Six Thoughts on Computer Networking Education and How to Leverage Its Nature

Sergey Madaminov
Stony Brook University, USA
smadaminov@cs.stonybrook.edu

ABSTRACT
This year the world got to learn how unpleasant and horrifying global pandemics can be. Numerous measures were taken to prevent the spread of a dangerous virus, including a shift to online education by many universities. That transition came abruptly and didn’t leave much room for preparation. Surely, some of the issues in exercising remote learning activities were due to that but not all of them. Among those others, a non-trivial subset of them was known, however, a complement of that subset appeared to be new or deemed not that important previously. While these issues are shared among many academic disciplines, this brief note tries to discuss them in the scope of computer science education in general and then narrow it further to computer networking education.

INTRODUCTION
In these extraordinary times, academic life and academic activities got quite intertwined with events and happenings outside of university walls. That put a substantial strain on people’s health and affected the way schooling is done. Many universities had to escape to remote teaching without thorough preparation. In the absence of such, professors were forced to quickly adapt their model and use an old scientific method of “trial-and-error” in an attempt to find what works and what does not. While multiple issues were caused by the abrupt shift in the educational model, the important question is what lessons and knowledge we can extract from this unfortunate event. Not all issues are new, many were known. There are also ones that were previously unknown or deemed non-important. Moreover, most are shared across computer science disciplines. This is an attempt to give a thought to some of them and how computer networking can approach them.

This brief note tries to suggest six thoughts for a discussion. In no way, they cover, even partially, the myriad of issues that are ahead of us on a way to shape a computer science education in what it should be, but rather hope to make a step towards that direction.

1 IF WE HAVE A QUESTION, SOMEBODY ELSE ALSO DOES
When we teach, we often say to students that one of the reasons to ask questions is that if they have it then somebody else probably also has the same question. This statement holds its truth beyond the classroom. Online learning was not invented yesterday and as a phenomenon appeared well before the pandemic. There are multiple platforms for professionally-oriented computer science learning including Linkedin Learning [9], Udemy [19], and Pluralsight [15], which target studying specific technologies, and platforms such as Coursera [2], MIT OpenCourseWare [11], and edX [3], which provide academic-style type courses. There are even online computer science degrees available with one from Georgia Tech being one of the most famous [5]. There is a great write-up on online learning [4] written by Nick Feamster who was involved in the latter one.

If we take a look at most of them, we can find that they have several common traits. One of the most notable is immediate feedback that a student can get after the submission. Modern in-person education system heavily relies on manual grading by teaching assistants or professors and submission systems that are not necessarily well-tailored for computer science albeit existing efforts to create one [16]. As classroom-based education continued traditional system of collecting and grading homework with some flavor of technology, online education had to design its courses in such a way that will allow auto-grading so it can scale with an ever-increasing number of students. To be fair, there are many computer science courses that implemented auto-grading and some form of immediate feedback to students, those include operating systems course at MIT, databases course at CMU, and cloud computing course at Stony Brook University. Notably, the feedback itself can be a powerful instrument in education, which is rarely used to its full potential. For example, studying mathematics involves not just solving numerous problems, but also seeing how they are solved and what approaches are there to take to solve them. But in computer science, that second step is often overlooked. Seldom we can see that students are provided feedback not only on the correctness of their code but also on its quality. Code reviews that are almost a standard in the industry are yet uncommon in institutions teaching computer science. Experience is gained through practice and it will not be too far from truth to say that students keep practicing writing a bad quality code. It could have been, at least, partially, avoided if students are provided with qualitative feedback on their work [8]. Thus, rethinking grading and feedback mechanisms are crucial for improving the way we teach and accumulated experience in other learning formats can become handy when trying to find a fitting solution.

During the process of integrating a feedback mechanism into the course and grading specifically, the important question to ask ourselves is to which extent provide feedback. Options range from leaving students in a dark or giving them the exact grade they will receive with a current submission and anything in between those two. That can be a discussion by itself and likely there is no-one-size-fits-all solution. Ultimately, it depends on a curriculum, pursued goals, and expected learning outcomes set by the course instructor. But we still can say that a good feedback mechanism will enable students to achieve some minimally expected level of progress. An important remark to make here, the feedback mechanism isn’t just about grading tools, but also includes constant communication.
with students through various media such as online forums, emails, and office hours. The availability of well-designed feedback system coupled with established and clearly communicated expectations, may provide some relief to the students by helping them understand where they currently stand in the course and even improve their performance in some cases [12, 20].

