320 Sherrerd Hall
Princeton, NJ
✉ arpitg@cs.princeton.edu
www.cs.princeton.edu/~arpitg
agupta13

Arpit Gupta

Education

Summer 2018 (Expected) **Ph.D.**, *Princeton University*, *Computer Science*.
Advisor: Nick Feamster

Spring 2013 **M.S.**, *NC State University*, *Computer Science*.

Spring 2009 **B.Tech.**, *Indian Institute of Technology, Roorkee*, *Electronics & Comm.*

Professional Experience

2013–Present **Research Assistant**, *Princeton University*, Princeton, NJ.
Mentors: Nick Feamster and Jennifer Rexford
Designed and implemented: a network streaming telemetry system, *Sonata*; a software-defined Internet exchange platform, *SDX*; an industrial-scale software-defined Internet exchange platform, *iSDX*; and an event based network management tool, *Kinetic*. Also, analysed multiple active and passive measurement dataset to model ISP interconnectivity in developing regions.

Summer 2016 **Research Intern**, *Microsoft Research*, Redmond, WA.
Mentors: Ratul Mahajan and Monia Ghobadi
Designed and implemented a wide-area network controller, *Roshan*, that configures both the optical (physical) and the network layer to make optimal use of limited available resources under failures.

2011–2012 **Research Assistant**, *NC State University*, Raleigh, NC.
Mentor: Injong Rhee
Designed and implemented *WiFox*, solving the problem of performance degradation for large audience environments. This technology has been licensed out to Intel.

Summer 2011 **Research Intern**, *Google*, Mountain View, CA.
Mentor: Nandita Dukkipati
Worked on quantifying the role played by TCP time outs on Google's search traffic. Instrumented the TCP stack for Google's front end servers to collect the data required for this measurement study.

Spring 2010 **Project Assistant**, *Indian Institute of Science*, Bangalore, India.
Mentor: Anurag Kumar
Designed and implemented a WiFi AP based scheduling algorithm ensuring fairness to clients with disparate link qualities.

Publications

Conferences

Workshops

Teaching and Advising Experience

Teaching

- Advanced Computer Networks (COS 561), Princeton University — Fall 2017
- Computer Networks (COS 461), Princeton University — Spring 2016
- Securing Cyberspace with Big Data (COS 598E), Princeton University — Spring 2016
- Software Defined Networking (SDN02), Coursera — Summer 2015
- Software Defined Networking (CS 4270), Georgia Tech — Fall 2014
- Software Defined Networking (SDN02), Coursera — Summer 2014
- Computer Organization & Assembly Language, (CSC 236), NCSU — Spring 2013
- Internet Protocols (CSC 573), NCSU — Fall 2012

Guest Lecture

- Advanced Computer Networks (COS 561), Princeton University — Fall 2017
- Computer Networks (COS 461), Princeton University — Spring 2017

Advising

- *Robert MacDavid*, supervised on the iSDX [**?**] and the PathSets [**?**] projects. The two projects won the *Community* and *Best Paper* awards at USENIX NSDI and ACM SOSR respectively.
- *Rüdiger Birkner*, supervised on the iSDX [**?**], SIDR [**?**], and Sonata [**?**] projects.
- *Rob Harrison*, supervised on the Sonata [**?**] project. Currently advising on the distributed Sonata problem.
- *Bridger Hahn*, currently advising on the SonataEdge project, where we are trying to infer attacks for smart-home devices in real time.
- *David Liu*, currently advising on the SonataML project, where we are trying to co-design the algorithms for both machine learning and query planning for network telemetry tasks.
- *Jill Jermyn*, supervised on the Campus IPS project, where we explored how SDN can reduce workload for IPS boxes at Princeton. This project was the precursor to the Sonata [**?**] project.
- *Joshiah Chavula*, supervised on the African interconnection project, where we modeled the state of interconnection in the African subcontinent using various measurement platforms, such as BISMark, Ripe Atlas etc.

