

Wyatt Lloyd  
Computer Science Department  
35 Olden St.  
Princeton, NJ 08540  
(814) 880-1392  
wlloyd@cs.princeton.edu

December 14, 2012

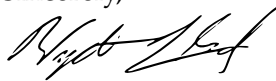
To Whom It May Concern:

I am seeking a tenure-track position as an assistant professor in your department. I am a Ph.D. candidate in the Computer Science Department at Princeton University, and I expect to graduate this year. My research interests include the distributed systems and networking problems that underlie the architecture of large-scale websites, cloud computing, and big data.

I have enclosed my curriculum vitae with a list of references, research statement, teaching statement, and four representative papers. The most up-to-date version of these materials is available online at <http://www.cs.princeton.edu/~wlloyd/application/>.

I look forward to discussing my application with you.

Sincerely,



Wyatt Lloyd

- encl:
- Curriculum Vitae (including name of references)
  - Research Statement
  - Teaching Statement
  - “Stronger Semantics for Low-Latency Geo-Replicated Storage”  
**Wyatt Lloyd**, Michael J. Freedman, Michael Kaminsky, David G. Andersen  
to appear in NSDI 2013, 14 pages (preprint)
  - “Don’t Settle for Eventual: Scalable Causal Consistency for Wide-Area Storage with COPS”  
**Wyatt Lloyd**, Michael J. Freedman, Michael Kaminsky, David G. Andersen  
from SOSP 2011, 16 pages
  - “Coercing Clients into Facilitating Failover for Object Delivery”  
**Wyatt Lloyd**, Michael J. Freedman  
from DSN 2011, 12 pages
  - “Prophecy: Using History for High-Throughput Fault Tolerance”  
Siddhartha Sen, **Wyatt Lloyd**, Michael J. Freedman  
from NSDI 2010, 16 pages

# Wyatt Lloyd

<http://www.cs.princeton.edu/~wlloyd>

205 Hudson St., Apt 601  
Hoboken, N.J. 07030

wlloyd@cs.princeton.edu  
(814) 880-1392

## Education

**Princeton University** ..... Princeton, NJ  
Ph.D. in Computer Science ..... Expected June 2013  
M.A. in Computer Science ..... 2009  
*Advisor:* Michael J. Freedman

**Pennsylvania State University, University Park** ..... State College, PA  
B.S. in Computer Science with distinction ..... 2007  
Schreyer Honors College  
*Advisor:* Thomas F. La Porta

## Research Interests

The distributed systems and networking problems that underlie the architecture of large-scale websites, cloud computing, and big data.

## Dissertation

2010– **Stronger Consistency and Semantics for Geo-Replicated Storage.** Geo-replicated storage systems provide the backend for massive-scale websites such as Twitter and Facebook, storing data that includes your profile, friends lists, and status updates. These storage systems seek to provide an “always-on” experience where operations always complete quickly, because of a widely demonstrated link between page load times, user engagement, and revenue. We term systems that can handle data of this scale and provide the always-on experience *ALPS systems*, because they provide four key properties—availability, low latency, partition tolerance, and scalability.

Our COPS [2] system is the first distributed data store to guarantee the ALPS properties and achieve consistency stronger than eventual. Eventual consistency specifies only that writes in one datacenter eventually show up in the others. Causal consistency, which is what COPS provides, maintains the partial order over operations established by potential causality. Under causal consistency, all of a user’s operations appear in the order they are issued and interactions between users, e.g., conversations in comments, appear in their correct order as well. This improvement in consistency gives users a better experience and makes the data store easier for programmers to reason about. A key technical contribution of the COPS work is its fully distributed and scalable architecture that uses explicit metadata and off-path dependency checks to enforce ordering instead of relying on any single point of coordination.

Our Eiger [1] system further pushes on the semantics an ALPS data store can provide. Eiger provides high-performance, guaranteed low-latency read-only and

write-only transactions across the thousands of machines in a cluster. Read-only transactions allow a client to observe a consistent snapshot of an entire cluster. Write-only transactions allow clients to atomically write many values spread across many servers at a single point in time. One important use case for write-only transaction is maintaining symmetrical relationships, e.g., Alice “isAFriendOf” Bob and Bob “isAFriendOf” Alice should both appear or disappear at the same time. Eiger also improves the semantics of ALPS data stores by providing the column-family data model—which is used in BigTable and Cassandra, and can be used to built real applications like Facebook—instead of the key-value data model provided by COPS—which is useful mainly as an opaque cache.

