Forty Years of Teaching Computer Networks: A Retrospective

Shivendra Panwar
New York University
New York, USA
panwar@nyu.edu

Fraida Fund
New York University
New York, USA
ffund@nyu.edu

Abstract

We present a forty-year retrospective on teaching computer networks, focusing on a graduate course in an Electrical and Computer Engineering department that evolved from a lecture-based course into a lab-intensive course centered on hands-on experimentation. Over this period, the Internet became the substrate for everyday computing, cloud services, mobile systems, security, and AI infrastructure. We describe how the course content has evolved, and why many core networking ideas have remained stable despite changes in the field. We then examine how course infrastructure changed and how the delivery of the course has improved as a result. Finally, we discuss how the student population, student motivation, and student engagement may have shifted as networking has become a core systems area rather than a specialized frontier. We close with reflections on the role of networking education in the generative AI era.

CCS Concepts

• **Social and professional topics** → **Computing education**; • **Networks**;

Keywords

networking education, computer networking

ACM Reference Format:
Shivendra Panwar and Fraida Fund. 2026. Forty Years of Teaching Computer Networks: A Retrospective. In *Proceedings of SIGCOMM Education Workshop 2026: Networking Education for the AI Generation (A4NE '26).* ACM, New York, NY, USA, 10 pages.

1 Introduction

Over the last forty years, the Internet has evolved from a specialized research and commercial technology into the foundation for everyday computing, cloud services, mobile systems, and AI infrastructure. This expansion has made the subject substantially broader, while changes in how students learn and how employers evaluate graduates have altered the context in which it is taught. Students now approach technical material with a wider set of external resources, including search engines, online videos, and generative AI tools. At the same time, employers increasingly expect graduates to contribute quickly, with less time devoted to learning basic systems skills on the job.

In this paper, we reflect on our experiences teaching a graduate course in computer networks over this forty year period [1] with the goal of understanding how the goals, constraints, and methods of networking education have changed. The course has engaged thousands of students, many of whom have gone on to careers in networking research and industry. It has evolved from a lecture-based course to one that is heavily focused on hands-on application and experimentation, and has adapted to changes in the field and in student needs.

This retrospective paper is organized as follows. First, in Section 2, we describe the original motivation for the course and the context in which it is offered. In Section 3, we discuss how the networking material itself has evolved over four decades, and why the durable parts have remained central. Following this, in Section 4, we describe the evolution of the course infrastructure and educational technology, where the most substantial changes have occurred.[2] Next, in Section 5, we consider how students' expectations and help-seeking behavior have changed, especially in the presence of generative AI. Finally, in Section 6, we conclude with reflections on the role of networking education in the generative AI era.

2 Context

When the graduate computer networks course was first offered by the Electrical and Computer Engineering department at Brooklyn's Polytechnic University[3] in the early 1980s, it was a lecture-only survey titled "Principles of Data Communication Networks." The catalog description from

A4NE '26,

[1] The first author has been involved with the course across the full forty-year period; the second author's experience covers the most recent decade.
[2] The lab materials and practice problems are available as open educational resources at: https://ffund.github.io/tcp-ip-essentials/.
[3] Polytechnic University was integrated into New York University in 2014, and is now NYU Tandon School of Engineering.

that era mentions terminals, modems, multiplexers, satellite and packet radio. At the time, the Internet was a research network. Textbooks and lectures were intentionally protocol-agnostic, focusing instead on general principles, because the field was still in flux. TCP/IP was not yet the dominant standard, and the number of engineers who understood how packet networks worked was small. A graduate student who could explain Dijkstra's algorithm, packet routing, and stop-and-wait protocols had distinctive, marketable knowledge.

Through the late 1980s and into the 1990s, this began to change. The commercialization of the Internet brought networking out of research laboratories and into banks, corporations, and eventually homes. As TCP/IP emerged as the dominant protocol stack (displacing competing networking ecosystems from IBM, DEC, and the telephone industry) the body of knowledge that students needed to master became both more concrete and more demanding. Employers were hiring network engineers, not just people who understood how networks worked in theory. They needed graduates who could troubleshoot a live network, interpret packet traces, and configure network devices. Commercial network management tools were proliferating, and the gap between textbook knowledge and operational practice was widening.

This reshaped the course in two ways. First, it pushed the course content toward TCP/IP specifically: in the 1990s the catalog description dropped the generic framing of earlier years and named protocols such as IP, TCP, ARP, ICMP, and UDP directly. Second, and more consequentially, it motivated the introduction of a laboratory component. Learning network protocols from a textbook was dry, and did not offer opportunities to practice the practical networking skills that were increasingly required of graduates. Meanwhile, Stevens' *TCP/IP Illustrated* [25] showed the value of getting into what protocols actually do under the hood, with practical packet analysis and real-world examples. This inspired the addition of labs - one or two per semester at first, added gradually, in part because desktop PCs and routers were expensive. By the end of the 1990s, a standalone lab-intensive course "Protocols for Local Area Networks," had been added to the course catalog, offering hands-on experiments covering the full TCP/IP stack. Students worked in groups of eight, with each student operating one PC and each group sharing a router to create various network scenarios. Eventually, the lecture-only prerequisite was quietly eliminated: most students wanted to go straight to the hands-on course.

By the early 2000s, many networking educators were making similar arguments. At the first ACM SIGCOMM education workshop in 2002, a major theme was the design of new hands-on laboratory experiences for computer networks [1, 19]. These examples point to the same conclusion that motivated our course: networking education needed a strong hands-on component, both to increase student interest and to support deeper understanding of networking principles. With demand for lab-based networking instruction growing, in 2004 the first author published *TCP/IP Essentials: A Lab-Based Approach* [21], a textbook based on the Polytechnic University course materials and laboratory exercises. Hands-on labs, in which students constructed networks and ran experiments in controlled settings, were the central networking education innovation of that era.