Awards

- Facebook Fellowship finalist, 2017
- Best Paper Award winner, ACM SOSR, 2017
- Community Award winner, USENIX NSDI, 2016
- Juniper/Comcast SDN Throwdown winner, 2016
- Facebook Fellowship finalist, 2015
- Internet-2 Innovation Award winner, 2013
- Meissner Fellowship (Purdue University) winner, 2013
- College of Engineering Fellowship (NC State University) winner, 2010
- IEEE-Motorola Innovation Award winner, 2007

Presentations

Sonata: Query-Driven Streaming Network Telemetry

- *Conferences:* ACM HotNets (11/16), NANOG 70 (05/17), P4 Workshop (05/17)
- *Industry:* Comcast (12/16), NIKSUN Inc. (06/17), AT&T (10/17)
- *Universities:* Boston University, New England Network Seminar (10/16)

iSDX: An Industrial-Scale Software Defined Internet Exchange Point

- *Conferences:* USENIX NSDI (03/16), USENIX ATC (06/16), GENI NICE (12/16)
- *Industry:* AT&T (10/15), Project Endeavour (10/15), Corsa (11/15), CloudRouter (01/16), ONF Webinar (04/16), ONF Appfest (05/16)
- *Universities:* USC, Networked Systems Laboratory (08/15)

Authorizing Network Control at Software Defined Internet Exchange Points

- *Conferences:* ACM SOSR (03/16)
- *Industry:* Verisign Inc. (08/15)

SDX: A Software Defined Internet Exchange

- *Conferences:* ACM SIGCOMM (08/14), GEC 20 (06/14), NANOG 59 (10/13), OpenIX Summit (04/15)
- *Industry:* Facebook Inc.(08/14), Microsoft (08/14)
- *Universities:* Stanford NetSeminar (10/14)

WiFox: Scaling WiFi Performance for Large Audience

- *Conferences:* ACM SIGCOMM CoNEXT (12/12)
- *Industry:* Facebook Inc.(08/14), Microsoft (08/14)
- *Universities:* Duke University (10/12), UNC Chapel Hill (10/12)

Professional Activities

Reviewer: NSDI 2014 (Shadow PC), ICNP 2016, SIGCOMM 2017, SIGCOMM 2018 Workshop on Self-Driving Networks, Transactions on Networking, Transactions on Mobile Computing, Computer Networks

Panelist: GENI-NICE 2016, Interconnection at CITP 2016

References

Prof. Jennifer Rexford
Department of Computer Science
Princeton University
35 Olden Street, CS 306
Princeton, NJ 08540
jrex@cs.princeton.edu

Prof. Nick Feamster
Department of Computer Science
Princeton University
310 Sherrerd Hall
Princeton, NJ 08540
feamster@cs.princeton.edu

Prof. Ethan Katz-Bassett
Department of Electrical Engineering
Columbia University
500 West 120th Street
New York, NY 10027
ethan@ee.columbia.edu

Dr. Walter Willinger
NIKSUN Inc.
457 N Harrison St
Princeton, NJ 08540
wwillinger@niksun.com

Research Statement

Arpit Gupta

As a **systems researcher**, I design algorithms that address problems at the intersection of **networking**, **security** and **data science** areas to build and deploy easy-to-use and scalable **real-world** systems. First, I discover real-world problems that network operators and vendors face while working "in the trenches" and understand their practical constraints. I then design innovative and provable solutions to these problems using various theoretical techniques (*e.g.*, optimization theory, and graph theory) and emerging technologies (*e.g.*, programmable data plane, scalable stream processors). Finally, I build realistic, easy-to-use, and scalable systems with minimal deployment overhead. My software packages are widely used in both academia and industry. For example, my software-defined Internet exchange point (IXP) prototype, which won the **Community Award** at *USENIX NSDI* in 2016, is used by many IXP operators across the globe. Also, the recently released, "network streaming telemetry" prototype, is currently being used by network operators and researchers at AT&T.

How can networks run themselves? My research centers around answering this question. My goal is to design and build self-driving networks that satisfy the increased security, availability, and performance requirements of modern networks, by automatically (without humans in the loop) make holistic (not protocol-specific) control decisions in real time. Although the research community explored the idea of self-driving networks in the past, there have been no notable attempts at building practically deployable solutions that satisfy current networks' requirements.