My dissertation research shows that ALPS systems do not need to settle for eventual consistency and weak semantics. Taken together, Eiger and COPS show that causal consistency and stronger semantics are possible for low-latency geo-replicated storage.

## Other Research Experience

- 2009–2011 **Low-Overhead Transparent Recovery for Static Content.** Client connections to web services break when the particular server they are connected to fails or is taken down for maintenance. We designed and built TRODS [3], a system that transparently recovers connections to web services that delivers static content, e.g., photos or videos. TRODS is implemented as a server-side kernel module for immediate deployability, it works with unmodified services and clients. The key insight in TRODS is its use of cross-layer visibility and control: It derives reliable storage for application-level state from the mechanics of the transport layer. In contrast with more general recovery techniques, the overhead of TRODS is minimal. It provides throughput-per-server competitive with unmodified HTTP services, enabling recovery without additional capital expenditures.
- 2007–2010 **Using History for High-Throughput Fault Tolerance.** Byzantine fault-tolerant (BFT) replication provides protection against arbitrary and malicious faults, but its performance does not scale with cluster size. We designed and built Prophecy [4], a system that interposes itself between clients and any replicated service to scale throughput for read-mostly workloads. Prophecy relaxes consistency to delay-once linearizability so it can perform fast, load-balanced reads when results are historically consistent, and slow, replicated reads otherwise. This dramatically increases the throughput of replicated services, e.g., the throughput of a 4 node Prophecy web service is ~4X the throughput of a 4 node PBFT web service.
- 2007 **IP Address Passing for VANETs.** In Vehicular Ad-hoc Networks (VANETs), vehicles have short connection times when moving past wireless access points. The time required for acquiring IP addresses via DHCP consumes a significant portion of each connection. We reduce the connection time to under a tenth of a second by passing IP addresses between vehicles. Our implementation improves efficiency, reduces latency, and increases vehicle connectivity without modifying either DHCP or AP software [5].

2006–2007 **Multi-Class Overload Controls for SIP Servers.** When SIP servers that are used for signaling in VoIP network are overloaded, call-setup latency increases significantly and critical calls—e.g., 911 calls—can be denied. My undergraduate thesis on multi-class overload controls [6] reduces call latency by suppressing retransmissions and prioritizes critical calls so they always connect.

## Professional Experience

- 9/07– **Research Assistant.** Princeton University, Princeton, NJ  
Major projects include providing stronger consistency for scalable storage systems (COPS), providing stronger semantics for scalable storage systems (Eiger), enabling transparent connection recovery for web services (TRODS), and using history for high-throughput fault tolerance (Prophecy).
- 5/12–8/12 **Ph.D. Intern.** Facebook, New York, NY  
Worked on a distributed-storage-systems team on a project to improve caching for static content. Was the first intern at the new New York office.
- 6/10–9/10 **Summer Research Fellow.** Intel Labs Pittsburgh / CMU, Pittsburgh, PA  
Began leading the COPS project, a collaborative effort between Princeton University, Intel Labs, and Carnegie Mellon University.
- 6/07–9/07 **Intern-Student Engineer.** The Boeing Company, Anaheim, CA  
Worked on an internal research and development project on routing in multi-tier wireless networks as part of the Network Systems group.
- 5/06–9/06 **Intern-Student Engineer.** The Boeing Company, Anaheim, CA  
Worked on an internal research and development project that utilized DHCP for intra-domain mobility management as part of the Network Systems group.

## Teaching Experience

- 11/29/12 **Guest Lecturer.** Distributed Systems, (CMU) 15-440  
Lectured on COPS to introduce cutting-edge research to undergraduates.
- 10/4/12 **Guest Lecturer.** Advanced Computer Networks, COS-561  
Lectured on inter-domain routing with BGP and led a discussion of research papers on network isolation and software-defined networking.
- 2/09–6/09 **Teaching Assistant.** Computer Networks, COS-461  
Graded, held office hours, helped design exams, and taught exam-review sessions.
- 9/08–1/09 **Teaching Assistant.** General Computer Science, COS-126  
Graded, held office hours, helped design exams, and taught twice-weekly recitations.

## Service

- 11/9/12 **Panelist.** Princeton Women in Computer Science Graduate School Panel  
Shared experiences and advice about graduate school.

