Research Statement

Vincent Liu
http://vincen.tl/

My research aims to question/expand our understanding of networked systems with the eventual goal of building better systems in the future. Networks like the ones that I study underpin much of today’s computing. Indeed, it is rare to find any modern application or service—from web search to driverless cars—that does not require some form of network connection. As they continue to grow in size, speed, and number, the design and implementation of networks must evolve accordingly.

My work takes a principled, but practical approach, and is inter-disciplinary by nature. At a high level, my work since graduate school falls into four main thrusts: (1) obtaining a complete view of networks, (2) predicting their behavior, (3) designing network-aware applications, and (4) investigating the implications of future cloud architectures. Rather than focusing on any one technique or area of computer science, my strategy is and has been to pursue “the biggest problem I can find, regardless of area.” Thus far, my pursuit of a better understanding and design of networked systems has required the inclusion of ideas from networks, operating/distributed systems, databases, deep learning, program synthesis, security, and human-computer interaction.

My attention to practical concerns has led to interactions and collaborations with Comcast, Facebook, Microsoft, Google, Cisco, Huawei, and VMWare. Facebook and VMWare have recognized the work with monetary awards, as has the NSF in the form of a CAREER award.

1 Obtaining a Complete View of the Network

Despite the foundational importance of large networks for so many current and future applications, a detailed and holistic understanding of these networks is often difficult to come by. This is problematic, as it is precisely their emergent properties that make large networks so difficult to handle. Instead, current measurement tools and techniques tend to be myopic; targeting only one device or path at a time. Network-wide metrics are typically a composition of these independent measurements. Neither provides a complete picture. This lack of visibility impacts every aspect of networking, from network operation (where operators need to understand overall behavior to debug issues) to network design (where architects need to understand workloads to build better networks) and the network protocols themselves (which need to react to current conditions). The behavior of large networks is inextricably tied to behavior that cannot be captured at any single location or path.

To highlight the importance of detailed, holistic network visibility, I designed and implemented a novel measurement tool that polled switch-level counters at μs granularity—orders of magnitude finer than previously possible. A colleague and I deployed this measurement tool in Facebook’s production data centers to study fine-grained behavior in the wild and its potential effects on current and future networks. Results were published in IMC ’17 [4]. Not only did we find evidence of microbursts (periods of high utilization lasting less than 1 ms), we found that nearly all congestion in their data center was due to these types of events. When compared to typical clock skews found in these systems, the implication is that, for any two measurements of any two network devices, the relationship between the measurements is both tenuous and difficult to bound.

In a SIGCOMM ’18 paper, my students and I thus sought to develop a novel, fine-grained, accurate, and precise measurement primitive that operates on the scale of an entire network [3].
The primitive that we were able to provide is that of a *Synchronized Network Snapshot*: a set of local measurements that together provide a coherent image of the entire network data plane at nearly a single point in time. Using this primitive, we were able to observe directly two network phenomena whose presence is difficult to detect using traditional methods: µs-level load balancing differences and fine-grained synchronization of application traffic in graph processing applications. Our current implementation guarantees a drift of at most tens of microseconds (less than a single RTT in most cases); drift can be decreased further using more advanced time synchronization techniques.

My students and I are currently building on the above work by developing a set of network tools that take an explicitly holistic approach. One line of work focuses on helping operators to design, understand, and optimize their networks. Rather than analyze the behavior of individual devices or network flows, what we aim to build is a *profiler* for network-wide traffic patterns. Just as traditional application profilers like gprof, Oprofile, and Valgrind have, for decades, helped programmers visualize how their programs spend their time and which control flows contribute, our system, *tpprof*, aims to reveal to network operators how their networks spend their time and the manners in which their network-wide traffic change over time. The other line of work takes a more automated approach. In collaboration with Microsoft Research, we are working on developing a scalable framework for the runtime verification of distributed, network-wide invariants. Many of the same challenges of network-wide consistency apply to this problem as well.