As the field matured, networking became embedded in nearly every industry. Our lab-based course (which was renamed from "Protocols for Local Area Networks" to "Internet Architecture and Protocols") grew into one of the most popular in the program. Through the 2000s and 2010s, a majority of ECE MS students enrolled in it, drawn by the expectation that networking expertise was broadly applicable - to internet companies, to financial institutions, to enterprise IT. The course had evolved from a specialist elective into something closer to a core offering. The broader networking education community was similarly concerned with preparing students to engage with networking both as a technology embedded in other domains, and as a research field. This is evident in the papers presented at the 2011 ACM SIGCOMM education workshop, which discuss how networking can be taught as a cross-disciplinary subject, taught with business context, taught with security considerations, and taught as preparation for networking research careers[2].

The growth of the Internet also brought the next major push to reshape the course: the need to accommodate online delivery. While the course was available to online students from the mid-2000s, for many years the lab component for remote students was improvised. Online students were given remote desktop access to a physical station in the on-campus lab, they would communicate over Skype with their in-person lab group while performing the lab in a coordinated, synchronous manner, and a teaching assistant would plug and unplug Ethernet cables on their behalf. This arrangement was fragile and did not scale well. The lab was adapted to a fully online format on the GENI testbed [6] in 2017, initially to accommodate a single online student whose time zone prevented synchronous participation with in-person students. Following this successful adaptation, by 2018, all online students - still a small number each semester - were completing the labs on GENI.

For many in the networking education community, the COVID pandemic in 2020 was the first time they had to confront the problem of how to support networking education in an online setting without reducing the course to passive lecture delivery [3]. For us, the sudden pandemic-imposed transition to fully online instruction - involving 140 students across ten lab sections - was relatively painless thanks to

the existing infrastructure, materials, and knowledge we had developed for online students over the previous few years.

Even after in-person instruction resumed, we kept the lab online. The online version worked well, and it eased a long-standing scheduling problem: fitting large numbers of students into four-hour sessions in a physical lab configuration that could accommodate only 16 students at a time.

Today, the course is a moderately popular elective rather than a near-universal one. Networking knowledge is now so widely diffused that what was once specialist expertise is now baseline literacy. The Internet has become core infrastructure that everyone depends on but few bother to study in depth. The role of the course and the motivation of students taking it have therefore shifted. Earlier versions of the course offered access to knowledge that was difficult to acquire elsewhere. Today, students can find explanations of networking concepts from many sources. Against that backdrop, networking education has refocused on key concepts that are less easily picked up informally - the principles of protocol design, the reasoning behind layering and end-to-end arguments, the ability to diagnose failures and understand tradeoffs. The value of the course now lies in promoting a deeper understanding of the underlying fundamental ideas.

> *Takeaway:* Networking education has shifted from teaching scarce protocol knowledge, to helping students build judgment and intuition about networked systems.

3 Evolution of the subject

Despite the changes in computer networks over the last forty years, much of the core material has remained surprisingly durable: while we have made modern additions and changes, they are modest relative to the content that has persisted. There are three main reasons for this stability. First, original versions of protocols are often still better for teaching design principles than their modern counterparts, so we may retain protocols in the course material even if they are no longer widely deployed. Second, many of the core ideas are unchanged, even when new protocols apply them in different ways. Third, while we can no longer teach all of the details students may need in their later work, students now have many more resources for acquiring those details as needed.

The first reason reflects a broader dilemma in systems education: whether to teach the classic form of a technology or the contemporary form. The classic form is often simpler and better suited to exposing the underlying idea, while the contemporary form is often more relevant, but also more complex. ASCII, for example, gives students the basic idea of characters-as-numbers: 128 characters, covering the English alphabet, numbers, and basic punctuation, with each character represented in one byte. Modern text is encoded using UTF-8, but teaching UTF-8 in full would require students to deal with 1.1 million code points, variable-length encodings, combining characters, normalization, and other details before they have learned the basic idea. So although ASCII is no longer the modern standard, it is still taught in detail because it illustrates the useful and durable principle [8]. Similarly, classic versions of network protocols often make design principles clear in ways that the versions deployed on the Internet no longer do. For example, additive increase and multiplicative decrease gives students a clear model of congestion control, while modern variants like TCP CUBIC with nonlinear window growth rules can obscure the basic idea. So, although TCP Reno is no longer the dominant TCP variant on the Internet, it is still taught in detail because it illustrates the useful and durable principle.

With respect to the second reason, the key ideas of the course - separation of roles in the protocol stack, tradeoffs between distributed and centralized control, use of headers to carry information between layers and across administrative boundaries, and the need to share resources among many independent users - remain as relevant today as they were decades ago, although they are realized in new ways in newer protocols. For students encountering these topics for the first time, the main challenge is to grasp these fundamental ideas; then the protocol details make sense in context.

For example, we often see students arrive at the answer one would get by solving a spanning tree or routing problem with a global view of the network, rather than the answer that a distributed protocol will reach. The student can see the whole topology on the page, so it is natural for them to reason as if every router or switch can also see the whole topology, rather than acting on local information and messages from neighbors. The difficulty is not in understanding protocol details, but in grasping an underlying model of network behavior that often runs counter to their initial intuition. We therefore continue to emphasize these core ideas.

Once students have that foundation of core ideas, however, they can learn many of the more detailed or more recent changes incrementally on their own. For example, students who understand TCP and UDP can grasp the motivation for and ideas behind QUIC. Similarly, even with the softwarization of networks, the interface to the network has changed but the underlying abstractions have not: students still need to reason about addresses, paths, and resource allocation.

Incidentally, we note that layering, end-to-end design, interoperability, fault tolerance, naming, addressing, resource sharing, and incremental deployment are not only networking topics; they are central ideas in computer systems. This is another reason why the focus of the course remains on these ideas: they are not just relevant to networking, but to many other areas of computer systems, and they are fundamental principles that students can apply in many contexts.