Self-driving networks need to complete the control-loop where they need to collect, analyse, and react to various network events by themselves. My dissertation research focuses on the expressiveness, scalability, and correctness of two critical building blocks for this control loop. They are:

- *Telemetry* [2, 4, 7], which refers to the process of collecting and analyzing the raw network data to discover various network activities of interest in real time. For example, inferring presence of DDoS attacks and link failures in real time; and
- *Programmatic Control* [1,3,5,6,9,10], which refers to the process of applying the fine-grained reactive actions without disrupting networks' default behavior. For example, redirecting traffic from congested peering links without creating any forwarding loops.

Telemetry with *Sonata*

Problem. For networks to run themselves, they need to monitor a wide range of network activities. For example, they need to concurrently detect whether the network is under attack and also determine whether there is a device failure in the network. This involves extracting multiple features from the traffic data and combining them together to infer activities of interests in real time. Most existing real-time telemetry systems are either not expressive, *i.e.*, they support an insufficient set of telemetry tasks, or they are not scalable, *i.e.*, they fail to scale as the traffic volume intensifies and the number of telemetry tasks increases. In contrast, self-driving networks require running *multiple expressive* telemetry queries over *high-volume* networks. My research focuses on building a distributed real-time network telemetry system that is both *expressive* and *scalable*.

Observation. Many existing telemetry solutions take advantage of either scalable stream processing or programmable data-plane targets for network telemetry—but not both. While at first glance, these two technologies seem inherently different, they both apply a sequence of transformations over packets (tuples). This similarity provides new opportunities to combine their strengths. Since for most the telemetry tasks, the fraction of relevant traffic is tiny. Thus, unlike existing task (query)-agnostic systems, it makes sense to build query-driven telemetry systems. Based on these observations, I developed an expressive query-driven streaming telemetry system, *Sonata* [2, 4], that scales query execution by making use of both streaming and programmable data-plane targets and letting the output of queries themselves drive further processing.

Challenges and Solutions. Ideally, one would like to execute all queries in the data plane itself, but limited data-plane resources (*e.g.*, memory) restricts which portions of the queries to execute in the data plane. Requiring network operators to decide how to partition each query and configure the data-plane targets

to run the partitioned queries by themselves can be overwhelming. To address this problem, Sonata provides a declarative query interface where network operators express queries for a range of telemetry tasks as a sequence of dataflow operations over packet tuples. Internally, Sonata takes these queries, representative workload data, and various data-plane constraints to determine the partitioning plan for each query that minimizes the workload at the stream processor. It then compiles the partitioned queries to target-specific configurations. Sonata's query planner also determines query-specific refinement plan that iteratively zooms-in over portions of traffic that satisfy the queries, making the best use of limited data-plane resources.

As a first step, I focused on a single-switch implementation for Sonata [2,4]; I'm currently extending this work to build a network-wide telemetry system. For the network-wide settings, Sonata's query planner needs to determine: where to evaluate each query; how to partition the stateful operations over multiple hops while making the best use of limited resources along a path; and how to select the thresholds for filter operations [7].

Impact. Building an *expressive* and *scalable* telemetry system is a big step forward for self-driving networks. This project has generated a lot of interest in self-driving networks which is evident from two, recently announced, workshops (NSF and ACM SIGCOMM) on this topic. I have also open-sourced this project, and I am currently working with researchers at ISPs (AT&T) and security vendors (NIKSUN Inc.) to express and scale queries for various real-world telemetry tasks with Sonata.

Programmatic Control with *SDX*

Problem. After determining the reactive actions with telemetry, a self-driving network needs to apply these fine-grained actions in the data plane, usually in multi-domain settings. My research focuses on applying programmatic control for the inter-domain networks settings, where by default networks use border gateway protocol (BGP) to exchange traffic with each other. In my study, I demonstrate that BGP is not suited for applying fine-grained control actions. Ideally, one would like to replace BGP with a clean-slate solution supporting programmatic control at scale. However, the Internet-wide deployment of BGP-only routers makes such an approach impractical. Thus, the goal of my research was to design and implement a dirty-slate solution that ensures maximal impact with minimal deployment overhead while safely interoperating with BGP.

