Computing)

1. "Task Scheduling Strategies for Workflow-based Appl ications in Grids", Jim Blythe, Sonal Jain, Ewa Deelman, Yolanda Gil, Karan Vahi, Anirban Mandal, Ken Kennedy.
2. "Adaptive Workflow Processing and Execution in Pega sus", Kevin Lee, Norman W. Paton, Rizos Sakellariou, Ewa Deelman, Alvaro A. A. Fernandes, Gaurang Mehta

Project 5 (Atomic Snapshots)

1. Attiya and Rachman, “Atomic Snapshots in O(nlogn) o perations”, SIAM J. on Computing, 27(2): 319-340, 1998 (http://epubs.siam.org/SICOMP/volume-27/art_27946.html, until page 328, not up until the end).
2. Israeli, Shaham and Shirazi, “Linear-Time Snapshot Implementations in Unbalanced Systems”, Mathematical Systems Theory, 1995. (hard copy in my office, up until page 479).

Project 6 (Counting Networks)

1. James Aspnes, Maurice Herlihy, and Nir Shavit. Counting networks. *Journal of the Association for Computing Machinery* 41(5):1020–1048, September 1994.
2. Herlihy, Shavit and Waarts, “Linearizable Counting Networks”, Distributed Computing, 1995. (http://citeseer.ist.psu.edu/herlihy91linearizable.html)

Project 7 (Adaptive Mutual Exclusion)

1. Attiya & Bortnikov, "Adaptive and Efficient Mutual Exclusion", ACM Symposium on Principles of Distributed Computing, 2000. (http://www.springerlink.com/content/78c27mhk3cp04e9n/fulltext.pdf)
2. J. Anderson and Y. J. Kim, “Adaptive mutual exclusi on with local spinning”, DISC 2000.

Project 8 (Failure Detectors)

1. Tushar Deepak Chandra, Sam Toueg, “Unreliable Failu re Detectors for Reliable Distributed Systems”, Journal of the ACM, 1996. (http://portal.acm.org/citation.cfm?id=226647)
2. Tushar Deepak Chandra, Vassos Hadzilacos, Sam Toueg, “The Weakest Failure Detector for Solving Consensus”, 1994. (http://theory.lcs.mit.edu/classes/6.852/05/papers/jacm96.pdf)

Project 9 (Distributed Data Structures I)

1. M. Herlihy, S. Tirthapura and R. Wattenhofer, “Comp etitive Concurrent Distributed Queueing”, pp. 127-133, ACM PODC, 2001.
2. M. Demmer and M. Herlihy, “The arrow distributed Di rectory Protocol”, pp.119-133, DISC 1998.

Project 10 (Distributed Data Structures II)

1. N. Shavit and A. Zemach, “Scalable Concurrent Prior ity Queue Algorithms”, pp. 113-122, ACM PODC 1999.
2. Shavit & Zemach, “Diffracting Trees”, ACM Transacti ons on Computer Systems, 14(4): 385-428, 1996. (http://citeseer.ist.psu.edu/68066.html)

Project 11 (Fast & Adaptive Renaming Algorithms)

1. Attiya & Fouren, "Algorithms Adapting to Point Contention", Journal of the ACM, 50(4):444-468, 2003.
   (http://portal.acm.org/citation.cfm?id=792541&dl=ACM&coll=portal)

2. Afek, Attiya, Fouren, Stupp and Touitou, "Long-live d renaming made adaptive", Proceedings of the 18th Annual ACM Symposium on Principles of Distributed Computing, 1999. (http://citeseer.ist.psu.edu/5432.html)

Project 12 (Fast & Adaptive Collect with Applications)

1. Afek, Stupp and Touitou, "Log-lived Adaptive Collec t with Applications", FOCS 1999. (hard copy in my office)
2. Attiya, Fouren and Gafni, "An Adaptive Collect Algo rithm with Applications", Distributed Computing, 2002. (http://www.springerlink.com/media/bn42d3yhvm7jqvqugt33/contributions/h/h/m/1/hhm1cbpndex2570x.pdf)

Project 13 (Synchronous Atomic Snapshots)

1. Brodsky and Fich, "Efficient Synchronous Snapshots" , PODC 2004. (www.cs.utoronto.ca/~abrodsky/papers/p62-brodsky.pdf)
2. G. Neiger and R. Singh, "Space-Efficient atomic sna pshots in synchronous systems", TR GIT-CC-93-46, Georgia Institute of Technology, 1993. (http://etd.library.gatech.edu/dspace/bitstream/1853/6777/1/GIT-CC-93-46.pdf)

Project 14 (Measuring Contention)

1. Dwork, Herlihy and Waarts, "Contention in Shared Me mory Algorithms", Journal of the ACM, 44(6):779-805, November 1997. (http://citeseer.ist.psu.edu/dwork93contention.html)
2. Anderson and Yang, "Time Contention Trade-offs for multiprocessor synchronization", Information and Computation. http://citeseer.ist.psu.edu/anderson96timecontention.html

Project 15 (P2P Systems II)

1. Crainiceanu, Linga, Gehrke and Shanmugasundara, "Querying P2P Networks using P-trees", WebDB 2004.
2. Zhang, Kalnis, Chin Ooi, Tan, "Generalized Multi-Di mensional Data Mapping and Query Processing", ACM Trans. on Database Systems, 2005.