This emphasis also fits the course's position in an Electrical and Computer Engineering program. Many communications courses in this department focus on the physical layer, so the computer networks course takes a bottom-up approach from lower-layer mechanisms toward the higher-layer protocols needed to provide end-to-end communication. The goal is not to cover the newest version of every protocol, but to give students the conceptual structure they need to understand how the larger system supports end-to-end communication.

Finally, we note that earlier versions of the course could try to teach most of the protocol details students were likely to need for their work. Today, the field is too broad for that approach, and students also have gained access to tools to pick up new details as needed, first through search engines and online documentation, and later through generative AI.

Taken together, these reasons explain why the content of the course has remained remarkably stable even as the Internet itself has evolved almost beyond recognition.

> *Takeaway:* The most durable networking topics are not specific protocol details, but the abstractions they expose: layering, naming, addressing, resource sharing, failure, distributed control, and incremental deployment.

While the key ideas have endured, some parts of the course do change. The lab material requires frequent updates as the standard tools used in the field evolve (e.g., `net-tools` → `iproute`), and it has also been updated where necessary to introduce new key ideas. For example, to understand modern congestion control, students should understand how congestion control affects queuing latency (not just throughput, which is the main focus of the classic TCP congestion control material). They also need to understand how protocol heterogeneity affects coexistence at shared bottlenecks, which was less of a concern when all deployed TCP was essentially Reno with small variations. Therefore, the lab now includes content on TCP Vegas [7], TCP CUBIC [13], TCP BBR [10], and ECN [22], with an emphasis on those new ideas. Similarly, the network security material (which initially was non-existent!) has evolved substantially as relevant threat models and best practices have changed. These updates do not change the main conceptual arc of the course, but they keep the hands-on work aligned with current practice.

The lecture material also incorporates small pieces of current research, swapped in and out over time, as a supplement to the main ideas. Some of the most exciting additions come from the students themselves noticing that what they observe in the lab does not match the textbook model. For example, in one TCP experiment, students themselves noticed that retransmissions occurred before three duplicate ACKs were received, contrary to the classic "triple duplicate ACK" rule described in standard textbooks. We added a discussion of modern TCP retransmission behavior, including RACK [11], to explain why the deployed protocol behavior does not match the textbook. While we cannot cover all of the modern protocol changes in the lectures, when students discover them organically, it is exciting to turn that discovery into a discussion of how the field has evolved and why.

4 Evolution of course infrastructure

Over the last forty years, the course infrastructure changed along three dimensions: the infrastructure for the hands-on lab experiments, the infrastructure for out-of-lab practice and formative assessment, and the infrastructure for summative assessment. We discuss these changes in this section.

First, the infrastructure for hands-on lab experiments. In the early years, each desktop PC or router required a small capital request, so the introduction of lab material depended on the pace at which we could assemble enough equipment. As of the mid-2000s, the estimated budget for the equiment required to support an 8-seat lab was $17,500. Eventually, the lab grew to 16 seats, organized in two groups of eight, with routers and hubs that could be used to organize each group of PCs into various network topologies. The original experiments ran on Sun workstations with Solaris, but the lab later moved to Red Hat Linux, and then, Ubuntu Linux.

Lab sessions were scheduled in four-hour blocks, and students worked in groups of eight, coordinating their machines to orchestrate various network scenarios, with a teaching assistant in the room to advise and to help troubleshoot. At this point, the main infrastructure challenge was scheduling: we could accommodate only 16 students in each weekly 4-hour block, which became a bottleneck as enrollment grew.

The next major infrastructure change moved the lab onto the GENI testbed. In Fall 2017, we used GENI for a single online student. After that experiment succeeded, we created a regular online lab section for students enrolled in the online degree program; those students completed the same labs on GENI, but at small scale. We scaled this model in Spring 2020 in response to COVID, when 140 students were split across ten lab sections. Because the course already had instructions for the same lab assignments on GENI, the transition was manageable. Teaching assistants who had been responsible for in-person lab sessions hosted video conference calls during which students could complete the lab, or students could work asynchronously because the GENI version let each student coordinate the whole topology.

Large-scale online delivery still required additional student support infrastructure. As more students completed the labs asynchronously, the course eventually eliminated the synchronous video calls because students stopped attending them. To support asynchronous work, we provided students

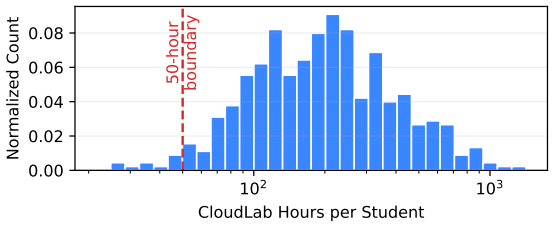


Figure 1: CloudLab resource-hours used per student.

with recorded videos of an instructor completing each lab, so students could watch the procedure while working through their own experiments. The course also added a Q&A forum (first on Piazza, later on Ed) where students could ask course staff for help. In the first year of large-scale online delivery, there were more than 2,500 forum posts (questions, comments, and answers), almost 9 posts per student on average.

Moving to GENI eliminated much of the infrastructure bottleneck we had with the local lab, but it did not remove infrastructure constraints entirely. The educational workload is highly bursty: many students worked on the lab shortly before the deadline, creating high peak demand for testbed resources. Although GENI was national-scale infrastructure, the course enrollment was large enough, and some experiments were large enough, to impose non-negligible load. The course staff also had less control over the infrastructure, which reduces the maintenance burden, but can also create pain points. For example, if a particular GENI site was broken, course staff could notify operators and students, but could not temporarily remove that site from the user interface, the way we might have marked a desktop PC in the in-person lab as out of service. This caused some frustration for students, and additional overhead for course staff who had to mediate between students and GENI operators to resolve issues.

