Recommended Program of Study

This page describes a recommended program of study for a typical PhD student interested in algorithms and theoretical computer science. Please keep two caveats firmly in mind.

First, there is no such thing as a typical PhD student. Your program of study should be tailored to your talents, interests, educational background, and career goals.

Second, although the algorithms faculty generally agree about course requirements, we disagree about some details. We strongly encourage you to consult directly with faculty and other PhD students, both within and outside the algorithms group, as you design your curriculum. In particular, see the Advice for New PhD Students provided by our recent students.

High-level issues

- All theory students are expected to register for the algorithms seminar and give a talk every semester. This is partly so that you get regular practice in presenting research results, and partly to keep the rest of the theory group informed about your research (or at least your research interests).

- All theory students are expected to register for independent study credit (CS 591) or thesis credit (CS 599) every semester. The faculty realize (and you should, too) that new students are unlikely to produce any solid research results for the first few semesters. Nevertheless, it is important to sign up for independent study credit even in your first year, to get to know the faculty better and to get used to the research process. And who knows? You might get lucky!

- Never take more than two "real" courses (like CS 473 or Math 580) in a single semester, especially if you are also a teaching assistant. To qualify as a full-time student, you only need to register for 8 credit hours (although some fellowships require 12 hours); most graduate-level classes are worth 4 hours, and you can register for any number of seminar and independent study credits.

- If you have the choice to take a 400-level course for either 3 hours or 4 hours, take the 4-hour option. The 3-hour option is most likely equivalent to an undergraduate course, and you should only take undergraduate courses if you have serious gaps in your background.

- You will not have time to take every course you want, or even every course you need for your research. Prioritize carefully!

Theory Courses

- All algorithms students must take CS 473 (advanced algorithms) and CS 579 (complexity theory), in their first year if possible. Although you may already be familiar with some or all of the material in one or both courses, there will almost certainly be material that you have not seen in the same depth. If you have already taken an equivalent graduate-level algorithms or complexity course, we may waive this requirement, but we will still strongly encourage you to follow the course(s) informally (doing all the homework) and/or volunteer to tutor less experienced students.

- We offer at least one special topics class (CS 498 or 598) almost every semester. Take as many of these classes as possible, but no more than one per semester (at least before the qual). Classes on cryptography and logical foundations of computer science are offered almost every year; classes on randomized algorithms, approximation algorithms, computational geometry, and data structures are usually offered once every two years. Other recent special topics have included expander graphs, algorithms for massive data, mesh generation, lower bounds, and algorithmic game theory. Courses on broader topics (like randomized algorithms) are best taken as early as possible, as the material will be useful in almost any research topic. More specialized topics (like expander graphs) are best taken later.

- Consider taking graduate-level classes outside the algorithms group that have a heavy theoretical component. Examples include most courses in formal methods, programming languages, machine learning, robotics, bioinformatics, optimization, numerical analysis, information theory, and coding theory.

Math Courses

- Take Math 580 (combinatorial mathematics). Be aware that this course has a heavy workload, but it is considered an absolute prerequisite for the theory qual by most post-qual theory students. Other combinatorics classes are also useful, but less essential. Resist the urge to focus exclusively on combinatorics!

- If you didn't have a solid, theoretical linear algebra course as an undergrad, take Math 415.

- If you didn't have a solid, theoretical probability course as an undergrad, take Math 461 or 561/562.
A strong mathematics background is essential for successful theory research, but precisely *which* mathematics depends on your specific research interests. Students interested in computational geometry, for example, should consider courses in topology (e.g., Math 525 and 526) and/or differential geometry (e.g., Math 481 or 520); cryptography students should consider courses in algebra, number theory (e.g., Math 417 and 453), coding theory and information theory (ECE 556 and 563); students interested in combinatorial approximation algorithms should consider courses in optimization (Math 482, 484, 583, and 589) and operations research (IE 410, 411, 412, 413). Several other good mathematically-oriented courses are also offered by the ECE department, including random processes (ECE 534), and control systems (ECE 486).

**Non-Theory Courses**

- Take at least one graduate-level non-theoretical computer science course in your first two years. Choose a course that is related to your (potential) research interests. For example, students interested in computational geometry should consider courses in graphics, vision, robotics, learning, and/or scientific computing. Students interested in cryptography should consider courses in security and/or machine learning. Students interested in optimization or approximation algorithms should consider courses in compilers, networking, distributed systems, data mining, or bioinformatics. Courses with nontrivial programming projects are better than pure-homework classes. Remember that the ECE has several courses that overlap with computer science, like modeling and analysis of wireless networks (ECE 559PRK).

- Take at least one 500-level software class whose material is *as far as possible* from your (intended) research area. Build a system. Get your hands dirty. See firsthand what the other 90% of the computer department does, and do it better.

- Don't be afraid to think outside the box! Consider taking a course in control theory, or linguistics, or quantum mechanics, or ceramics, or systems biology, or ....