Project 16

1. Hagit Attiya, Arie Fouren, "Algorithms adapting to point contention", Journal of the ACM, 50(4): 444-468 (2003) (http://portal.acm.org/citation.cfm?id=792541)

Project 17

1. M. Michael. "High Performance Dynamic Lock-Free Hash Tables and List-Based Sets", In Proceedings of the 14th annual ACM Symposium on Parallel algorithms and architectures (SPAA), pages 73-82, August 2002 (http://www.research.ibm.com/people/m/michael/spaa-2002.pdf)
2. M. Fomitchev and E. Ruppert. "Lock-Free Linked Lists and Skip Lists", In Proceedings of the 23rd annual ACM symposium on Principles of distributed computing (PODC), pages 50-59, July 2004 (http://www.cse.yorku.ca/~ruppert/papers/lfll.pdf) (you should study only the first

part; skip list and amortized complexity analysis should not be studied).

Project 18

1. Rachid Guerraoui and Michal Kapalka, "On the correctness of transactional memory", Proceedings of the 13th ACM SIGPLAN Symposium on Principles and practice of parallel programming (PPoPP '08). (http://dl.acm.org/citation.cfm?id=1345233)
2. Damien Imbs and Michel Raynal, "Virtual world consistency: A condition for STM systems (with a versatile protocol with invisible read operations)", Theoretical Computer Science. 444 (July 2012), 113-127.

Project 19

1. Y. Afek, G. Stupp, and D. Touitou, "Long-lived Adaptive Collect with Applications", Proceedings of the 40th Annual Symposium on Foundations of Computer Science, 1999 (http://portal.acm.org/citation.cfm?id=796475)
2. M. Herlihy, V. Luchangco, and Mark Moir , "Space- and Time-adaptive Nonblocking Algorithms" (bhttp://labs.oracle.com/projects/scalable/Papers/CATS2003.pdf)

Project 20

1. F. Fich, V. Luchangco, M. Moir, and N. Shavit, "Obstruction-free algorithms can be practically wait-free", *Proceedings of the 18th International Symposium on Distributed Computing*, pp. 78-92, September 2005.
2. Gadi Taubenfeld: Efficient Transformations of Obstruction-Free Algorithms into Non-blocking Algorithms. DISC 2007: 450-464

Project 21

1. T. Anderson, "The performance of spin lock alternatives for shared-memory multiprocessors", IEEE Transactions on Parallel and Distributed Systems, 1(1):6–16, 1990.
2. James H. Anderson, Yong-Jik Kim: A generic local-spin fetch-and-phi-based mutual exclusion algorithm. J. Parallel Distrib. Comput. 67(5): 551-580 (2007)

Project Description

The goal of each theory project is to study two papers and write a report describing them. This report should be appropriately structured, based on the cohesion of the results presented in each paper, their relation, and the improvements among papers.

Paper study: You should understand in detail the papers you have undertaken. More specifically, you should:

- understand the algorithms and the techniques presented in these papers,
- easily answer to questions like: "Why is each of the algorithm's code lines necessary and what happens if we remove any of them?",
- invest time on the algorithms you study, produce your own bad execution examples,

and understand the way each algorithm handles them,

- study a large number of examples that describe the way each algorithm works and detect the most difficult scenarios each of them should handle,
- prepare a large number of your own examples (possibly in addition of those presented in the paper) that prove you have understood the algorithms of your papers in detail,
- insist analyzing the algorithms and understanding the proofs described for them in the papers.

Finally, in your report each algorithm should be presented together with its analysis and an intuitive description of its correctness and its complexity.

Report

Your report should not exceed 16 (closely written) pages. Extra pages will be evaluated negatively. In your report you should include the important (according to your opinion) results of the papers you have studied. Also, you should present several examples that prove you have understood in detail the algorithms.

Your report should have a form close to the following:

1. Introduction: Abstract description of the problem and its significance. Abstract description of the results presented later (approximately 1-2 pages). One paragraph explaining the way your report is organized.
2. Model: Presentation of the model (the system you work, definitions you need for the description, and analysis of the algorithms, etc.) (approximately 2 pages).
3. Technical part of the project: Here you should describe the algorithms, their correctness, and their analysis according the directions given above (approximately 12 pages).
4. Epilogue: Describe open problems (if any) (1/2 page).
5. References (1/2 page that can be additional to the 16 pages)

Your report will be evaluated based on the following criterions:

1. how much it is convincing that you have understood in detail the results presented in the papers studied,
2. how well organized it is,
3. the selection of results you have performed (e.g., it should not be only the easier results presented in the papers),
4. how much it is differentiated from a simple translation of the papers,
5. the number of your own examples you have studied and how good you explain the algorithms, the techniques, and the proofs you have included,
6. correctness of the intuitive descriptions you have added,
7. how well written it is (formalism and correctness),
8. at which extent the above goals are accomplished without violating the bound of 16 pages.