The course also exposed less obvious scale limits. In 2022, the course instructor began encountering an error message on each login to the GENI Portal. GENI operators traced the error to an inefficient query that failed for users whose accounts were associated with many GENI "slices" (experiments). As of February 2022, the lab instructor had supervised 52,527 slices, and so became the first GENI user to trigger this particular failure.

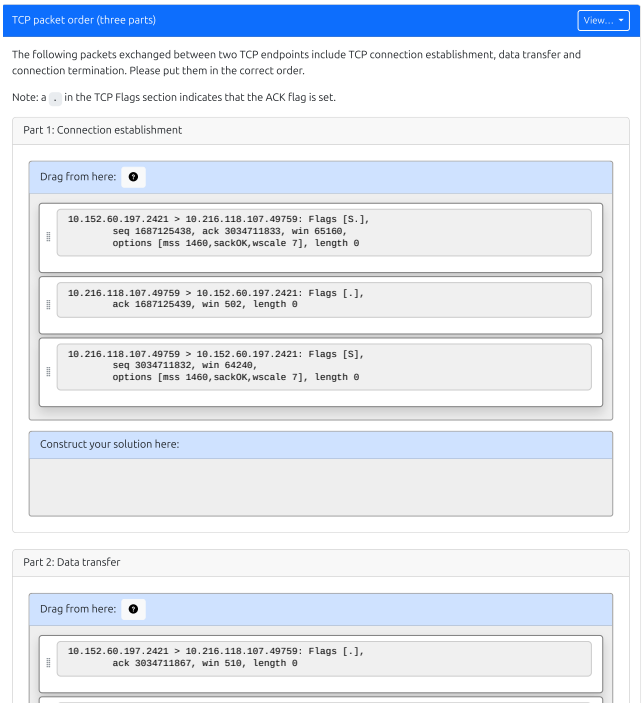
After COVID restrictions ended, we kept the labs online because the format had benefits beyond emergency remote instruction. Students could work at their own pace, rather than slowing down or speeding up to match the rest of a group in a fixed four-hour session. They also had access to each experiment over a longer period. As shown in Figure 1, testbed usage data from the last few years shows that students kept resources active for a mean of 236 hours per student per semester, and more than 92% exceeded the 50 hours they would have been allotted in the traditional in-person format. This does not mean that students spent much more time actively working on the lab; some students may have spent more active time, and some may have spent less. It does show that the online format gave them more flexibility in how they managed each experiment and their time.

The online format also changed what students could see and what the labs could ask them to do. In the physical lab, each student operated one PC within a group topology. Online, each student could inspect and control the whole topology, so every student saw every part of the experiment. This also made it possible to redesign labs that had been constrained by the physical room, the eight-PC group structure, and the available routers and hubs. For example, the online version could include a larger-scale multicast experiment than the local lab had been able to support.

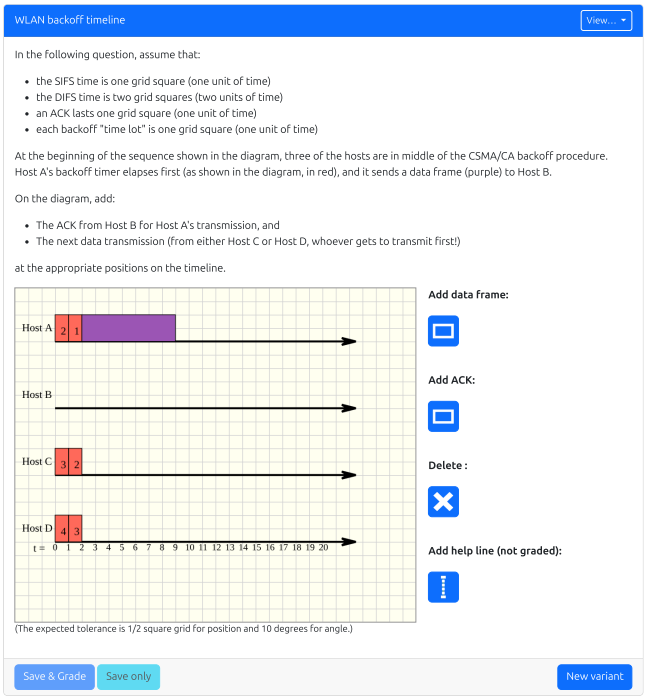
The next major transition came in 2023, in preparation for the deprecation of the GENI research testbed. Deprecation is a known risk of relying on non-local infrastructure, especially publicly funded research infrastructure, which tends to have a limited lifetime. To reduce this risk, we adapted the lab materials to run on any of the CloudLab [12], Chameleon [18], and FABRIC [4] testbeds. We currently use CloudLab as the lab platform for our graduate computer networks class, and since September 2023, students in our computer networks course have run 14,480 experiments on CloudLab, with a total cumulative duration of 111,785 hours. However, we maintain materials for all three platforms to support a broad goal of platform portability. Teaching materials that are tightly coupled to a single platform are harder to reuse, while materials that can run across platforms are also easier to adapt for different student populations, instructor backgrounds, and learning goals.

> *Takeaway:* Online labs removed the fixed-session bottleneck while also improving the pedagogy: each student could control a complete topology, work over a longer time window, and repeat experiments independently.

The second dimension of infrastructure change was in out-of-lab practice and formative assessment. By formative assessment, we mean low-stakes practice that gives students feedback and allows them to course-correct while they are still learning the material, rather than evaluating them only after instruction is complete. Earlier versions of the course used PDF problem sets, which gave students some limited practice: these had approximately 18 problems in total, with some problems having multiple parts. Solutions were released one week after the problem sets so students could check their work and review an instructor-provided solution. This format reinforced students' natural default of passive rereading before exams as a primary studying method [5], even though retrieval practice (actively recalling material from



(a) Packet ordering.



(b) CSMA/CA backoff.

