**CS5330: Randomized Algorithms**

# Final Project

*Due: See below*

Randomized algorithms are important in a variety of domains: machine learning, big data, security, privacy, geometry, mechanism design, and many more! Unfortunately, we do not have time in the semester to cover all of these topics. Thus one of the major goals of the final project is to give a quick overview of all the areas in which randomized algorithms are used.

As part of the project, you will choose one or two research papers, or a specific improtant technique or algorithm. Your goal will be to teach the rest of the class about this topic, as well as to explore/extend/implement/evaluate the basic idea. More specifically, the project will consist of three parts:

1. *Explain:* Your goal is to produce a paper that explains the paper or topic or technique. The paper should not simply reproduce the result, but instead it should try to present the topic in a manner that can be understood by other members of the class. This may require including additional background material, presenting additional steps, organizing the material differently, etc. Think hard about the best way to clearly explain the algorithm/idea/technique.

2. *Presentation:* You will give a short presentation that attempts to *teach* the material to the other students in the class. Again, your task here is not simply to reproduce the claims in the paper, but instead to think hard about how best to teach the material to our class.

3. *Extend:* Finally, the third part of the project is to extend what is known about the chosen paper/algorithm/technique. This may involve extending the algorithm, proving something new. It may involve applying the algorithm, showing that it can be used to solve an intresting problem. It may involve implementing the algorithm and testing it on an interesting data set. It may involve evaluating the technique and comparing it against other possibilities. At the end of the semester, you will submit a report explaining how you have extended this algorithm

The project will be completed in teams of two. Below, you will find a list of possible papers/topics—but you may choose other options not listed as well. This week, you will submit three things:

- Your name and the name of your partner.
- Your top two choices of topics.
- For each choice, a short description of the paper or topic, and a a proposal for extensions you might do.

There are two reasons why I am asking for two topics. First, I want you to think a little bit about two different papers. Second, I want to ensure that each team is working on a different topic. If more than one team chooses the same topic, then I will see which team has the better proposal for that topic! (Or I will flip a coin.)

Overall, there will be five deadlines: the topic submission, the explanatory report, draft slides, the presentation, and the final report. Each should be submitted on IVLE (except the topic submission which should be submitted via the specified google form).

| Milestone | Deadline |
|---|---|
| Topic submission | February 19 |
| Explanatory Paper | March 19 |
| Draft slides | One week before presentation |
| Presentations | April 9 and 16 |
| Final report | April 19 |

# Possible Topics

## Graph Theory

- *Graph sparsification:* One of the most powerful techniques for speeding up graph algorithms has been sparsification. The key technique here was developed in a paper by Benczur and Karger (and generalized and extended many times since then). This has turned out to be important for a variety of real-world problems associated with network flows and more.
  *Randomized Approximation Schemes for Cuts and Flows in Capacitated Graphs*
  `https://arxiv.org/abs/cs/0207078`

- *Maximum satisfiability:* An important breakthrough in randomized algorithms came when Goemans and Williamson showed how to approximate the maximum satisfiability problem. Even more important than the result itself, the randomized rounding techniques used in this paper are a great example of a (now standard) technique for solving combinatorial optimization.
  *New $3/4$-approximation algorithms for the maximum satisfiability problem*
  `http://www-math.mit.edu/ goemans/PAPERS/GoemansWilliamson-1994-NewApproximationAlgorithmsForTheM`

## Big Data and Machine Learning

- *Stochastic Gradient Descent:* One of the most important optimization techniques used today in machine learning (and deep learning) is stochastic gradient descent. By adding a little randomization, it significantly improves the performance of traditional gradient descent. Two of the key papers on stochastic gradient descent are:
  *Making Gradient Descent Optimal for Strongly Convex Stochastic Optimization* by Rakhlin, Shamir, and Sridharan
  `https://arxiv.org/abs/1109.5647`
  *Accelerating Stochastic Gradient Descent using Predictive Variance Reduction* by Johnson and Zhang
  `http://papers.nips.cc/paper/4937-accelerating-stochastic-gradient-descent-using-predictive-varia`
  *Hogwild: A Lock-Free Approach to Parallelizing Stochastic Graident Descent* by Recht, Re, Wright, and Niu
  *Byzantine Stochastic Gradient Descent* by Alistarh, Allen-Zhu, Li
  *The Convergence of Stochastic Gradient Descent in Asynchronous Shared Memory* by Alistra, De Sa, Konstantinov

