Paper Chase Assignment

For this assignment, I'd like you to get into the mindset of a reviewer, and to practice the skills of hunting down references to find clarifications and explanations. I also want you to start hunting for topics for your end-of-semester project.

- Find a research paper on data structures; let's call it Paper 1. Paper 1 can be something related to your own research that you'd like to examine more closely, or you can dive into a relevant conference/journal and pick something you find interesting. I've included a list of potential starting points at the end of this handout, but only as suggestions; you're welcome to use papers outside this list.

- Now imagine that you have been assigned Paper 1 as a reviewer. The first part of any paper review is a succinct summary (at most two short paragraphs) of the main contributions of the paper. In short, what is the paper about? **Write that summary.** Your summary should be objective; keep your *opinions* about the paper limited.. (An actual paper review would also include your opinions about the paper after the summary, but they will not be a part of this assignment.)

- **Identify and explain the major technical ingredients of the paper**, using up to one full page. In short, how are the paper's results obtained?

- Skim the rest of the paper to get a general idea of what is going on, and then read as much detail as you can. **Describe where you got lost**, using up to one full page. (This is *not* normally part of a review.) I'm specifically looking for something you found *confusing* or *overwhelming* or *unconvincing* in the core of the paper, not merely background results that you find *unfamiliar*. Include enough context that your point of confusion is clear to someone who is not familiar with the paper. If you can't figure out the major contributions of the paper, or if you never get lost, start over with a different Paper 1.

- Identify another paper that would help enhance your understanding of Paper 1; let's call this Paper 2. This could be a paper that Paper 1 cites, a paper that cites Paper 1, or just a result of your mad googling skills. Read Paper 2, first skimming to get a general idea of its content, and then trying to read as much detail as you can. **Briefly summarize paper 2, describe its connection to where you got lost in paper 1, and describe where you got lost in Paper 2.** Use up to one full page.

- Repeat the previous step one more time. Identify yet another paper that would help your understanding of Paper 2; let's call this Paper 3. Read Paper 3. **Briefly summarize Paper 3, describe its connection to where you got lost in previous papers, and describe where you got lost in Paper 3.** Use up to one full page.

Your entire review should be clearly readable by a fictional reader with general expertise in data structures, but with no previous familiarity with your chosen papers or the problems they attack. (Imagine the PC member who assigned you Paper 1 to review, or you six months ago.) Altogether, your review should be roughly 3–5 pages long. Don't forget to properly cite the papers you read (and any other papers that you reference in your review).

Please submit your review on Gradescope by **Monday, March 3, 2025**. Reviews will be graded using the rubric on the next page. All reviews will also be made available to the entire class to help plan potential projects.

Grading Rubric

	Excellent	Good	Needs work	Unacceptable
Paper 1 contributions ("what")	Clear summary of most significant contributions, written for someone not familiar with the paper	Discussion of key contributions, but with some confusion or lack of clarity	Summary describes some contributions, but does not identify most important contributions or contains major gaps/errors	Description of key contributions is completely unclear, incorrect, or missing
Paper 1 tools and techniques ("how")	Clear summary of most significant tools and techniques, written for someone not familiar with the paper	Discussion of key tools and techniques, but with some confusion or lack of clarity	Summary describes some techniques, but does not identify most important techniques or contains major gaps/errors	Description of key tools and techniques is completely unclear, incorrect, or missing
Paper 1 confusion	Clear description of where and why you got lost in Paper 1, written for someone not familiar with the paper	Some discussion, but not entirely clear what the problem was	No context, or discussion of paper without a clear description of where you got lost	Omitted entirely
Paper 2	Brief summary of Paper 2; connection to where you got lost in Paper 1; clear description of where you got lost in Paper 2	Some discussion, but not at excellent level	More serious inaccuracies or issues in description	Major issues; for example, not clear what paper 2 was about, or how it relates to Paper 1
Paper 3	Brief summary of Paper 3; connection to where you got lost in Papers 1 and 2; clear description of where you got lost in Paper 3	Some discussion, but not at excellent level	More serious inaccuracies or issues in description	Major issues; for example, not clear what paper 3 was about, or how it relates to Papers 1 and 2

Choosing Papers

You're welcome to choose any three related papers on any aspect of data structure design, analysis, implementation, application, or experiment, subject to the following restrictions:

- The papers must explicitly propose, analyze, evaluate, or apply a nontrivial data structure, or apply a related design, analysis, or lower-bound technique. Most papers that use only textbook data structures (arrays, lists, stacks, queues, heaps, binary search trees, black-box hash tables) don't qualify.
- Don't review papers that you already know well. I want you to deliberately go outside your comfort zone. I do *not* expect you to understand your papers in complete detail; in fact, I expect you *not* to understand your papers in complete detail.
- In particular, don't review any papers written by either you or your advisor; you have an obvious conflict of interest. Similarly, don't review any papers written by me or my students; I have an obvious conflict of interest! (But you are welcome to use those papers to find *other* papers to read and review.)