10/11–	<b>Regional Lead.</b>	Siebel Scholars Foundation Organized events for Princeton region and served on the advisory board.
6/11–8/11 6/09–8/09	<b>Student Advisor.</b>	Princeton Summer Programming Experience Advised novice undergraduate programmers on 6-week-long projects.
2/06–5/07	<b>Student Representative.</b>	Penn State CSE Curriculum Committee Helped shape undergraduate Computer Science curriculum.

## Honors

2012	Wu Prize for Excellence (Princeton)
2012	Facebook Fellowship Finalist
2012	Siebel Scholar
2007	Princeton University Graduate Fellowship
2003-2007	Dean's List (Penn State)
2003-2007	Schreyer Honors College Scholar (Penn State)
2006	College of Engineering General Scholarship (Penn State)
2003	Maryland State Distinguished Scholar
2002	National Merit Scholarship Honorable Mention
2000	Eagle Scout

## Refereed Conference Publications

- [1] **Wyatt Lloyd**, Michael J. Freedman, Michael Kaminsky, and David G. Andersen. Stronger Semantics for Low-Latency Geo-Replicated Storage. To appear in *Proc. 10th Symposium on Networked Systems Design and Implementation (NSDI 13)*, April 2013. 14 pages.
- [2] **Wyatt Lloyd**, Michael J. Freedman, Michael Kaminsky, and David G. Andersen. Don't Settle for Eventual: Scalable Causal Consistency for Wide-Area Storage with COPS. In *Proc. 23rd ACM Symposium on Operating Systems Principles (SOSP 11)*, October 2011. 16 pages.
- [3] **Wyatt Lloyd** and Michael J. Freedman. Coercing Clients into Facilitating Failover for Object Delivery. In *Proc. 41st IEEE/IFIP International Conference on Dependable Systems and Networks, Dependable Computing and Communication Symposium (DCCS) track (DSN 11)*, June 2011. 12 pages.
- [4] Siddhartha Sen, **Wyatt Lloyd**, and Michael J. Freedman. Prophecy: Using History for High-Throughput Fault Tolerance. In *Proc. 7th Symposium on Networked Systems Design and Implementation (NSDI 10)*, April 2010. 16 pages.
- [5] Todd Arnold, **Wyatt Lloyd**, Jing Zhao, and Guohong Cao. IP Address Passing for VANETs. In *Proc. 6th IEEE International Conference on Pervasive Computing and Communications (PERCOM 08)*, March 2008. 10 pages.

## Theses

- [6] **Wyatt Lloyd.** Multi Class Overload Controls for SIP Servers. *Honors Thesis*, The Pennsylvania State University, May 2007.

## Refereed Conference Presentations

- [7] Stronger Semantics for Low-Latency Geo-Replicated Storage. To appear at *10th Symposium on Networked Systems Design and Implementation (NSDI 13)*, April 2013.
- [8] Don't Settle for Eventual: Scalable Causal Consistency for Wide-Area Storage with COPS. In *23rd ACM Symposium on Operating Systems Principles (SOSP 11)*, October 2011.
- [9] Coercing Clients into Facilitating Failover for Object Delivery. In *41st IEEE/IFIP International Conference on Dependable Systems and Networks, Dependable Computing and Communication Symposium (DCCS) track (DSN 11)*, June 2011.

## Other Presentations

- [10] Don't Settle for Eventual: Scalable Causal Consistency for Wide-Area Storage with COPS. Facebook Ph.D. Intern and Distributed Systems Reading Group Talk, August 2012.
- [11] Don't Settle for Eventual: Scalable Causal Consistency for Wide-Area Storage with COPS. Berkeley, Cloud Seminar, April 2012.
- [12] Don't Settle for Eventual: Scalable Causal Consistency for Wide-Area Storage with COPS. Intel Science and Technology Center on Cloud Computing Retreat, Research Talk, December 2011.
- [13] Don't Settle for Eventual: Scalable Causal Consistency for Wide-Area Storage with COPS. University of Maryland, SysChat Group Talk, October 2011.
- [14] Don't Settle for Eventual: Scalable Causal Consistency for Wide-Area Storage with COPS. Johns Hopkins University, Computer Science Seminar, October 2011.