- *Dimensionality Reduction:* One of the best way to deal with big, high-dimensional data is to project it into a lower dimensional subspace. This technique of dimension reduction has turned out to be remarkably powerful for a variety of applications, e.g., classifiers, recommendation engines, etc. The critical result here is a theorem by Johnson and Lindenstrauss. Papers that rely on this include: *Nearest Neighbor Preserving Embeddings* by Indyk
  `https://graphics.stanford.edu/courses/cs468-06-fall/Papers/proxy99.pdf`
  *The fast johnson-lindenstrauss transform and approximate nearest neighbors* by Ailon and Chazelle
  `http://www.cs.technion.ac.il/ nailon/fjlt.pdf`

- *Streaming distinct elements:* One approach to big data is to treat it as a massive unending stream that you can only read once. In this paper, they develop an optimal algorithm for counting the number of distinct elements in the stream.
  *An optimal algorithm for the distinct elements problem*
  `http://www.cs.cmu.edu/afs/cs/user/dwoodruf/www/knw10b.pdf`

- *Compressive Sensing:* An amazing popular recent technique for sampling data from the world has been *compressive sensing*: it allows you to sample many fewer samples than you would expect, and yet still be able to reconstruct the data from the world. A key paper in this development:
  $(1+\epsilon)$*-approximate Sparse Recovery* by Price and Woodruff
  `https://arxiv.org/pdf/1110.4414.pdf`

- *Stochastic Optimization:* One of the key challenges in optimization is that the underlying data is often uncertain. Kleinberg, Rabani and Tardos developed a beautiful set of techniques for dealing with this uncertainty. This has led to powerful techniques for a variety of problems, such as scheduling, load balancing, bin packing, and more! This paper introduces a beautiful set of techniques for coping with randomness in the world (and has been immensely influential). *Allocating bandwidth for bursty connections* by Kleinberg, Rabani and Tardos
`http://www.cs.huji.ac.il/ yrabani/Papers/KleinbergRT-SICOMP-revised.pdf`

## Privacy and Security

- *Differential Privacy* This paper has been one of the most influential recent papers on privacy, and differential privacy has become the standard ideal for private data management. This is an immensely important paper. *Differential Privacy*
`http://people.csail.mit.edu/asmith/PS/sensitivity-tcc-final.pdf`

- *Multiparty Computation:* How do you compute something securely with other people? This is an old paper that has been immensely influential, and for which there is a huge amount of follow-up work. (You may want to also identify a more recent paper that builds on this.)
*How to play any mental game or a completeness theorem for protocols with honest majority* by Goldreich, Micali and Wigderson
`http://www.math.ias.edu/ avi/PUBLICATIONS/MYPAPERS/GMW87/GMW87.pdf`

- *Obfuscation:* How you obfuscate a program so that no one can understand what it does? It was long though to be impossible, but this breakthrough paper shows how (and created a huge amount of recent interest): *Candidate Indistinguishability Obfuscation and Functional Encryption for all circuits* `https://eprint.iacr.org/2013/451`

## Economics / Mechanism Design

- *Auctions:* This paper developed some of the first algorithms for truthful auctions with optimal outcomes (focusing on single-round, sealed bid auctions for items in unlimited supply).
*Competitive Auctions*
`https://homes.cs.washington.edu/ karlin/papers/comp-journal.pdf`

- *Mechanism Design:* A key question is how much randomization can lead to better mechanisms. Here we see that randomization provides significant improvements!
*On the Power of Randomization in Algorithmic Mechanism Design*
`https://dl.acm.org/citation.cfm?id=1748066`
`http://theory.stanford.edu/ shaddin/papers/randompower-current.pdf`