I've included a list of possible starting points in the references, all published in 2023 or later. I've tried—with limited success—to emphasize papers on topics that we will cover in class, that are relatively accessible, and that don't look like each other. Please don't be offended if your favorite papers, authors, venues, or topics are missing. You can also find a good list of older papers on the web page for Timothy Chan's Fall 2023 class. **You are *not* required to choose your papers from either of these lists.** If you don't otherwise know where to start, I recommend looking at the abstracts of these papers for something interesting.

Once you've chosen a paper to start with, there are several good strategies for finding additional papers to read, not only for this assignment or your project, but in your own research.

- **By citation**: Go backward in time: Look at the papers that your favorite papers cite. Or go forward in time: Look at the papers that cite your favorite papers, using Google Scholar or Scopus. These are often the most fruitful strategies, and the ones I recommend for this assignment, but they're not the only strategies you should use in general.
- **By keyword/topic**: Look for more papers (or lecture notes, or slides, or whatever) that use *similar* key terminology as your favorite papers. Try likely synonyms, even if you think they mean something completely different; different research communities use different words for the same concepts, and the same words for different concepts.
- **By venue:** Look at other papers in the venues (conference, journal, workshop, seminar series, arXiv category, github, . . .) that published your favorite papers.
- **By author:** Look at other papers (or lecture notes, or slides, or whatever) produced by the authors of your favorite papers, as well as their coauthors, students, and advisors.
- **By recommendation:** Ask people who are likely to know something about your favorite topics to suggest papers you might not otherwise think of.
- **By chance:** Keep your eyes peeled for anything vaguely reminiscent of your favorite papers: figures in other papers, StackExchange questions, 3Blue1Brown videos, old Martin Gardner columns, whiteboard scribbles, video games, children's toys, medieval manuscripts, or whatever. Use them to launch a Google Images/Scholar search. No, I'm not joking.

References

[1] Boris Aronov, Tsuri Farhana, Matthew J. Katz, and Indu Ramesh. Discrete Fréchet distance oracles. *Proc. 40th Int. Symp. Comput. Geom. (SOCG)*, 10:1–10:14, 2024. Leibniz Int. Proc. Informatics 293, Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[2] Muhammad A. Awad, Saman Ashkiani, Serban D. Porumbescu, Martín Farach-Colton, and John D. Owens. Analyzing and implementing GPU hash tables. *Proc. 4th Symp. Algorithmic Principles Comput. Sys. (APOCS)*, 33–50, 2023.

[3] Joyce Bacic, Saeed Mehrabi, and Michiel Smid. Shortest beer path queries in outerplanar graphs. *Algorithmica* 85(6):1679–1705, 2023.

[4] Michael A. Bender, Alex Conway, Martín Farach-Colton, Hanna Komlós, Michal Koucký, William Kuszmaul, and Michael E. Saks. Nearly optimal list labeling. *Proc. 65th Ann. IEEE Symp. Found. Comput. Sci. (FOCS)*, 2253–2274, 2024. arXiv:2405.00807.

[5] Michael A. Bender, Alex Conway, Martín Farach-Colton, Hanna Komlós, and William Kuszmaul. Layered list labeling. *Proc. ACM Manag. Data* 2(2):101:1–101:19, 2024. arXiv:2404.16623.

[6] Michael A. Bender, Alex Conway, Martín Farach-Colton, William Kuszmaul, and Guido Tagliavini. Iceberg hashing: Optimizing many hash-table criteria at once. *J. ACM* 70(6):40:1–40:51, 2023.

[7] Michael A. Bender, Alex Conway, Martín Farach-Colton, William Kuszmaul, and Guido Tagliavini. Tiny pointers. *ACM Trans. Algorithms* , 2024. arXiv:2111.12800.