Figure 2: Examples of auto-graded PrairieLearn question formats used for networking practice, illustrating richer interactions than standard multiple choice or numeric inputs: arranging packets in order (a) and annotating a CSMA/CA backoff timeline by adding lines and shapes (b).

memory), and distributed practice (spreading practice across time rather than cramming just before an exam) are more effective study strategies that have been shown to support more durable learning [17, 24].

As infrastructure became available to support better formats, we took advantage of it. In 2022, we moved practice problem sets to PrairieLearn, an open-source assessment platform that supports algorithmically generated questions, automatic grading, individualized feedback, and repeated attempts [20, 27] with a wide variety of question types. With this platform, we were not restricted to multiple choice, matching, and numeric input questions. We could generate and auto-grade questions using much more sophisticated input types, as illustrated in Figure 2. For example, we could have students drag and drop packets in blocks to show what packets would appear on a network, and in what order, in given circumstances (Figure 2a). We could have them add to a diagram or drawing to show the CSMA/CA backoff procedure on a timeline view (Figure 2b). We could have them configure a hypothetical home router to expose public-Internet-facing services behind a NAT, by entering realistic values into an interface copied from an actual home router. We could have them design a network or subnetting scheme to meet a set of requirements, then generate an explanation including both text and a personalized visual representation of the submission to explain why it did or did not meet requirements (e.g. to show that subnets were too small, or not disjoint, or not aligned on CIDR boundaries). Furthermore, we could generate realistic randomized variants of all of these network scenarios, including different addresses, topologies, configurations, or packet trace evidence, and then grade the student's answer against that specific variant. This makes repeated attempts without penalty meaningful: when a student gets something wrong, they can read an explanation tied to the specific variant, and then immediately try a new version of the problem. With their expanded role in the course, the practice problems now serve a role similar to the labs. They push students to understand and apply the mechanics of a network concept before using it in a larger experiment.

The move from PDF problem sets to interactive problem sets on PrairieLearn changed practice from a small number of static problems into a continuous system for retrieval practice and self-correction. The submission records for our course on PrairieLearn support our own observations that students use this capability for real practice. On homework problems that are required for credit, the average number of

submissions per student per problem is close to 2, indicating that students are using the opportunity to make multiple attempts and to learn from feedback. We also observe that most students take advantage of optional practice problems, with the average student submitting approximately 60 "extra" problems that are not required for credit, per semester.

This shift also changed the structure of the practice. The number of practice problems available to students has increased 10x, from 18 to 180, including many small scaffolded questions and optional questions that are not counted toward the course grade. This makes it easier for students to fill gaps in their knowledge. For example, a student who struggles with subnet design can first do extra practice on binary representations of dotted decimal IP addresses, basic subnet arithmetic, and other smaller skills before moving to more open-ended subnet design problems. A student who does not need that extra scaffolding can skip it.

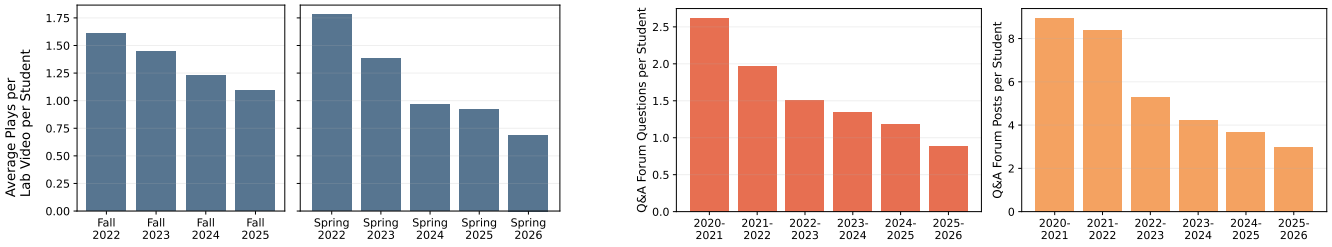
Although we have not undertaken any systematic investigation, our observations and platform data suggest that students are using the practice problems in a way that supports learning. For selected questions that had historically been associated with poor performance on related exam questions, we observe that students who submit more practice problems of that type tended to perform better on the related exam question. For example, for exam questions where students execute Dijkstra's algorithm on a small network, we saw a statistically significant positive association between the number of submissions a student makes on Dijkstra practice questions and their score on the associated exam question. Similarly, for subnet design questions, we observed a statistically significant positive association between the number of submissions on subnet design practice questions and the score on the associated exam question. This effect persists even if we control for the number of homework problem submissions in general, suggesting that it is not just that more diligent students do more practice and also do better on the exam, but that doing more practice on a particular topic is sometimes associated with better performance on the related exam question. These observations suggest that students are using the practice problems in a way that supports learning, although we cannot establish causality from these associations.

> *Takeaway:* Randomized variants, topic-specific feedback, and repeated attempts let students find and fill gaps in their own understanding.

PrairieLearn has substantially shortened the feedback loop for practice problem sets, and we are now exploring whether it can do the same for lab assignments. Unlike practice problems, labs are more open-ended and have traditionally been graded by human graders using a rubric, since they cannot be checked fully by algorithmic tests. PrairieLearn now supports AI-assisted grading using the same rubric: the grader marks specific rubric items and exposes those marks to students, rather than surfacing raw model output or inventing a separate grading scheme. This makes it possible to compare the AI grader systematically with human graders, measure agreement at the rubric-item level, and revise the grader instructions to improve that agreement [28]. We are currently piloting this approach, with AI and human graders working in parallel on the same lab submissions, to evaluate its feasibility and reliability. If successful, this approach could further shorten the feedback loop for labs, giving students more timely feedback on their work and freeing up human graders to focus on edge cases.