The nature of computer networks makes it adaptable to auto-grading and feedback. Huge amounts of network data leave no excuses for an inability to find data sets for homework. When processing network packets, modern computer networks provide functional behavior depending on the type of those packets. It allows us to verify actions taken by network devices irrespective of input data as long as it has a particular type and format. And altogether that eases auto-grading.

2 EQUAL ACCESS
For many computer science students, it is expected that they have access to computer devices. While many students have their own personal laptops, universities also may provide access to IT services and work stations. However, with remote education, the expectation that every student has stable access to a computer may not be upheld anymore. Not everyone has a stable internet connection, not everyone has a personal computer, and, even if they have, not all personal devices are powerful. When students partaking purely online courses, they are required to have access to stable internet and computer, but it is questionable if students enrolled at classic universities should be required to meet the same expectations.

Recent advances in cloud technologies and virtualization enable lowering the specification requirements of students’ computer devices. It is possible to create whole virtual networks using technologies such as VIRL [6], OVN [13], and mininet [10]. Public clouds can be utilized to greatly reduce specification requirements for students’ devices or can be used for other educational activities [18]. And it is also worth mentioning that many cloud providers have special programs for educators.

3 SOME ARE MORE THIRSTY THAN OTHERS
Not all students in the classroom are in the same situation. Some may have a full-time job to support family, some may have a very different background. And most of the students have varying skill set among them. Furthermore, not all students even want to be in the classroom but rather have to take that particular course as it may be a required course for a major. In a way, none of them should be punished for that. And it works both ways. Students who lack some skills should be provided the opportunity to learn and enjoy the course given they put enough effort, and students who are more versed in that particular topic should be given an opportunity to learn beyond curriculum.

This is a universal problem for many courses, not just computer networking or even computer science. However, it is something to always keep in mind. The material should be accessible to everyone in the classroom who is willing to work through it. But at the same time, there should be a component that will help to learn more advanced material and concepts whenever a student asks for it or you identify the one who may be interested.

4 AT THE END IT IS JUST A COURSE
Homework is stressful. During the rough times (such as current pandemic) it is even more stressful. Recently, more and more attention started being given to mental health. We all know that trying to do work while not being in the right state of mind can seriously hamper our productivity and ability to learn. Despite universities offering services for mental health and course instructors being explicit about greater importance of one’s health over the grades, students still need to take action to use such services or make an explicit effort of dropping out of the course. Moreover, some students could be from different countries and cultures, where mental health issues are taken very differently. Such students may be inadvertently reluctant to seek help. So what educators can do about it?

The goal of homework is to teach but not to impose fears. If student get stuck in the very beginning of the homework, it may affect self-esteem and ability to make good progress, thus, becoming detrimental to their learning [12]. One of the ways to tackle this problem is to introduce a series of “quick success” exercises [7]. The homework may start with simple enough tasks that can be as straightforward as going through provided steps or tutorial (and you need to make sure that tutorial works!). It will help students to see immediate progress and advance through the homework without being stuck on the first page of its description. Luckily, computer networking has plenty of such exercises! Even more, with carefully crafted set up, those exercises may look quite substantial, such as sending text messages between two virtual machines given that the virtual machines are configured and initial code is provided. We should keep repeating that a course at the end is just a letter on a piece of paper and taking one’s health is much more important. But that could be not enough. We also should make active steps to help students to feel successful, which is especially important when there is no support in the close physical distance or in-person guidance.

5 ENFORCE KNOWLEDGE, NOT PUNCTUALITY
Grades should reflect knowledge. Many courses have deadlines for homework, projects, or most of the other in-class assignments. However, does it really make a difference if a student submitted a homework one day later? Does it make student to know less? To some extent, those deadlines help to structure the course, but they may also result in assessing students’ punctuality in addition to academic merit. We do want to enforce knowledge but if we only rely on grades obtained through submitting on-time, we loose the opportunity of checking student’s knowledge later in the course. Thus, it may be important to be more flexible.