Observation. In recent years, we have seen the emergence of Internet exchange points (IXPs). They provide switching fabric for various networks for exchanging traffic with each other and are strategically located to influence a significant portion of the inter-domain traffic. I proposed building a software-defined exchange (SDX) [6] that replaces the conventional switching fabric with the programmable switches at IXPs. While simple in theory, my research shows that building a practically deployable SDX requires striking a delicate balance between expressiveness, correctness, privacy, and scalability.

Challenges and Solutions. Though SDX enables flexibility at the IXP, everyone else is still using BGP. Thus, it needs to ensure that this additional flexibility does not come at the cost of creating forwarding loops in the network. Requiring SDX participants to express their control program, while guaranteeing forwarding correctness, can be cumbersome, as they not only have to worry about their control program but also their peers' programs. To address this problem, SDX provides a virtual switch abstraction to each participant which they can use to express their policies without worrying about others. SDX augments participant's programs to ensure correctness. It then composes them together and compiles the composition to forwarding table entries for the data plane. While such an augmentation ensures correctness, the cost of additional forwarding rules in the data plane can create scalability problems. I developed novel "attribute encoding" schemes [5, 6] that make use of several existing protocols and mechanisms to encode the forwarding attributes in the header of each packet before it enters the switching fabric—reducing the number of forwarding table entries required in the data plane.

To ensure that SDX participants can trust the computations at the IXP, I designed and implemented *SGDX* [8], that uses trusted execution environments such as Intel SGX to keep participants' policies private. To ensure that SDX participants can only influence traffic that they are authorized to over shared physical switch, I built *FLANC* [3], which authorizes action requests over a shared switch for each participant. Finally, to stop participants at multiple IXPs from expressing conflicting policies, I developed *SIDR* [1], which makes it easier for participants to trade expressiveness (*i.e.*, fine-grained control policies) with privacy (*i.e.*,

share control policies with other SDXs) to ensure correctness.

Impact. My SDX paper [6] had a significant impact on the research community. It was the first project to demonstrate how programmable switches can enable flexible inter-domain routing at scale. It inspired NSF's first workshop on SDX in 2014, which was followed by **multi-million grants** for research in this area. The work also inspired the creation of *Endeavour*, a multi-university consortium in Europe for SDX-related research. To date, my SDX paper [6] has over 210 citations making it one of the **highest cited** papers from *SIGCOMM 2014*.

I open-sourced the *iSDX* [5] project with the Open Networking Foundation (ONF) and worked closely with two full-time software developers to make the SDX code production ready. I also worked with various network operators, switch vendors, and service providers to help deploy SDX at an inter-government agency exchange and many IXPs across the globe. Project *Endeavour* currently uses *iSDX* as the default platform for their projects. For these efforts, the project won the **Community Award** at *USENIX NSDI 2016*. Our *PathSets* [10] paper, that demonstrated how iSDX's encoding scheme was applicable for more general network settings, won the **Best Paper** award at *SOSR 2017*.

Future Directions

Sonata and *SDX* are only the basic building blocks for self-driving networks. In future, I'd like to build-up on these works to continue working on the problems related to self-driving networks to make this distant dream a reality.

Expanding Networking Telemetry Footprint

Self-driving networks require insights from multiple vantage points in the network. So far, *Sonata* focuses primarily on inferring network activities at the core of the Internet (*e.g.*, border routers for large ISPs). Though such deployments provide broader visibility with limited overhead, they miss several insights such as end-to-end performance for specific web applications (*e.g.*, Netflix, CNN, etc.) in a region. These insights, available only at the edge of the Internet (*e.g.*, home routers, web browsers, etc.), are not only critical for robust self-driving networks but are also key to answering various policy questions on socio-political issues such as net neutrality, data privacy, censorship, etc.