## Distributed Algorithms

- *Maximal Independent Set:* One of the key benchmark problems in distributed networks is finding a maximal independent set. This paper contains the current best known solution.
*An Improved Distributed Algorithm for Maximal Independent Set*
`https://arxiv.org/abs/1506.05093`

- *Network Coding:* Network coding is an immensely powerful technique for disseminating information more quickly in a network. These techniques have becomes increasingly prevalent, and are used in many domains.
*A Random Linear Network Coding Approach to Multicast* by Ho
`https://authors.library.caltech.edu/5107/1/HOTieeetit06.pdf`
*Analyzing Network Coding Gossip Made Easy* by Haeupler
`https://arxiv.org/abs/1010.0558`

- *Agreement Protocols:* There has been a huge amount of work on randomized consensus protocols. Consensus has turned out to be remarkably important for applications ranging from distributed lock servers to distributed transactions to Bitcoin!
  *Communication-Efficient Randomized Consensus*
  `http://www.cs.yale.edu/homes/aspnes/papers/disc2014-proceedings.pdf`
  *Tight bounds for asynchronous randomized consensus*
  `https://dl.acm.org/citation.cfm?id=1411510`

## Other options

- *LLL:* The Lovasz Local Lemma is an immensely powerful technique for proving things about weakly correlated random variables. In this paper, which created much excitement, Moser and Tardos showed how to turn this into an algorithm techniques!
  *A constructive proof of the general Lovasz Local Lemma*
  `https://arxiv.org/abs/0903.0544`

- *Smoothed analysis:* Why is the Simplex Algorithm fast in practice, even though it has poor worst-case performance? Spielman and Teng answered this question by adding a little bit of randomness!
  *Smoothed Analysis of Algorithms: Why the Simplex Algorithm Usually Takes Polynomial Time*
  `https://arxiv.org/abs/cs/0111050`

- *Smoothed analysis:* Why is knapsack easy in practice?
  *Typical Properties of Winners and Losers in Discrete Optimization* by Beier and Vocking
  `SIAM Journal on Computing, 35(4) 2006`

- *Cobra Walks:* What if you want more than one random walk? Use Cobra Walks! This a generalization of simple random walks to provide faster cover times and better performance.
  *Coalescing-Branching Random Walks on Graphs*
  `http://www.ccs.neu.edu/home/str/main_long.pdf`

- *K-Server:* The $k$-server problem is one of the classic online algorithm questions that models a variety of real-world problems where you have a set of entities/servers/etc. that need to service a collection of tasks arriving online. The question of how to do this efficiently was long unsolved, but now has recently been answered!
  *A Polylogarithmic-Competitive Algorithm for the k-Server Problem*
  `https://dl.acm.org/citation.cfm?id=2783434`
  `http://people.csail.mit.edu/madry/docs/kserver.pdf`

- *Expanders and more:* There has been a huge amount of exciting work on how to leverage a small amount of randomness to accomplish a lot. How do you build pseudorandom generators? How do you build expander graphs? These questions turn out to be very closely related! Below are a few papers that explore these topics. (The last is a survey that contains many references to look through.)
  *Extractors and pseudorandom generators* by Trevisan
  `https://people.eecs.berkeley.edu/ luca/pubs/extractor-full.pdf`
  *Loss-less condensers, unbalanced expanders, and extractors* by Ta-Shma, Umans, Zuckerman
  `http://delta-apache-vm.cs.tau.ac.il/ amnon/Papers/TUZ.C07.pdf`
  *Survey:* `https://people.seas.harvard.edu/ salil/research/unified.pdf`

## Miscellaneous Ideas

If you are interested in one of these topics and have not found a good paper, then talk to me and I can help.

- Cuckoo Hashing
- Compressed sensing (and the restricted isometry property)
- Similar estimation (e.g., Charikar 2002)
- Randomized rounding for linear programming.
- Random graphs
- Streaming algorithms and sketches
- Error-correcting codes (and coding theory)
- Stochastic gradient descent and variants.
- Spectral graph theory
- Privacy
- Cryptography
- Distributed algorithms
- Parallel algorithms
- Scheduling
- Online combinatorial optimization (e.g., EXP3, EXP4, etc., and other bandit-style learning algorithms)