On the other hand, there also should be a clear set of what is expected to pass a course. Will you be comfortable with giving a passing grade to a student who got a 0% on an important topic? However, it will be not fair to give a failing grade based on a single assignment. A system that will require to pass a specific threshold on each assignment but will allow doing that even after the deadline may turn out to be a good approach. For example, allowing re-submissions [12], which will have to score at least 50% on an
assignment but will not affect the final grade anymore and keep con-
tinually provide feedback so a student can complete that required
part [1]. Additionally, if there are required readings they also fall
within enforcing knowledge and, thus, should be enforced. Other-
wise they are not required. That brings back clear expectations. If
something is required then it should be verified.

It is important to maintain realistic expectations when setting
such minimum requirements. While the bar should not be low-
ered just because more students happened to underperform, and,
similarly it should not be raised just because more students were
able to outperform it. That could be an indicator that expectations
were unreasonable or not well aligned with the level of students.
And if there is a strong sign that this may be a case and students
are doing poorly, then the grades should be curved to account for
such discrepancy. That will be an immediate remedy. Then comes
a second step, which is a reflection of why that happened. There
is nothing inherently wrong to give many good (or bad) grades,
but if a significant part of the class is not doing well it is usually a
sign that something in teaching does not work or the assignment
is ill-designed. And that may be a good time to introduce changes
to the course. In a way, the curve should only help students and
account for overestimated expectations to enable a collaborative
atmosphere and avoid creating a competitive environment [17]. We
want to ensure that students are given an opportunity to succeed
and don’t leave the class unprepared.

6 ARE WE RELEVANT?
Importantly, we want to teach relevant material. We want students
to be able to convert the textual problem into code. Computer
networking is a huge domain, but there are plenty of interesting
technologies that are important out there, in the wild. Those include
SDN, Open vSwitch [14], and many others. That is what is used
in industry and getting exposure and experience working with
them is important. Often, it is solved by having projects within the
course, which comes with two potential challenges. First, how to
identify interesting projects, which are interesting not only from
the research but also from the industrial perspective. And second,
how to avoid situations when projects will complicate grading.

Choosing a project that is balancing explanation of the course
material, has practical value, and is dated may look like a com-
plicated task, which it probably is. That is where getting feedback
from the industry may turn out to be helpful. The course instructor
has a better understanding of students’ capabilities and coupling
that with industry insights, our educator can modify the curriculum
to embed more practical exercises. Furthermore, we want projects
to help with grading. One approach to handle this is to introduce
a large semester–long project or several smaller projects and make
them modular. That may help students to incrementally work on a
project and gradually improve their grades. In such a setting, the
project can account for the major part of the grade. Doing so may
(and will) introduce additional challenges of planning the work,
which will be spread out over a long period of time. To make plan-
ning and working on a large project easier, it will be helpful to
set clear milestones with deadlines to avoid the situation when
students will frantically try to finish the work at the last minute.

CONCLUSIVE REMARK
The topic of computer science education is an area by itself and
conferences such as SIGCSE offer a great deal of research and inter-
esting ideas. Moreover, the education process is multidisciplinary
and that brings even more opportunities and places to look for ideas
on what can be done better or differently. The process of teaching
is a continuous effort of updating the material, gathering feedback,
and searching for ways of improving ourselves as educators. And
now, it may be a good time to review the existing teaching practices
and make necessary corrections.

ACKNOWLEDGMENTS
I would like to acknowledge all people who keep trying during
these times. Don’t give up and keep pushing.

REFERENCES
Computer-Based Tutoring: Impact on Learning Rate, Achievement and Attitudes.
245–252.
(2020). https://medium.com/great-research/online-courses-arent-better-or-worse-they-re-different-3a9a94375ae2
Key to Continued Success. Frontiers in Psychology 7, 1328 (31 August 2016)
Knowledge-Learning-Instruction Framework: Bridging the Science-Practice
and self-regulated learning: a model and seven principles of good feedback practice.
Education. In Proceedings of the 46th ACM Technical Symposium on Computer
Science Education. 90–95.