

The Cornell, Maryland, Max Planck Pre-doctoral Research School 2018

Lectures

Lorenzo Alvisi, Cornell University

The Pit and the Pendulum

A fundamental challenge in the design of distributed data stores is to balance two conflicting goals: strong consistency (which directly translates into ease of programming) and performance. We'll explore this dilemma across distributed systems and databases—we will understand the why and the how of this tension, and become familiar with notions such as linearizability, eventual consistency, causal consistency, and the ACID/BASE dualism. Then, we will deep dive into recent attempts in both fields at bridging this gap.

Marine Carpuat, University of Maryland

Introduction to Neural Machine Translation

Machine translation has long been a grand challenge for Artificial Intelligence, and a challenging testing ground for machine learning of natural language. Recently, deep neural networks have led to dramatic increases in translation quality. In these lectures, we will introduce fundamental neural network architectures, and show how they can be used to model sequences of words in one language, as well as translate from one language into another.

Maria Christakis, MPI-SWS

Static Program Analysis Meets Test Case Generation

Sound static analysis over-approximates the set of all possible executions in a given program in order to prove the absence of errors in the program. Due to this over-approximation, sound static analysis may generate spurious warnings about executions that are not wrong or even possible in the program. To become more practical, many static analyzers give up soundness by design. This means that they do not check certain properties or that they check them under certain unsound assumptions, such as the absence of arithmetic overflow. At the other end of the soundness spectrum, we have automatic test-case generation, which typically under-approximates the set of possible program executions. The goal of test-case generation is not to prove the absence of errors in the program but, rather, their existence.

In this series of lectures, I will give an overview of the fundamental principles and techniques in static program analysis and automatic test-case generation. I will then present how these program analysis techniques can be combined to improve their overall automation, performance, and accuracy on real programs.

John P. Dickerson, University of Maryland

Ethical Market Design via Optimization

Markets are systems that empower interested parties — humans, firms, governments, or autonomous agents — to exchange goods, services, and information. In some markets, such as stock and commodity exchanges, prices do all of the “work” of matching supply and demand. Due to logistical or societal constraints, many markets, e.g., school choice, rideshare, online dating, advertising, cadaveric organ allocation, online labor, public housing,

refugee placement, and kidney exchange, cannot rely solely on prices to match supply and demand. Techniques from artificial intelligence (AI), computer science, and mathematics have a long history of both being created via, and also finding application in, the design and analysis of markets of both types. AI techniques determine how to discover structure in an uncertain matching problem, learn how to decide between matching now versus waiting, and balance competing objectives such as fairness, diversity, and economic efficiency. Yet, even defining what “best” means is important, often not obvious, and frequently involves a feedback loop between human stakeholders — each with their own value judgments — and automated systems.

This talk covers optimization- and AI-based approaches to the design and analysis of markets, along with recent approaches to aggregating value judgments of human stakeholders and incorporating them into automated matching and resource allocation systems. The talk will use (at least) three real-world markets — kidney exchange, university admissions and hiring, and rideshare — as testbeds for developed theory; furthermore, it will present and discuss various open problems in the space.

This talk covers research performed with many collaborators, most of whom are homed at Maryland, CMU, Washington University at St Louis, or Duke University.

Nate Foster, Cornell University

Probabilistic Network Programming

Formal specification and verification of computer networks has become a reality in recent years, with the rise of software-defined networking and domain-specific languages and tools. But despite many notable advances, current systems have a key limitation: they model the network as a deterministic packet-processing function. This works well in simple settings where the network implements point-to-point forwarding and the properties of interest only concern the paths used to carry traffic, but it does not give a satisfactory account of more complicated phenomena such as fault-tolerance, utilization, and randomized routing schemes. There is a fundamental mismatch between the realities of modern networks and the capabilities of existing tools—e.g., equal-cost multi-path routing and fast-failover schemes are widely used, but current frameworks cannot handle properties that are quantitative or probabilistic in nature.

In these lectures, I will present the design of a new language for programming networks based on a probabilistic semantics. First, I will show how to design a language with primitives for modeling packet-forwarding behavior as well as random choice. Second, I will develop a formal semantics based on measurable functions on sets of packet histories. I will also discuss a number of properties of the semantics including suitable notions of continuity and approximation as well as how to decide equivalence. Finally, I will also describe an implementation and present case studies showing how the language can be used to model a diverse collection of scenarios drawn from real-world networks.

Probabilistic NetKAT is joint work with colleagues from Cornell, UPenn, and UCL.

Deepak Garg, MPI-SWS

Enforcing Data Privacy in Systems

Many services critically rely on private data for providing useful functionality. For example, Google and Facebook routinely store, index, use and serve private data such as emails, online posts and user search histories. Preventing leaks and improper use of such data, both by the system’s users as well as the system’s internal developers and administrators, is important. However, this is very difficult in large systems for many reasons: Code may have bugs that inadvertently leak data, privacy policies may be specified in places different from where they need to be

checked, and privacy requirements have to be balanced against system functionality. This lecture will cover the basics of data privacy and some techniques to enforce data privacy systematically in computer systems. There are no pre-requisites besides basic familiarity with programming, operating systems and elementary discrete mathematics.

The Cornell, Maryland, Max Planck Pre-doctoral Research School 2018

Proudly powered by WordPress.