[8] Michael A. Bender, Martín Farach-Colton, John Kuszmaul, and William Kuszmaul. Modern hashing made simple. *Proc. 7th Symp. Simplicity in Algorithms (SOSA)*, 363–373, 2024.

[9] Michael A. Bender, William Kuszmaul, and Renfei Zhou. Tight bounds for classical open addressing. *Proc. 65th Ann. IEEE Symp. Found. Comput. Sci. (FOCS)*, 636–657, 2024. arXiv:2409.11280.

[10] Ioana O. Bercea, Jakob Bæk Tejs Houen, and Rasmus Pagh. Daisy bloom filters. *Proc. 19th Scand. Symp. Workshops Algorithm Theory (SWAT)*, 9:1–9:19, 2024. Leibniz Int. Proc. Informatics 294, Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

[11] Benjamin Aram Berendsohn, László Kozma, and Michal Opler. Optimization with pattern-avoiding input. *Proc. 56th Ann. ACM Symp. Theory Comput. (STOC)*, 671—682, 2024. arXiv:2310.04236.

[12] Philip Bille, Inge Li Gørtz, and Tord Stordalen. Predecessor on the ultra-wide word RAM. *Algorithmica* 86(5):1578–1599, 2024.

[13] Guy E. Blelloch and Magdalen Dobson. The geometry of tree-based sorting. *Proc. 50th Int. Colloq. Automata Lang. Prog. (ICALP)*, 26:1–26:19, 2023. Leibniz Int. Proc. Informatics 261, Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[14] Prosenjit Bose, Jean Cardinal, John Iacono, Grigorios Koumoutsos, and Stefan Langerman. Competitive online search trees on trees. *ACM Trans. Algorithms* 19(3):25:1–25:19, 2023. arXiv:1908.00848.

[15] Jan van den Brand, Yang P. Liu, and Aaron Sidford. Dynamic maxflow via dynamic interior point methods. *Proc. 55th Ann. ACM Symp. Theory Comput. (STOC)*, 1215–1228, 2023.

[16] Bruce W. Brewer, Gerth Stølting Brodal, and Haitao Wang. Dynamic convex hulls for simple paths. *Proc. 40th Int. Symp. Comput. Geom. (SOCG)*, 24:1–24:15, 2024. Leibniz Int. Proc. Informatics 293, Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[17] Karl Bringmann, Nick Fischer, Ivor van der Hoog, Evangelos Kipouridis, Tomasz Kociumaka, and Eva Rotenberg. Dynamic dynamic time warping. *Proc. 35th Ann. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 208–242, 2024.

[18] Gerth Stølting Brodal, George Lagogiannis, and Robert E. Tarjan. Strict Fibonacci heaps. *ACM Trans. Algorithms* 21(2):15:1–15:18, 2025.

[19] Gerth Stølting Brodal, Casper Moldrup Rysgaard, and Rolf Svenning. External memory fully persistent search trees. *Proc. 55th Ann. ACM Symp. Theory Comput. (STOC)*, 1410–1423, 2023.

[20] Parinya Chalermsook, Manoj Gupta, Wanchote Jiamjitrak, Nidia Obscura Acosta, Akash Pareek, and Sorrachai Yingchareonthawornchai. Improved pattern-avoidance bounds for greedy BSTs via matrix decomposition. *Proc. 34th Ann. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 509–534, 2023.

[21] Timothy M. Chan and Zhengcheng Huang. Dynamic geometric connectivity in the plane with constant query time. *Proc. 40th Int. Symp. Comput. Geom. (SOCG)*, 36:1–36:13, 2024. Leibniz Int. Proc. Informatics 293, Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[22] Timothy M. Chan and Da Wei Zheng. Simplex range searching revisited: How to shave logs in multi-level data structures. *Proc. 34th Ann. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 1493–1511, 2023. arXiv:2210.10172.

[23] Li Chen, Rasmus Kyng, Yang P. Liu, Simon Meierhans, and Maximilian Probst Gutenberg. Almost-linear time algorithms for incremental graphs: Cycle detection, SCCs, s-t shortest path, and minimum-cost flow. *Proc. 56th Ann. ACM Symp. Theory Comput. (STOC)*, 1165–1173, 2024.

[24] Eldon Chung and Kasper Green Larsen. Stronger 3SUM-indexing lower bounds. *Proc. 34th Ann. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 444–455, 2023. arXiv:2203.09334.