The third dimension of infrastructure change was in summative assessment. The course originally used pen-and-paper exams. By summative assessment, we mean higher-stakes evaluation used to judge what students have learned after instruction and practice. Historically, pen-and-paper exams were the default because they supported a wide range of question types, including qualitative questions that asked students to explain their reasoning in writing, and quantitative questions that asked students to perform calculations. Computer-based exams on the university's learning management system only supported auto-grading for multiple-choice questions or very simple numeric responses, so there was no compelling reason to use them. As more of the assessment material moved into PrairieLearn, which supports a wide range of rich auto-graded question types alongside manual grading for open-ended qualitative questions, we also started using it for exams. Initially, we administered exams in a proctored setting with students using their own laptops. However, as generative AI tools became more powerful and more widely available, it became increasingly difficult to secure students' own laptops for exams. Therefore, in 2025 we moved to a computer-based testing facility [29, 30]. Students still take computer-based exams in an in-person proctored environment, but now they work on department-administered machines that are locked down to prevent access to unauthorized resources.

The main implementation challenge was space. The department did not have a dedicated computer-based testing facility, nor the space to create one. Instead, we configured a 25-seat instructional computer lab used by other courses in the department as a dual-use space. During regular use for instruction, the machines had unrestricted network access. During exam use, a lightweight configuration switched the machines into an exam environment, in which a network proxy enforced whitelist-only access to network resources. This made the transformation temporary and reversible: the same room could support ordinary instruction, then become



(a) Video plays per lab video per student by semester (Fall and Spring subpanels).

(b) Q&A forum questions and posts per student by academic year.

Figure 3: Student engagement metrics from course video analytics and Q&A activity.

a proctored testing room with tightly controlled network access for scheduled exam sessions.

Scheduling could potentially be a bottleneck for this dual-use approach, but we were able to address this barrier. With only 25 seats, the facility could not necessarily accommodate an entire class section at once, and given the dual use nature of the space, it would not necessarily always be available at the regularly scheduled meeting time for our course sections. The scheduling challenge was especially acute for online sections. Most students in the online sections still took exams in person, with exceptions for students enrolled in fully online programs or otherwise unable to come to campus. However, these students were enrolled in an asynchronous section, so they did not necessarily share a common class meeting time when they could all take an exam. Fortunately, other courses in the department were beginning to adopt computer-based testing at the same time. These courses pooled facilities, proctors, and exam sessions; multiple instructors share the same room and proctoring staff, so students from different courses can sit in the same testing session while each taking their own exam. Pooled exam sessions across courses made the coordination of exams relatively painless, and PrairieTest, a sibling system to PrairieLearn, manages this scheduling and exam administration layer.

5 Evolution of students

Students' interest in the course has changed along with the status of networking itself. In the early years of the course, networking was a specialized and rapidly expanding area, and the material could be directly career-defining: students who understood TCP/IP, routing, and protocol behavior had access to expertise that was scarce in industry. Today, networking is still important, but it is less often the hot topic that draws students on its own. For many students, it has become a core area of systems knowledge that they need in order to work on something else: cloud computing, cybersecurity, cellular systems, or AI infrastructure. In that sense, the motivation has shifted from "I want to become a networking specialist" toward "I need to understand networks well enough to build modern systems that rely on them." This changes the way students experience the course. With the availability of Internet content, search engines, and generative AI, students are more sophisticated consumers of instruction. They look for the value added by the course beyond what they can obtain independently, and the hands-on lab has become the central answer to that question. Whether that remains enough is an open question.

Students' help-seeking behavior also appears to be changing. When the course moved online at scale, recorded lab walkthroughs and the Q&A forum became important substitutes for in-person help, but Figure 3 shows that students are now making less use of both instructor-recorded lab videos and the course Q&A forum. Some of this may be attributed to reduced motivation in the subject more generally. It is also possible that students are increasingly seeking help outside the course staff, especially from generative AI systems, a pattern that has been observed in recent work on AI and student learning communities [14]. This change is not necessarily negative: students may be finding answers more efficiently and reserving staff attention for harder questions. But it also raises the risk that they are bypassing the social and diagnostic parts of help-seeking that reveal misconceptions and provide instructors with valuable feedback.

> *Takeaway:* The trend of declining visible engagement suggests that that students may be moving help-seeking outside the course, toward search, peers, or generative AI. This may raise new challenges for instructors.

6 Reflections

We cannot yet predict how generative AI will change networking, education, or networking education. We are operating in the middle of that change. However, the current moment already suggests one instructional priority: students need to develop capabilities that remain valuable when AI

can produce explanations, code, configuration examples, and troubleshooting steps, and even execute them autonomously.

This priority aligns with a long-standing choice in the course, as discussed in Section 3, to emphasize the motivations and core ideas behind network protocol design: layering, naming, resource management, failure, and incremental deployment. Generative AI reinforces that choice. Today's students can retrieve, rather than recall, protocol details and tool syntax. But students still - increasingly - will be called on to provide judgment about networked systems and solutions for them, which requires a deep intuition into the core ideas and their application in networks.

Beyond questions about what capabilities our current students will need, AI also raises questions about how they will acquire those capabilities. The lower rungs of the career ladder in many computing fields are becoming increasingly difficult to access, as AI can perform much of the work that used to be done by entry-level workers [9]. Those who do find entry-level roles are often working with AI assistance, which can be a double-edged sword for skill formation [23]. There is a real risk that many students will not have the opportunity to build expertise and intuition through hands-on experience with real systems, which has traditionally been a key part of preparation for professional work. At the same time, effective use of AI appears to depend on that domain expertise [26]. Together, these findings suggest that AI rewards users with domain expertise, but it is not clear how students will acquire that expertise if AI reduces their access to the work through which earlier cohorts built it.

Our course has faced a version of this problem before. When a lecture alone became insufficient preparation for professional networking work (the gap between what students could learn from lecture and what they needed on the job had become too large), we added labs so that students could move from knowing concepts to working with systems. The AI era version of that gap is more demanding. Students may need to work with AI from the first day of a job, but they also need enough networking judgment to direct AI agents, reject bad directions, and notice missing context.