## References

**Prof. Michael J. Freedman**  
Assistant Professor  
Computer Science Department  
Princeton University  
mfreed@cs.princeton.edu

**Dr. Michael Kaminsky**  
Senior Research Scientist  
ISTC for Cloud Computing  
Intel Labs  
michael.e.kaminsky@intel.com

**Prof. David G. Andersen**  
Associate Professor  
Computer Science Department  
Carnegie Mellon University  
dga@cs.cmu.edu

**Prof. Mike Dahlin**  
Professor  
Computer Science Department  
The University of Texas at Austin  
dahlin@cs.utexas.edu

# Research Statement

Wyatt Lloyd

## Vision

My research addresses emerging problems in the massive-scale distributed systems that support big data. The recent and dramatic growth in the demands on these systems—on their ability to store, process, and manage large data volumes—makes this area ripe for new research. This new scale, e.g., all of Twitter’s tweets, requires massively distributed systems with hundreds or thousands or more machines cooperating to provide the necessary capacity and throughput.

To handle the magnitude of this data, recent systems from industry and academia have stressed performance and scalability at the cost of strong semantic properties, e.g., linearizability and transactions, yet these same properties make systems easier to use and understand. My research questions whether such sacrifices are necessary and seeks to identify properties that make these systems easier to program to and reason about that are compatible with massive scale. I pursue properties that are rigorously defined for the clarity they bring to programmers using the systems, e.g., causal consistency instead of eventual, or guaranteeing low latency for all operations.

Looking at my research from another direction, it reexamines conventional design decisions and approaches given the new reality of big data. Centralized approaches that were simple, straightforward, and effective in the small systems of the past quickly hit bottlenecks that prevent scaling to the large systems of today. In contrast, I design systems that provide strong properties while keeping all operations distributed, so they can scale to the even larger systems of tomorrow.

My focus on scalability is maintained throughout the research process; I include it as a primary design requirement and then build prototypes that are tested with real data at scale. While building and experimentally verifying that a design is scalable is useful and necessary in itself, I have also found that doing so often exposes unanticipated bottlenecks and can reveal important new research topics, as discussed in my future directions. This focus on designing and building massive-scale systems that provide strong, rigorous properties defines my niche as a researcher.

## Dissertation Research

Geo-replicated storage systems provide the backend for massive-scale websites like Facebook, storing data that includes your profile, friends list, and status updates. These storage systems seek to provide an “always-on” experience where operations always complete quickly, because of a widely demonstrated link between page load times, user engagement, and revenue. We term systems that can handle such data at scale and provide an always-on experience as *ALPS systems*, because they provide four key properties: Availability, Low latency, Partition tolerance, and Scalability.

Previous ALPS systems such as Amazon’s Dynamo, LinkedIn’s Project Voldemort, and Facebook’s Cassandra (in some configurations) all made large usability sacrifices in pursuit of their scale and performance goals. These sacrifices—such as providing only eventual consistency, which gives no ordering guarantees about operations—make it hard for programmers to reason about the system and result in an end-user experience that is far from ideal. My dissertation research shows that these sacrifices are not fundamental; stronger consistency and semantics are achievable for ALPS storage systems.

Yet, known theoretical results show that low latency and the strongest types of consistency are incompatible.<sup>1</sup> Knowing this, the recent spate of low-latency geo-replicated systems settled for the weakest semantic property, eventual consistency. The first part of my dissertation research, COPS, shows that this sacrifice is not necessary. COPS is the first ALPS system to provide causal consistency, a middle ground between the two extremes where operations always appear in an order consistent with potential causality, e.g., all of a user's operations and all conversations between users appear in their original order. This improvement in consistency makes the distributed storage more intuitive for programmers to use and gives end users more of the experience they expect.

One key technical contribution in COPS is a design focused on the scalability of clusters where the keyspace is partitioned across nodes, replication is done in parallel from all nodes in each cluster, and then nodes use dependency metadata associated with the updates to issue distributed checks that ensure operations are always applied in the correct causal order. Naively, dependency metadata grows exponentially and throttles performance. COPS avoids this problem using multiple types of garbage collection and by exploiting the transitive structure inherent in the graph of potential causality.

COPS's use of distributed metadata runs counter to the traditional wisdom for enforcing causal consistency, which was to exchange logs of operations and then replay them at other locations. The log-exchange approach works well and admits a simple implementation when all data can reside on a single machine. With the new realities of massive scale where data is spread across many machines, however, the log-exchange approach breaks down as logging updates to all machines in a cluster into one place becomes a bottleneck. In contrast, because of COPS's distributed design, it is the first scalable and causally consistent system.