## 2 Predicting the Behavior of Networks

In addition to trying to understand the behavior of networks, my students and I have also been working on predicting it using machine learning (ML). Robust models and predictions of network behavior can enable us to better ask questions about future behavior and automate/optimize many critical tasks. In this direction, we have investigated methods of modeling the packet-level behavior of large networks, i.e., given a set of packets entering a network, is it possible to predict if/when the packets will egress? This problem turns out to be a very difficult one: while existing machine learning techniques can roughly approximate network behavior, some of the more subtle effects of networks are challenging for existing ML techniques. These include diverse multi-scale dependencies, compounding errors, and a need for generalizability across traffic patterns and network configurations among others. To encourage inclusive and interdisciplinary work on these challenges, my group, in collaboration with Prof. Lyle Ungar and his student, are currently working to present the problem of network modeling as a grand challenge problem for the ML community.

One interesting use of these techniques is our proposed techniques for taking observations of smaller networks and, combined with domain knowledge, extrapolating the approximate behavior of much larger ones. My group and our ML collaborators have designed a simulation framework called *MimicNet* that does exactly this for data center networks. Preliminary results were published in HotNets ’18 [2], with a full version under submission. The key observation behind our approach is that, by identifying and exploiting natural symmetry in modern data center network architectures, we can efficiently compose models of smaller networks into larger simulations of networks. Our preliminary experiments have shown promising results—for a standard network configuration, MimicNet trained on a 2-cluster data center can extrapolate the FCT of flows in a 16-cluster data center to within 6% in the median case. Our work on bridging the gap between the networking and ML communities will only help to improve the efficacy of this approach.

In parallel with the above work on modeling networks, my students and I are also investigating the classification of network issues; this is a joint project with Microsoft Research and Harvard. The
goal of this work is to be able to accurately and automatically assign incoming customer complaints and alerts to a responsible party. The origin of this project was an observation, by researchers at Microsoft, that a straightforward application of a classifier to every new incoming issue was not producing acceptable results. Instead, our approach, as above, was to build a small collection of hand-tuned classifiers for a subset of the teams within the organization and compose them cleverly.

3 Software Architectures for Current and Future Cloud Networks

Modern big data systems are complex and multi-layered. Beyond just a set of algorithms (such as PageRank, clustering, or common neighbors), these systems span all layers of the stack, including the data analytics applications, the big data platforms that supply the programming interface, and the data center infrastructure that perform the actual computation and storage. Over time, these layers have become increasingly blurred. This is particularly evident in single-tenant data centers where custom-built servers/switches are connected into engineered network topologies with the help of heavily tuned network protocols. The recent interest in microservices and disaggregated hardware will only lead to tighter integration between the various layers that support users’ applications. To achieve good performance, fault tolerance, and manageability, each of these layers must be acutely aware of the properties and limitations of the others.

In light of these trends, I am currently investigating the utility of vertical integration in current and future cloud environments, as well as its limitations. Part of this work is understanding what applications need from the underlying infrastructure, so much of this work is in collaboration with a team of researchers, including Profs. Boon Thau Loo and Sebastian Angel of Penn, and Prof. Ang Chen at Rice. We are woking through a set of students, for all of whom I am a co-advisor. Regardless of the application, a common theme among this cross-layer approach lies in finding the correct tradeoff between performance and portability.

For today’s data centers, we are concentrating on one particularly impactful application class: data processing. One artifact of this investigation, GraphRex, was recently published in SIGMOD ’19 [5]. Through it, we looked into how graph processing systems should be adapted to the cloud network architectures. To programmers, GraphRex presents a simple and declarative interface; under the covers, a set of optimized operators bring speedups of $3 - 109 \times$ for a set of representative workloads. Another artifact of this investigation is libTMC [1], a modular networking stack designed to allow users to add or remove features from their transport layers on-demand.

Looking toward the future, we are planning on investigating, in parallel, two trends: microservices and resource disaggregation. The first, microservices take small functions and deploy them automatically to an elastic pool of containers that are configured, scaled, and load-balanced—all without user intervention. Uber, for instance, is built from over 300 such microservices, and public cloud providers like AWS and Google are beginning to roll out support for users to build their own microservices. We envision a world in which, rather than each developer creating their own, that there would be a marketplace of these microservices. There are many challenges involved including the correct combination of these services as well as the security, isolation, and accountability of the composition. The second, disaggregation, argues that, rather than having monolithic servers with integrated compute, memory, and storage, disaggregated data centers allow these resources to be deployed and scaled independently. This has tremendous benefits to the management and elasticity of data centers, but it also fundamentally changes the structure of computing. We are currently investigating the implications of these changes on application design.
References