The course's movement toward more practice is a step in the right direction, although it is not a complete solution. The online labs give students more time with realistic systems and more opportunities for open-ended engagement than the old four-hour lab blocks allowed. The new approach to practice problems gives them many more opportunities to retrieve, apply, revise, and receive feedback on specific ideas before they encounter those ideas in a larger experiment. Optional scaffolding also lets students choose how much support they need while still requiring them to produce an answer. These structures give students repeated chances to form intuition, identify gaps, and correct misconceptions, which are some of the same skills they will need when they use AI as a professional tool.

The classic idea of T-shaped preparation is useful here. In the pre-Internet age, Johnston used the T-shaped metaphor to describe professionals who combine depth in one area with breadth across adjacent areas [15]. More recently, Kam et al. return to that framing for AI-enhanced computing, arguing that effective use of AI requires the kind of deep core expertise and broad adjacent-domain knowledge that has traditionally characterized more senior technical roles [16]. Networking already has this T shape. A network engineer needs depth in Internet architecture and protocols, but the field also touches hardware, electromagnetic propagation, operating systems, software engineering, security, optimization, queuing theory, economics, and operations. To prepare students for the AI era, a course in computer networks must still provide the long vertical part of the T, while exposing students to enough adjacent context to ask better questions.

This may be the strongest argument for preserving the hands-on parts of the course in the AI era. Students do not need a course to compete with AI as a source of definitions or syntax. The role of formal education in this field is to help them build a working mental model of networks, a sense of what can go wrong, and calibrated skepticism about fluent or sycophantic answers, to prepare students to use AI without outsourcing the reasoning that makes them useful engineers.

> *Takeaway:* In the AI era, networking courses should not compete with tools that can produce definitions, commands, and examples. Their role is to build the mental models and skepticism students need to use those tools without outsourcing the reasoning.

Acknowledgments

We gratefully acknowledge the many instructors and course assistants who have contributed to the courses through teaching, lab development, assessment, troubleshooting, and student support. We also thank the NSF-supported research testbeds that have made the course's hands-on networking laboratories possible, notably GENI and CloudLab. Finally, we thank the students whose questions, observations, struggles, and feedback have shaped the evolution of the course.

References

[1] 2002. ACM SIGCOMM Workshop on Computer Networking: Curriculum Designs and Educational Challenges. https://www-net.cs.umass.edu/sigcomm/education/workshop1.html August 20, 2002..

[2] 2011. 2011 ACM SIGCOMM Education Workshop. https://edusigcomm.info.ucl.ac.be/Workshop2011/Programme August 15, 2011..

[3] 2020. 2020 ACM SIGCOMM Education Workshop. https://gaia.cs.umass.edu/sigcomm_education_workshop_2020/white_papers.htm August 5-6, 2020..

[4] Ilya Baldin, Anita Nikolich, James Griffioen, Indermohan Inder S. Monga, Kuang-Ching Wang, Tom Lehman, and Paul Ruth. 2019. FABRIC: A National-Scale Programmable Experimental Network Infrastructure. *IEEE Internet Computing* 23, 6 (2019), 38–47. doi:10.1109/MIC.2019.2958545

[5] Aaron S. Benjamin and Jonathan Tullis. 2010. What makes distributed practice effective? *Cognitive Psychology* 61, 3 (2010), 228–247. doi:10.1016/j.cogpsych.2010.05.004

[6] Mark Berman, Jeffrey S. Chase, Lawrence Landweber, Akihiro Nakao, Max Ott, Dipankar Raychaudhuri, Robert Ricci, and Ivan Seskar. 2014. GENI: A federated testbed for innovative network experiments. *Computer Networks* 61 (2014), 5–23. doi:10.1016/j.bjp.2013.12.037 Special issue on Future Internet Testbeds – Part I.

[7] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. 1994. TCP Vegas. In *Proceedings of the Conference on Communications Architectures, Protocols and Applications (SIGCOMM '94)*. Association for Computing Machinery, New York, NY, USA, 24–35. doi:10.1145/190314.190317

[8] Neil Brown. 2012. Teaching Programming: Modern or Educational? Academic Computing. https://academiccomputing.wordpress.com/2012/11/30/teaching-programming-modern-or-educational/ Accessed: 2026-06-07.

[9] Erik Brynjolfsson, Bharat Chandar, and Ruyu Chen. 2025. *Canaries in the Coal Mine? Six Facts about the Recent Employment Effects of Artificial Intelligence.* Technical Report. Stanford Digital Economy Lab. https://digitaleconomy.stanford.edu/publications/canaries-in-the-coal-mine/

[10] Neal Cardwell, Yuchung Cheng, C. Stephen Gunn, Soheil Hassas Yeganeh, and Van Jacobson. 2016. BBR: Congestion-Based Congestion Control. *Queue* 14, 5 (2016), 20–53. doi:10.1145/3012426.3022184

[11] Yuchung Cheng, Neal Cardwell, Nandita Dukkipati, and Priyaranjan Jha. 2021. *The RACK-TLP Loss Detection Algorithm for TCP.* RFC 8985. Internet Engineering Task Force. doi:10.17487/RFC8985

[12] Dmitry Duplyakin, Robert Ricci, Aleksander Maricq, Gary Wong, Jonathon Duerig, Eric Eide, Leigh Stoller, Mike Hibler, David Johnson, Kirk Webb, Aditya Akella, Kuangching Wang, Glenn Ricart, Larry Landweber, Chip Elliott, Michael Zink, Emmanuel Cecchet, Snigdhaswin Kar, and Prabodh Mishra. 2019. The Design and Operation of Cloudlab. In *Proceedings of the 2019 USENIX Conference on Usenix Annual Technical Conference* (Renton, WA, USA) *(USENIX ATC '19)*. USENIX Association, USA, 1–14.