The big data in ALPS systems is spread across thousands of nodes, yet previous systems provided only inconsistent batch operations to read and write data across multiple nodes. Eiger, the second part of my dissertation research, shows that much stronger semantics are possible. These stronger semantics include read-only and write-only transactions that consistently read or write data spread across all the nodes in a cluster. Eiger's semantics also include counter columns and the hierarchical column-family data model used in BigTable and Cassandra, which makes building applications atop it much simpler. The limited transactions and rich data model in Eiger do not come at the expense of high scalability or performance, and designing the system to ensure this was one of Eiger's main challenges.

In particular, to guarantee low latency Eiger must eschew locks and blocking, the typical techniques used for transactions. And to enable scaling it must avoid the centralization that is also typical for transactions. Eiger overcomes both these challenges using distributed algorithms that utilize logical-time-validity metadata and temporarily maintaining multiple versions of the data. In addition, its algorithms for read-only and write-only transactions are designed to work together using indirection to ensure that clients obtain a consistent, up-to-date view of the system and can atomically update data spread across many nodes.

My dissertation research shows that ALPS systems do not need to settle for eventual consistency and weak semantics. Taken together, Eiger and COPS show that causal consistency and stronger semantics are possible for low-latency geo-replicated storage.

---

<sup>1</sup>This incompatibility is an implication of Brewer's famed CAP theorem from 2000, which was formalized shortly after by Gilbert and Lynch. Its first proof, however, was a lesser-known result from 1988 by Lipton and Sandberg.



## Future Directions

### Fully-Distributed General Transactions

While working on transactional algorithms for the low-latency geo-replicated setting of COPS and Eiger, I began to design algorithms for fully-distributed general transactions. General transactions are widely recognized as easy for programmers to reason about and necessary for certain scenarios, e.g., banking. Due to the fundamental trade-off between strong consistency and low latency, they are hard to reason about in—and perhaps incompatible with—the geo-replicated and low-latency setting of COPS and Eiger. They are, however, compatible with a low-latency single-datacenter setting or with a non-low-latency geo-replicated setting.

Currently, transactions are either scalable or general, but not both. My research and Google's recent work on Spanner, which provides read-once-then-write transactions, are examples of the former. On the other hand, examples of the latter include traditional databases with general transactions that are restricted to a small subset (shard) of the data and/or are scheduled by a master node. Developing and verifying fully-distributed transactions across large numbers of nodes will bridge this divide and provide scalability for general transactions.

### Scalable Transport for Massively-Distributed Systems

Current transport-layer protocols, e.g., TCP and UDP, are ill-suited for massive-scale distributed systems. TCP was designed to provide a reliable stream of data between two machines; UDP was designed to provide unreliable datagrams in the same setting. In contrast, the communication patterns of massive-scale distributed systems typically have large numbers of parallel, asynchronous RPCs between many machines.

One example of this mismatch is that while running experiments for Eiger with hundreds of nodes, I found I could not achieve perfect linear scaling of throughput as cluster size increased due to the overhead from the increasing number of TCP connections on each node. As another example, I've learned from industrial colleagues that it is common practice to aggregate connections from multiple processes on the same machine to reduce connection overhead and to increase batching. Yet another example is Facebook's modification to memcached that uses TCP for writes, but uses UDP for reads so they can build their own retransmission protocol atop it that is aware of how they batch reads (multiget).

All of these issues stem from a mismatch between what current transport layers provide and how massive-scale distributed systems communicate. While there has been much recent work on improving TCP for datacenter usage—e.g., DCTCP that improves congestion control, or the entire "Data Centers: Latency" session at SIGCOMM 2012 that improved flow completion times—a more fundamental approach is necessary and possible. Datacenter services provide a rare opportunity to deploy a clean-slate transport layer because they are in a single administrative domain and can be upgraded en masse. Along with networking colleagues, I am interested in exploring what new transport layer properties, abstractions, and mechanisms can better match the performance or programmability requirements of massive-scale distributed systems.

### Making Programming Distributed Storage Make Sense

Strong consistency and general transactions, while incompatible with low-latency geo-replication, are well understood by programmers and easier for them to reason about than weaker consistency and limited transactions. My dissertation research starts to bridge this gap, but there is still much to do before programming massive-scale distributed storage truly "makes sense."