[25] Jacobus Conradi, Anne Driemel, and Benedikt Kolbe. $(1+\epsilon)$-ANN data structure for curves via subspaces of bounded doubling dimension. *Comput. Geom. Topol.* 3(2):6:1–6:22, 2024.

[26] Bireswar Das, Anant Kumar, Shivdutt Sharma, and Dhara Thakkar. Linear space data structures for finite groups with constant query-time. *Algorithmica* 86(6):1979–2025, 2024.

[27] Danila Demin, Dmitry Sirotkin, and Stanislav Moiseev. Linear-algebraic implementation of Fibonacci heap. *J. Graph Algorithms Appl.* 28(1):27–50, 2024.

[28] Bernhard Haeupler, Richard Hladík, Václav Rozhon, Robert E. Tarjan, and Jakub Tetek. Universal optimality of Dijkstra via beyond-worst-case heaps. *Proc. 65th Ann. IEEE Symp. Found. Comput. Sci. (FOCS)*, 2099–2130, 2024.

[29] Adam Karczmarz and Marcin Smulewicz. On fully dynamic strongly connected components. *Proc. 31st Ann. European Symp. Algorithms (ESA)*, 68:1–68:15, 2023. Leibniz Int. Proc. Informatics 274, Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[30] Shashwat Kasliwal, Adam Polak, and Pratyush Sharma. 3SUM in preprocessed universes: Faster and simpler. *Proc. 8th Symp. Simplicity in Algorithms (SOSA)*, 158–165, 2025. arXiv:2410.16784.

[31] Sándor Kisfaludi-Bak and Geert van Wordragen. A quadtree, a Steiner spanner, and approximate nearest neighbours in hyperbolic space. *Proc. 40th Int. Symp. Comput. Geom. (SOCG)*, 68:1–68:15, 2024. Leibniz Int. Proc. Informatics 293, Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[32] Tuukka Korhonen, Konrad Majewski, Wojciech Nadara, Michal Pilipczuk, and Marek Sokolowski. Dynamic treewidth. *Proc. 64th Ann. IEEE Symp. Found. Comput. Sci. (FOCS)*, 1734–1744, 2023.

[33] Tuukka Korhonen, Wojciech Nadara, Michal Pilipczuk, and Marek Sokolowski. Fully dynamic approximation schemes on planar and apex-minor-free graphs. *Proc. 35th Ann. ACM-SIAM Symp. Discrete Algorithms (SODA)*, 296–313, 2024. arXiv:2310.20623.

[34] Zsuzsanna Lipták, Francesco Masillo, and Gonzalo Navarro. A textbook solution for dynamic strings. *Proc. 32nd Ann. European Symp. Algorithms (ESA)*, 86:1–86:16, 2024. Leibniz Int. Proc. Informatics 308, Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[35] Tianxiao Li, Jingxun Liang, Huacheng Yu, and Renfei Zhou. Tight cell-probe lower bounds for dynamic succinct dictionaries. *Proc. 64th Ann. IEEE Symp. Found. Comput. Sci. (FOCS)*, 1842–1862, 2023.

[36] Brice Minaud and Charalampos Papamanthou. Generalized cuckoo hashing with a stash, revisited. *Inf. Process. Lett.* 181:106356, 2023.

[37] Solon P. Pissis. Optimal prefix-suffix queries with applications. *Proc. 8th Symp. Simplicity in Algorithms (SOSA)*, 166–171, 2025. arXiv:2411.03784.

[38] Arash Pourdamghani, Chen Avin, Robert Sama, and Stefan Schmid. Seedtree: A dynamically optimal and local self-adjusting tree. *Proc. 42nd IEEE Conf. Comput. Commun. (INFOCOM)*, 1–10, 2023.

[39] Yaniv Sadeh and Haim Kaplan. Dynamic Binary Search Trees: Improved Lower Bounds for the Greedy-Future Algorithm. *Proc. 40th Int. Symp. Theoretical Aspects Comput. Sci. (STACS)*, 53:1–53:21, 2023. Leibniz Int. Proc. Informatics 254, Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[40] Corwin Sinnamon and Robert E. Tarjan. Efficiency of self-adjusting heaps. *ACM Trans. Algorithms* , 2024. arXiv:2307.02772.

[41] Robert E. Tarjan and Uri Zwick. Optimal resizable arrays. *SIAM J. Comput.* 53(5):1354–1380, 2024. arXiv:2211.11009.

[42] Haitao Wang. Unit-disk range searching and applications. *J. Comput. Geom.* 14(1):343–394, 2023.