[13] Sangtae Ha, Injong Rhee, and Lisong Xu. 2008. CUBIC: A New TCP-Friendly High-Speed TCP Variant. *ACM SIGOPS Operating Systems Review* 42, 5 (2008), 64–74. doi:10.1145/1400097.1400105

[14] Irene Hou, Owen Man, Kate Hamilton, Srishty Muthusekaran, Jeffin Johnykutty, Leili Zadeh, and Stephen MacNeil. 2025. 'All Roads Lead to ChatGPT': How Generative AI is Eroding Social Interactions and Student Learning Communities. In *Proceedings of the 30th ACM Conference on Innovation and Technology in Computer Science Education V. 1* (Nijmegen, Netherlands) *(ITiCSE 2025)*. Association for Computing Machinery, New York, NY, USA, 79–85. doi:10.1145/3724363.3729024

[15] D. L. Johnston. 1978. Scientists Become Managers-The T-Shaped Man. *IEEE Engineering Management Review* 6, 3 (1978), 67–68. doi:10.1109/EMR.1978.4306682

[16] Matthew Kam, Cody Miller, Miaoxin Wang, Abey Tidwell, Irene A. Lee, Joyce Malyn-Smith, Beatriz Perret, Vikram Tiwari, Joshua Kenitzer, Andrew Macvean, and Erin Barrar. 2025. What do professional software developers need to know to succeed in an age of Artificial Intelligence?. In *Proceedings of the 33rd ACM International Conference on the Foundations of Software Engineering* (Clarion Hotel Trondheim, Trondheim, Norway) *(FSE Companion '25)*. Association for Computing Machinery, New York, NY, USA, 947–958. doi:10.1145/3696630.3727251

[17] Jeffrey D. Karpicke, Andrew C. Butler, and Henry L. Roediger III. 2009. Metacognitive strategies in student learning: Do students practise retrieval when they study on their own? *Memory* 17, 4 (2009), 471–479. doi:10.1080/09658210802647009

[18] Kate Keahey, Jason Anderson, Zhuo Zhen, Pierre Riteau, Paul Ruth, Dan Stanzione, Mert Cevik, Jacob Colleran, Haryadi S. Gunawi, Cody Hammock, Joe Mambretti, Alexander Barnes, François Halbach, Alex Rocha, and Joe Stubbs. 2020. Lessons Learned from the Chameleon Testbed. In *Proceedings of the 2020 USENIX Annual Technical Conference (USENIX ATC '20)*.

[19] Jim Kurose, Jörg Liebeherr, Shawn Ostermann, and Theresa Ott-Boisseau. 2002. Workshop Report: ACM SIGCOMM Workshop on Computer Networking: Curriculum Designs and Educational Challenges. August 20, 2002..

[20] Firas Moosvi, Dirk Eddelbuettel, Craig Zilles, Steven A. Wolfman, Fraida Fund, Laura K. Alford, and Jonatan Schroeder. 2023. Creating Algorithmically Generated Questions Using a Modern, Open-sourced, Online Platform: PrairieLearn. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 2* (Toronto ON, Canada) *(SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 1177. doi:10.1145/3545947.3569634

[21] Shivendra S. Panwar, Shiwen Mao, Jeong-dong Ryoo, and Yihan Li. 2004. *TCP/IP Essentials: A Lab-Based Approach.* Cambridge University Press, Cambridge. 288 pages.

[22] K. K. Ramakrishnan, Sally Floyd, and David L. Black. 2001. *The Addition of Explicit Congestion Notification (ECN) to IP.* RFC 3168. Internet Engineering Task Force. doi:10.17487/RFC3168

[23] Judy Hanwen Shen and Alex Tamkin. 2026. How AI Impacts Skill Formation. arXiv:2601.20245 [cs.CY] doi:10.48550/arXiv.2601.20245

[24] Amy M. Smith, Victoria A. Floerke, and Ayanna K. Thomas. 2016. Retrieval practice protects memory against acute stress. *Science* 354, 6315 (2016), 1046–1048. doi:10.1126/science.aah5067

[25] W. Richard Stevens. 1993. *TCP/IP illustrated (vol. 1): the protocols.* Addison-Wesley Longman Publishing Co., Inc., USA.

[26] Luca Vendraminelli, Matthew DosSantos DiSorbo, Annika Hildebrandt, Edward McFowland, III, Arvind Karunakaran, and Iavor Bojinov. 2025. *The GenAI Wall Effect: Examining the Limits to Horizontal Expertise Transfer Between Occupational Insiders and Outsiders.* Harvard Business School Technology & Operations Management Unit Working Paper 26-011. Harvard Business School. doi:10.2139/ssrn.5462694

[27] Matthew West, Geoffrey L. Herman, and Craig Zilles. 2015. PrairieLearn: Mastery-based Online Problem Solving with Adaptive Scoring and Recommendations Driven by Machine Learning. In *2015 ASEE Annual Conference & Exposition.* ASEE Conferences, Seattle, Washington. https://peer.asee.org/24575.

[28] Chenyan Zhao, Max Fowler, Yael Gertner, Seth Poulsen, Matthew West, and Mariana Silva. 2026. AI-Supported Grading and Rubric Refinement for Free Response Questions. In *Proceedings of the 57th ACM Technical Symposium on Computer Science Education V.1* (USA) *(SIGCSE TS 2026)*. Association for Computing Machinery, New York, NY, USA, 1207–1214. doi:10.1145/3770762.3772545

[29] Craig Zilles, Matthew West, Geoffrey Herman, and Timothy Bretl. 2019. Every University Should Have a Computer-Based Testing Facility. In *Proceedings of the 11th International Conference on Computer Supported Education - Volume 1: CSEDU.* INSTICC, SciTePress, 414–420. doi:10.5220/0007753304140420

[30] Craig Zilles, Matthew West, David Mussulman, and Tim Bretl. 2018. Making Testing Less Trying: Lessons Learned from Operating a Computer-Based Testing Facility. In *2018 IEEE Frontiers in Education Conference (FIE)*. 1–9. doi:10.1109/FIE.2018.8658551