This introduces two exciting avenues of research. One is, how can we make it easier for

programmers to reason about and use low-latency primitives? My current approach has been to make the primitives as strong as possible and I believe this direction will bear more fruit. But, we will also need ways to make it easier for programmers to express themselves. Perhaps a query language will help? Or, a compiler that will deconstruct general transactions into limited ones?

The other avenue of research is, how can we present a single interface that is simple for programmers to reason about that provides access to strong-but-slow general transactions and fast-but-weaker limited transactions? Should transactions and their results be typed so we can reason about their use throughout a program? Would a domain-specific language help? Can we allow programmers to write the simplest code initially and then only specialize it if dictated by performance?

Each of these avenues provides an interesting mix of programming languages and distributed systems problems. I look forward to collaborating on them with PL colleagues and I believe solving them will have a large and lasting impact on the way web services are built and the way programmers interact with big data.

In the last 20 years the field of distributed systems has changed dramatically. The field moved from systems on the order of 10s of nodes in one administrative domain, to peer-to-peer systems with 1000s of nodes in many administrative domains, to datacenter services with a small number of datacenters each with 1000s of nodes within it that are back in a single administrative domain. With billions of edge devices that are increasingly capable, I am intrigued to see how they fit into the distributed systems of the future.

Increasingly, distributed systems problems have connections to networking, databases, programming languages, algorithms, and security. I look forward to working with colleagues in these areas in my future research career, as I believe that many of the most productive types of research come from inter-area collaboration.

# Teaching Statement

Wyatt Lloyd

Teaching is an important and exciting part of being a faculty member. I look forward to being involved in teaching at all levels, from sparking student interest in Computer Science in introductory courses to advising students pursuing their own research. I am qualified to teach introductory CS classes and particularly qualified to teach classes on systems and networking at the undergraduate level, the advanced/graduate level, and in seminars on more focused topics like distributed storage systems or datacenter networking.

I first gained teaching experience as a TA for Princeton's introductory course (COS 126). My favorite part of the course was teaching a twice-weekly precept to my section of about a dozen students. I focused on being enthusiastic about the material and keeping the class highly interactive. I believe the more you involve students and show them the interesting facets of a subject, the more they will be motivated to learn.

I also served as a TA for Princeton's networking course at the advanced undergraduate level (COS 461). The course had a strong focus on projects where the students built what we were learning about in class, e.g., a lightweight TCP implementation. This focus on building helped students understand the material more deeply, actively engaged them in the subject, and provided them with practical system building experience. The core parts of course were supplemented with discussions of recent research advancements, showing students our field is still evolving and that they can have an impact. This inspired several students to do research with our group, some of whom ultimately went to graduate school in systems and networking. The effectiveness of the class greatly impressed me, and I will incorporate a focus on building and discussion of research results into any advanced systems or networking course I teach.

Graduate courses offer an in-depth look at a topic and are an excellent vehicle for initiating research. I have found the format of lectures on a topic, followed by reading related papers, and then finally working on a research project to be a successful strategy. Lectures ensure a basic level of knowledge so that papers are accessible and framed by current and historical practices. Paper readings and discussions allow much more depth in exploring a topic as well as help students learn how to best focus their work and its exposition for maximum impact. This fall I had the privilege to guest teach the graduate level advanced networking class (COS 561) where I delivered a lecture with my enthusiastic and interactive style, followed by a discussion of two papers the students had read. This experience, combined with my earlier experience as a TA, confirmed that I enjoy and excel at teaching all levels of courses.

Research projects are an essential part of graduate courses because they give students the opportunity to explore a topic in greater depth, enabling them to make a potentially publishable contribution of their own. This is, in fact, how I published my first paper. An aspect of student projects that I will emphasize is building a prototype and experimentally verifying its design. I consider this an essential component of research in systems as it grounds their work in reality, gives them system-building experience, helps them focus on what is novel about their design, and can lead them to new areas of research.

I have advised one undergraduate on a semester-long research project and three other undergraduates on six-week summer projects. In each case, the experience was rewarding and I look forward to longer-term advisement of graduate students. I view advisement as a tremendous opportunity, and responsibility, to guide students down worthwhile paths without stifling their individuality or creativity. I am particularly excited to see how long-term collaboration will merge my ideas with those of my students into new research directions.