Project Guidelines

Components of Project

1. Find a partner to work with. You can ask for an exception to work in a group of three, but **I think working in a group of two is usually best**. Ed Discussion can be used to look for students with similar interests.
2. Select, read, and understand in depth **1 research paper** on a topic related to the course material. It must involve a theoretical result. Most students will choose algorithms papers, but it is okay if your paper is about something else, as long as it uses similar tools to those studied in class. For example, you might pick a learning theory paper that uses exponential tail bounds and ε-net methods (which we will learn). Additionally, when working on your project, I fully expect you to look at other papers related to the one you choose, but you should start with one.
3. Formulate a new **research question** about that paper. This can be a theoretical question **or** an empirical question, but empirical questions need to be more complicated than "Is Algorithm A faster than Algorithm B in practice?". See below for example questions.
4. Try to solve this question, with the understanding that you may not succeed, or you might succeed in learning something different from what you originally intended! You will not be graded on if you obtain a novel, research-level result, but on how you approach the problem, what you try, and ultimately how you present your results.
5. Alternatively, pick an important result from the last few years and create a digestible exposition of the result, its implications, and some simplified proofs. This is also a valuable contribution to the research community, and can be just as challenging as proving a new result. Often, simplified proofs of existing results are the first step to improving them.
6. Write-up what you learned in a **report of at least 6 pages**. You should clearly explain the problem you aimed to solve, any necessary context to appreciate the problem's relevance, and what you accomplished. If you found barriers to accomplishing what you originally desired, describe those! Empirical papers should report any empirical findings with effective plots and figures. If you complete an empirical project, you should also share your code with me in some way (GitHub, Collab notebook, zip file).
7. A list of strong example projects from previous years can be found here: https://www.aineshbakshi.com/almds26/example_projects.html

Deadlines

- By **Friday, 2/27**, decide if you will complete a project, and choose your partner(2). Let the TA's know if you would like to meet to discuss possible project topics, or are having trouble choosing between a few options. Add your group members and selected paper to the spreadsheet.

- By **Thursday, 4/7**, submit a 1-page proposal describing the paper's subject, and listing 3 possible ideas for research questions. Alternatively, describe why you think the paper is important and why it could benefit from better exposition. The TAs and I will set up individual group meetings if needed to help you narrow down and refine your questions. Alternatively, feel free to post on ED.
- By **Friday, 5/8**: Turn in final project report.

Tips for Choosing a Paper

- Look at my list of recommended papers, which will likely grow over the next few weeks. Feel free to add to this list yourself! If you don't have a partner, put your name down next to any papers you are interested in, in case someone else is also interested.
- If you are already working on research (many Ph.D. students are) it is fine to choose a paper related to your work, but it should be theoretical and focus on an algorithmic question. If you are having trouble thinking of possible "algorithmic angles" on your research, talk to me and we can brainstorm.
- If you are an MS student or BS student who hopes to do research with me in the future (unfortunately I can usually only advise a few MS + BS thesis projects) it might make sense to choose a paper closer to my own research interests.
- If you find a paper on a topic you like, but don't want to work on that specific problem, the paper seems too complicated, etc., look at what else the authors have been publishing (either on their webpages or Google scholar.)
- Look at papers from recent conferences. Lists of accepted papers can typically be found online. Try to choose papers from relatively well regarded conferences – there will be a big difference in quality. Here are some I recommend:
  - Symposium on Foundations of Computer Science (FOCS), Symposium on Theory of Computing (STOC), Symposium on Discrete Algorithms (SODA), Innovations in Theoretical Computer Science (ITCS).
    - These are the "top 4" theoretical algorithms research conferences. Papers are typically high quality. Papers from ITCS are often on slightly more creative problems, and might be a bit shorter and less technical, so would be easier to start with.
  - European Symposium on Algorithms (ESA), International Colloquium on Automata, Languages and Programming (ICALP), International Conference on Randomization and Computation (RANDOM), International Symposium on Distributed Computing (DISC), Principles of Distributed Computing (PODC), etc.
    - Other great algorithms conferences. There are many more and I can't list them all, but feel free to run a venue by me if you are unsure of a paper's quality.
  - **Symposium on Simplicity of Algorithms**

    - **A relatively new theoretical algorithms conference emphasizing short + simple results. A great place to find project papers.**
  - Neural Information Processing Systems (NeurIPS), International Conference on Machine Learning (ICML), International Conference on Learning Representations (ICLR), AAAI Conference on Artificial Intelligence (AAAI), International Conference on Artificial Intelligence and Statistics (AISTATS)
    - Big machine learning conferences. Lots of papers, and not all contain theory (or good theory) but many do. Quality of papers is less consistent.
  - Conference on Learning Theory (COLT), Conference on Algorithmic Learning Theory (ALT).
    - Theory focused machine learning conferences, typically with high quality papers.
- **Talk to any of the TAs or me.** Even if we don't know about the specific topic you are interested in, we can often help point you in the right direction, help you find researchers whose work you might want to look at, etc. This can often save students a lot of time.

Example Research Questions

- Choosing a good research question is challenging. I'm here to help. A few tips:
  - Start small. Think about the smallest possible extension or generalization of a result you read in a paper and start with that. Big research results often come out of starting with small questions.
  - At the same time you do want to avoid *trivial* questions which e.g. could be solved by a very direct modification of the authors' techniques. It's good to find extensions that, while small, would likely have been included in the paper if the authors themselves knew how to solve them.
  - Consider a different goal: if the goal of a paper is to minimize time complexity, can you ask about space complexity? Or vice versa?
  - Consider the other side of the coin: e.g. if a paper proves that a problem can be solved with $O(S)$ space, a natural question is to try to prove that *no algorithm* can do better, or no algorithm can do better than $O(T)$ for some $T < S$. I.e., prove a lower bound. This will often be easier than improving the authors' algorithm directly, and can provide another angle to thinking about the problem.
  - The more important a problem is, the more it makes sense to ask smaller questions about it. For very important problems (e.g. distinct elements) improving even the constant factor in front of a runtime or space complexity result can really matter. Can you make a minor algorithmic change that improves the constant factor for some method? For important problems, it is also nice to find "parameter free" algorithms that are simple to implement.