Existing tools, such as BISmark, Ripe Atlas, etc., are not suited for edge telemetry as they are neither expressive or scalable. They either collect too much or too little data for analysis and are not suited for real-time query-driven telemetry. For example, one can use RIPE Atlas nodes to only express a limited set of telemetry queries tied to statically configured tools such as `ping`, `traceroute` etc. or use BISmark to collect packet traces only for limited duration and locations. I observed that it is possible to use cheaply available tools, such as programmable NICs ($< \$40$) and Raspberry Pis ($< \$50$) as data-plane and streaming targets respectively to build Sonata-like edge telemetry system.

Requiring network operators to express telemetry queries, while taking the computational capabilities of edge nodes into consideration and individually configure each edge node, can be overwhelming. Similar to Sonata's query interface, I envision providing a centralized query interface which abstracts away such details from the network operators. Under the hood, it determines a query plan that makes the best use of local compute resources before sending the packet tuples to a central processor.

To make this project more practical, I envision to design a wide-scale user study with *SonataEdge* to not only understand the real-world challenges for running it as scale, but also provides a platform for policy experts, journalists, and ISPs to answer various performance and policy related questions. For example, I am currently designing a wide-scale user study to quantify the relationship between subscribed uplink bandwidths and bit rates for streaming applications (*e.g.*, Netflix) for home networks.

Enabling Privacy-Preserving Network Telemetry

As we expand the footprint of network telemetry to build more robust self-driving networks, we encounter scenarios where consumers of the data are not the ones producing it. For example, end users generate quality-of-experience data for various latency-sensitive applications, but an eyeball ISP can consume this data to improve their network's performance. Though, in some cases, data consumers implicitly collect data in exchange for services they offer. However, currently, there are no solutions that let data producers explicitly share their data in a privacy-preserving manner. My goal is to enable data producers chose *what* portions of their data they'd like to share and with *whom* aiding networks in running themselves.

One approach is to share the data over multiple remote servers and rely on existing secure multi-party computation (SMPC) tools to ensure privacy-preserving computations. However, such an approach is query-agnostic and thus requires transferring all the (partitioned) raw data to remote servers for processing. An alternative is to process the data locally in a decentralized manner. I envision to build a system, which taps into the recent advances in the area of federated learning, where data consumers can execute their learning models over computing nodes that are local and private to end users. An end user can either use a dedicated device such as *SonataEdge*, or their mobile devices as local compute nodes. I also envision designing protocols that let remote data consumers train their models using these remote nodes to process the data locally in a privacy-preserving manner.

Developing such a decentralized system and protocols that are expressive, scalable, robust, and secure is non-trivial. My research agenda comprises of three inter-related thrusts that address these challenges: (1) *Expressive User Interface* for end users to express *what* data to trade and with *whom*. I envision collaborating with programming logic and privacy experts to create a domain-specific language for expressing data sharing policies; (2) *Data Privacy and Integrity* guarantees are required for data producers and consumers respectively. To guarantee data privacy, I plan to extend my previous work (*FLANC* [3]) to employ formal logic tools for generating policy compliance proofs for input telemetry queries. I also plan to design protocols that make use of TPMs at local nodes for guaranteeing data integrity; and (3) *Model Privacy* guarantees are required for data consumers. Learning models are the intellectual property of the consumers, and they might want to keep it private. To guarantee model privacy, I plan to design protocols that make use of existing works in the area of differential privacy and secure multi-party computation to ensure model privacy.

Co-designing Machine Learning and Query Planning Algorithms

Self-driving networks need to train learning models to discover activities of interests. Currently, learning algorithms only focus on training the learning models to maximize accuracy, and query planning algorithms solely focus on minimizing the execution cost for queries that represent these learning models. This disconnect between the two algorithms might result in very accurate, yet prohibitively expensive, learning models that cannot be deployed in the production networks.

I observed that it is possible to augment the optimization problem that most state-of-the-art learning algorithms solve to trade accuracy with the query execution cost while taking various resource constraints (*e.g.*, limited memory in the data plane) to determine the learning model as well as the query plan. Based on this observation, I'd like to design and build telemetry systems, *SonataML*, that co-designs the algorithm for the machine learning and query planning problems to train realistic and deployable learning models.

