Spring 2022 – CS Special Topics

1. **CS 494* – Mansky – Provably Correct Programming**
2. **CS 594 – Kash – Modern Reinforcement Learning**
3. **CS 594 – Sun – Representations in Algorithm Design**

*CS Undergraduate students must submit a modification of major to use the class as a technical elective.

BACK TO LIST

SPRING 2022 – CS SPECIAL TOPICS

CS 494– Provably Correct Programming

- Instructor: William Mansky
- Meeting time: TR 2-3:15pm
- CRN: 42278/42289

Course Description:
Software as written today inevitably contains bugs. It mishandles corner cases and unexpected input, it allows null dereferences and buffer overflows, and it incorrectly implements the developers' goals, which may not have been clearly specified at any point in the development process. This course will explore a different approach to writing code: one in which every program is proved to satisfy a logically precise specification. The course will teach the fundamentals of logic needed to precisely describe the intended and actual behavior of programs, and introduce students to the Coq proof assistant, a tool for building machine-checked proofs of program correctness. We will pay particular attention to the trickiest aspects of modern programs, including pointer manipulation, high-performance concurrency, and persistent memory. By the end of the course, students will be able to formally prove that a program is free of common bugs and correctly implements its intended functionality, and will have at their disposal a range of techniques for writing more correct and reliable software.

Coursework (subject to change): regular in-class exercises, programming assignments, final project (no exams)

Prerequisites: CS 301. Experience with functional programming (such as F# in CS 341) is helpful but not required.

SPRING 2022 – CS SPECIAL TOPICS

CS 594– Modern Reinforcement Learning

- Instructor: Ian Kash
- Meeting time: MW 3-4:15pm
- CRN: 33649

Course Objectives:
In this course, students will become broadly familiarized with fundamental techniques of RL including value iteration, policy gradient, and actor critic methods. It will provide both practical experience using recent deep RL algorithms and an opportunity to understand where the state-of-the-art is in research in this space. Upon successful completion of this course, students should be able to pick up most recent papers on RL and when reading the introduction have some idea of the context in which it was written and the methods used. They should also be able to apply RL algorithms to problems in application areas that may interest them.

Tentative Schedule:

Week	Topic
1	Value-Based Methods
2	Policy Gradient Methods
3	Actor Critic Methods
4	Monte-Carlo Tree Search
5	RL with multiple agents
6	Alpha Go / Alpha Zero
7	Deep Q Networks: DQN + DDQN
8	Deep Q Networks: ADQN + Rainbow
9	Continuous State / Action Spaces: NAF
10	Policy Gradient Methods: TRPO
11	Actor Critic Methods: A2C, A3C
12	Learning from Demonstrations: DQfD, DDPGfD
13	Cooperative Deep RL- COMA, QMIX
14	Multi-agent Deep RL – ACPO, NeuRD
15	Final project presentations
16	Final project reports due

Spring 2022 – CS Special Topics

Student Deliverables:
There will be 3-5 homeworks during the course – written during the first third and programming after. Students will have to read all assigned readings and be expected to actively participate in class discussions. For the final project, students will submit a proposal of their selected topic by week 8, an initial draft with a review of the relevant literature and proposed research question by week 12 and a final version (updated from the initial draft based on their results) in week 16. There will also be a short presentation during week 15. Grades will be evenly split between the homeworks, the participation component, and the project component.

Prerequisites:
Prior experience with reinforcement learning (at the level of 411 or 511) or comfort learning from a relatively rapid treatment of the basics.

SPRING 2022 – CS SPECIAL TOPICS

CS 594– Representations in Algorithm Design

- Instructor: Xiaorui Sun
- Meeting time: TR 12:30-1:45pm
- CRN: 33648

Course Overview:
In the long history of algorithm design, it is crucial to transform objects to appropriate structures (i.e., representations) to make properties of objects more concrete so that solving problems on the representations becomes easier. Most notably, many matrix algorithms are based on eigendecom- position to represent matrices in terms of eigenvalues and eigenvectors.

Additionally, in the era of big data, dynamic, streaming, or parallel data processing becomes necessary, and also raises new challenges in algorithm design, as the tradeoffs between different efficiency measures need to be balanced. To design efficient algorithms in these computational models, it is also desirable to represent data properly so that dynamic, streaming, or parallel process is possible.

This course aims at timely dissemination of foundational algorithmic developments from a rep- resentation viewpoint. This course will introduce variant representations of combinatorial objects, and discuss how these representations are used to design efficient algorithms for computational problems.

Prerequisites:
You should be familiar with the following topics:

1. **Important:** Big-O notation and basic analysis of algorithms. CS 401: Computer Algorithms I or equivalent will be assumed.
2. **Important:** You should be comfortable with proofs, basic discrete math, and probability. A solid undergraduate-level mathematics background is good preparation. The course CS 151 is a good sources. If you have questions about your mathematical readiness you should contact the instructor before enrolling.
3. **Less important:** Prior graduate level coursework in theoretical computer science will be helpful, but the course should be accessible to mathematically strong undergraduates as well. It may be helpful to have taken CS 501 (Computer Algorithms II), but this is not a requirement.

The course will not include any programming.

Spring 2022 – CS Special Topics

Student Deliverables:
Students are expected to come to lecture and are encouraged to participate. Each student will prepare scribe notes for one or two lectures during the semester. All the lecture notes will be made available on the class web page. Each student will present at least one research paper to the class.

Exams:
There will be no exam.

Grading:
- Homework (30%)
- Scribe notes (30%)
- Paper presentation (30%)
- Class participation (10%)