Though the idea appears to be straightforward in principle, yet just embedding the cost and constraint metrics to the learning problems is not enough. The computational complexity of optimal solution grows exponentially with the increasing number of features for the learning problem. One needs to design heuristic algorithms that can find good enough solutions to this problem.

Another challenge is the availability of labeled training data. As most of the activities of interests are needles in a haystack, collecting labeled training data to bootstrap *SonataML* is extremely hard. I envision adapting existing active learning algorithms that can leverage the knowledge of the domain experts (security analysts and network operators) to generate as much of labeled data as possible while minimizing the workload for these human experts. As the nature of security and performance problems evolve with time, rather than replacing humans in one blow, I envision the active learning to be an intrinsic part of the self-driving networks, where the workload for the human experts reduces over time as the networks become more capable of running the networks by themselves.

To ensure that this solution is practically deployable, I plan to work with my collaborators at cloud (Google, Microsoft, etc.) and Internet (AT&T and Comcast) service providers, who have already shown great interest in this research direction. I also plan to work with my collaborators that make programmable switches (Barefoot Networks) to explore switch architectures that are better suited to support in-network machine learning.

Beyond Self-Driving Networks

The computational complexity of machine learning models has grown significantly in recent years. To catch up with this increased computational requirements, most cloud service providers have responded by

horizontally scaling the compute clusters. However, rather than mindlessly throwing more resources, one needs to step back and think more deeply to explore if there are new technologies on the horizon that can make better use of limited available resources.

I believe programmable data-plane targets provide the opportunity to vertically scale the analytics computation, *i.e.*, *Sonata*'s idea of offloading query execution to data-plane targets applies to more general analytics settings. I plan to design and implement **general-purpose analytics stack** that is not only aware of the compute capabilities and limitations of CPU-based compute nodes but also the set of data-plane targets connecting these nodes.

To make this system a reality, I need to first demonstrate that using data-plane targets for offloading analytics computations is cheaper than using more CPU nodes. Answer what kind of data center topologies will make the best use of programmable data-plane targets? What kind of targets to use, *i.e.*, whether one should deploy cheaper but low throughput targets (*e.g.*, smart NICs), or high throughput but expensive targets (*e.g.*, switching fabrics), or both? Also, whether one should use off-the-shelf programmable data-plane targets or design custom switch architectures tailored to data analytics tasks? Second, I plan to design a query planner that for given topology, compute nodes and programmable data-plane targets; determines packet processing pipelines that execute over both the data-plane and the streaming targets. The goal of this query planner is to minimize the execution time while considering various data-plane constraints (*e.g.*, limited memory) and limitations (*e.g.*, cannot perform multiplication or division operation). The query planner also needs to determine the data structures for packet's payload—reducing the parsing overhead for the intermediate data-plane targets. Finally, I'd also like to expand the idea of co-designing the algorithms for learning and query planning by applying it to more commonly-used machine learning problems.

References

[1] R. Birkner, A. Gupta, N. Feamster, and L. Vanbever. SDX-Based Flexibility or Internet Correctness?: Pick Two! In *ACM Symposium on SDN Research (SOSR)*, 2017.

[2] A. Gupta, R. Birkner, M. Canini, N. Feamster, C. Mac-Stoker, and W. Willinger. Network Monitoring as a Streaming Analytics Problem. In *ACM Workshop on Hot Topics in Networks (HotNets)*, 2016.

[3] A. Gupta, N. Feamster, and L. Vanbever. Authorizing Network Control at Software Defined Internet Exchange Points. In *ACM Symposium on SDN Research (SOSR)*, 2016.

[4] A. Gupta, R. Harrison, A. Pawar, R. Birkner, M. Canini, N. Feamster, J. Rexford, and W. Willinger. Sonata: Query-Driven Network Telemetry. *arXiv preprint arXiv:1705.01049*, 2017.

[5] A. Gupta, R. MacDavid, R. Birkner, M. Canini, N. Feamster, J. Rexford, and L. Vanbever. An Industrial-Scale Software Defined Internet Exchange Point. In *USENIX NSDI*, 2016.

[6] A. Gupta, L. Vanbever, M. Shahbaz, S. P. Donovan, B. Schlinker, N. Feamster, J. Rexford, S. Shenker, R. Clark, and E. Katz-Bassett. SDX: A Software Defined Internet Exchange. In *ACM SIGCOMM*, 2014.

[7] R. Harrison, C. Qizhe, A. Gupta, and J. Rexford. Network-Wide Heavy Hitter Detection with Commodity Switches. *Under Submission*.

[8] X. Hu, **Arpit Gupta**, A. Panda, N. Feamster, and S. Shenker. Preserving Privacy at IXPs. *Under Submission*.

[9] H. Kim, J. Reich, A. Gupta, M. Shahbaz, N. Feamster, and R. Clark. Kinetic: Verifiable Dynamic Network Control. In *USENIX NSDI*, 2015.

[10] R. MacDavid, R. Birkner, O. Rottenstreich, A. Gupta, N. Feamster, and J. Rexford. Concise Encoding of Flow Attributes in SDN Switches. In *ACM Symposium on SDN Research (SOSR)*, 2017.

Teaching Statement

Arpit Gupta

Teaching

I believe teaching is a position of power, and thus a position of great responsibility. I look forward to the responsibility and privilege to teaching and advising students. Among many things, I believe a teacher has two primary responsibilities:

- Exposing students to problems that help them discover their interests; and
- Empowering them with the skills, confidence, and independence required to solve these complex problems.

A good teacher uses a course not only to impart knowledge but also to instill excitement about the material and help students discover new interests. Lectures provide a unique opportunity to present new problems to students and challenge them to think in new ways. A good lecture should pay particular attention to engage students by making the unfamiliar topics more relatable to them. For example, in my lectures, rather than speaking exclusively in the abstract, I motivated content-delivery networks with cat videos and used real-world configurations from Princeton's network to explain firewalls and border routing. I also believe a good lecture should discuss recent research advancements, showing students how the field is still evolving and encouraging them to make an impact of their own.

Given the vast array of careers that students embark upon after graduation, I believe that it is crucial to provide students with transferable problem-solving skills. I think programming assignments are the right medium to hone such skills. They not only help students to understand the concepts taught in the classroom more deeply by applying them in practice but also provides them a practical system building experience using various state-of-the-art tools. For example, one of my programming assignment required students to express monitoring tasks as map-reduce queries using Apache Spark. A few months later, some of the students complimented me saying that their experience with Spark helped them excel in their internships that were unrelated to networking.

I also believe, translating cutting-edge research into programming assignments not only creates a lot of excitement for the students but also adds a new dimension to the research itself. For example, my experience designing programming assignments for the *SDX* [8] and the *Sonata* [9] projects forced me to make the programming interface for these systems more intuitive and easy-to-use. This experience paid-off when I had to train network operators deploying my solutions in the production network.

So far, I have worked with the instructors for Princeton's undergraduate (COS 461 [1]) and graduate (COS 561 [2]) networking courses to prepare precept lectures and designed programming assignments. I have also contributed to the graduate seminar course on **Securing Cyberspace with Big Data** (COS 598 [3]) at Princeton and the undergraduate seminar course on **Software-Defined Networking** (CS 4270 [5]) at Georgia Tech. I have also worked as TA for the popular *Software-Defined Networking* course over **Coursera** [4] . My teaching experiences confirmed that I enjoy teaching at all levels and over all mediums.

Among all my teaching experiences, the SDN Coursera course was the most satisfying experience. It provided me with an opportunity to reach out to students of different ages and background that were beyond the reach of conventional education systems. Many of the students were mid-career networking employees honing their programming skills to embrace the changing ecosystem, *i.e.*, shift to automated programmable networking. Being able to contribute to such a massive workforce retraining was an extraordinary experience for me. I have also contributed to the workforce retraining efforts with online seminars on *SDX* and *Sonata* projects.

I am looking forward to opportunities for teaching **operating systems**, **distributed systems**, **computer networks**, **cybersecurity**, and **big-data systems** courses at the undergraduate and graduate levels. I am also interested in developing interdisciplinary graduate courses (with possibly a *MOOC* version) that combine the areas of **networking**, **systems**, **security**, and **data science**.

Advising

I believe an advisors' goal is to help students discover their interests and develop their taste of problems. I take students background and research experience into consideration while helping them identify problems where they can capitalize on their strengths. For example, one of my student, Robert MacDavid,

did an undergraduate thesis in *theory* and the other, Rüdigar Birkner worked as *systems* engineer in the industry for a couple of years. I worked with them to design *attribute-encoding* algorithms and build the production-quality system for the iSDX project respectively. Giving them a well-defined problem, tailored to their strengths, at the beginning boosted their confidence to do research and make an impact. Both these students, enrolled for a Master's program, transferred to Ph.D. programs at their respective institutions. I encouraged Robert to apply the *attribute-encoding* algorithms to more general settings, and it was a very proud moment when this project won the **best-paper** award at SOSR 2017 [7].

I also believe it is imperative for systems researchers to interact with developers, operators, and vendors "in the trenches" to learn about real-world problems and identify the ones that require a more principled solution. For example, my interactions with network operators at NANOG encouraged me to revisit the SDX project and make the platform scale for production networks while satisfying the realistic hardware constraints [6]. I benefited from my mentors' relationship with the industry (*e.g.*, researchers, and operators at Microsoft, Google, AT&T, Comcast, NIKSUN Inc., etc.). I intend to do the same for my students. I will also encourage my students to learn more about the real-world problems with regular site visits and internships.

As there are many interesting problems to solve. Identifying the ones that are interesting and impactful requires training. I believe advisors should play an active role in developing the research taste for their students, training them to identify interesting problems and not waste their time on the dull ones. Both, Nick Feamster and Jennifer Rexford, taught me by providing constructive feedback, encouraging me to refine my ideas iteratively and pursue impactful research problems. I intend to do the same for my students.

I was privileged to get an opportunity to advise many undergraduate and graduate students. I have mentored seven graduate students from Princeton University (Robert MacDavid, Bridger Hahn, David Liu, and Rob Harrison), ETH Zürich (Rüdiger Birkner), Columbia University (Jill Jermyn), and University of Cape Town (Josiah Chavula) on specific projects and many junior students (Hooman Mohajeri, Disney Yan Lam, and Laura Roberts) in general. Personally, I learned a lot from this experience, and it further strengthened my resolve to pursue a career in academia where I get the privilege not only to teach but also mentor bright students.

References

[1] COS 461: Computer Networks, Princeton University, Spring 2016. https://www.cs.princeton.edu/courses/archive/spring16/cos461/.

[2] COS 561: Advanced Computer Networks, Princeton University, Fall 2017. http://www.cs.princeton.edu/courses/archive/fall17/cos561/.

[3] COS 598: Securing Cyberspace with Big Data, Princeton University, Spring 2016. https://registrar.princeton.edu/course-offerings/course_details.xml?courseid=002132&term=1164.

[4] Coursera SDN Course. class.coursera.org/sdn-002/.

[5] CS 4270: Software Defined Networking, Georgia Tech, Fall 2014. http://noise.gatech.edu/classes/cs8803sdn/fall2014/.

[6] A. Gupta, R. MacDavid, R. Birkner, M. Canini, N. Feamster, J. Rexford, and L. Vanbever. An Industrial-Scale Software Defined Internet Exchange Point. In *USENIX NSDI*, 2016.

[7] R. MacDavid, R. Birkner, O. Rottenstreich, A. Gupta, N. Feamster, and J. Rexford. Concise Encoding of Flow Attributes in SDN Switches. In *ACM Symposium on SDN Research (SOSR)*, 2017.

[8] SDX Programming Assignment. https://github.com/PrincetonUniversity/Coursera-SDN/tree/master/assignments/sdn-ixp.

[9] Sonata Programming Assignment. https://github.com/Sonata-Princeton/SONATA-DEV/tree/tutorial/sonata/tutorials/Tutorial-1.