5-2021

A Golden Age for Computing Frontiers, A Dark Age for Computing Education?

Christof Teuscher
Portland State University, teuscher@pdx.edu

Citation Details

Christof Teuscher. 2021. A golden age for computing frontiers, a dark age for computing education? In Proceedings of the 18th ACM International Conference on Computing Frontiers (CF '21). Association for Computing Machinery, New York, NY, USA, 140–143. DOI: 10.1145/3457388.3458673

A Golden Age for Computing Frontiers, a Dark Age for Computing Education?

Christof Teuscher
teuscher@pdx.edu
Portland State University, Department of Electrical and Computer Engineering
Portland, OR, USA

ABSTRACT

There is no doubt that the body of knowledge spanned by the computing disciplines has gone through an unprecedented expansion, both in depth and breadth, over the last century. In this position paper, we argue that this expansion has led to a crisis in computing education: quite literally the vast majority of the topics of interest of this conference are not taught at the undergraduate level and most graduate courses will only scratch the surface of a few selected topics. But alas, industry is increasingly expecting students to be familiar with emerging topics, such as neuromorphic, probabilistic, and quantum computing, AI, and deep learning. We provide evidence for the rapid growth of emerging topics, highlight the decline of traditional areas, muse about the failure of higher education to adapt quickly, and delineate possible ways to avert the crisis by looking at how the field of physics dealt with significant expansions over the last centuries.

CCS CONCEPTS

• **Social and professional topics**; • **Applied computing**;

KEYWORDS

computing disciplines, computer architecture, education, crisis

ACM Reference Format:

Christof Teuscher. 2021. A Golden Age for Computing Frontiers, a Dark Age for Computing Education?. In *Computing Frontiers Conference (CF '21), May 11–13, 2021, Virtual Conference, Italy*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3457388.3458673

1 INTRODUCTION

If you studied computer architecture as a graduate or undergraduate, chances are high that you were exposed to Hennessy and Patterson's acclaimed "Computer Architecture: A Quantitative Approach" textbook. First published in 1990, and now in its 6^{th} edition (2017), it remains essential reading by instructors, students and practitioners of computer design. The latest edition has been updated with RISC-V examples and also includes a chapter on domain-specific architectures. While the 1^{st} edition had 784 pages, the 6^{th} edition adds up to a whopping 936 pages. One may wonder: do 152 additional pages—assuming that little foundational subject materials became obsolete—cover what happened in computer design for the 27 years since the 1^{st} edition was published?

In this position article, we will argue that what many in the field consider to be the golden age of computing frontiers, may, alas, turn into a dark age for computing education. There is no doubt that the computing disciplines have gone through unprecedented growth in depth and breadth over the last century, despite the fact that the field had been declared as "mature" previously. A decade ago, nobody talked about deep learning, mem-devices, in-memory computing, and neural engines, all of which have quite radically changed the way new chip architectures are designed today.

In 2004, Rosenbloom [4] mapped the relationships between computing and other fields. What was already a rich set of interactions has most definitely grown into an even richer set. As Denning stated: "A hallmark of the computing field has been its close relations with numerous other fields. The reason is not hard to understand: Information processes are part of most fields. All the taxonomies of the field from the 1960s to the present day contain some sort of entry for 'applications', which expressly acknowledges these many links" [2].

While the foundations—as covered comprehensively by Hennessy and Patterson's and many other texts—remained mostly unchanged, the last decade has seen a tremendous expansion of the computing frontiers in specific sub-areas. Not so in the computing education. How should higher education institutions address this seeming "disconnect" in what we teach in our curricula and what the job market expects from fresh graduates?

2 SOMETHING MISSING? A QUICK LOOK AT COMPUTER ENGINEERING EDUCATION IN THE US

Over the last decade, the author observed an increasing problem in the classroom, on his research team, and when students were looking for industry jobs: the vast majority of the topics of interest of this conference are not taught at the undergraduate level and most graduate courses will only scratch the surface of a few selected topics. Yet, to be hired in industry, it is increasingly expected that recent graduates have a substantial understanding of computing technologies at the frontiers, such as deep learning co-processors, approximate computing, in-memory processing, analog computing with memristors, ReRAM, and spintronics, to name a few. Among students, the saying goes that a computer engineer with great Python and machine learning skills will almost instantly land a job in these days.

CF '21, May 11–13, 2021, Virtual Conference, Italy

ACM ISBN 978-1-4503-8404-9/21/05…$15.00
https://doi.org/10.1145/3457388.3458673

We observe that higher education seems to do an insufficient job at teaching students crucial new skills. As a result, they are frequently taking online courses to learn on their own what they believe is needed to be successful in their careers. This seems to confirm an often-stated hallmark of US higher education: much is in fact learned outside of the classroom.

But let's take a quick—and admittedly incomplete—look at typical US undergraduate (B.S.) and graduate degrees (M.S.) in computer engineering in 2021. We will leave the PhD out of our discussion because it is not part of our concerns.

2.1 Undergraduate Education

Since the 1960s, ACM has provided undergraduate curricula recommendations [3] in response to the rapidly changing landscape of computer technology. For the computer engineering sub-discipline, ACM produced guidelines in 2004 and 2016. Both recommendations are based on a 420 hour core curriculum. Table 1 lists the knowledge areas that contain material that should be included in *all* computer engineering curricula. As one can see, the biggest change is a reduction of the number of areas from 18 to 12. Some areas were consolidated, others entirely eliminated. Note that the knowledge areas are not courses and the core components do not constitute a complete curriculum. The 2016 report specifically mentions that "[a]dditional technical areas, as well as supporting mathematics, science, and general studies, are necessary to produce a competent computer engineer." Nevertheless, the fact that no emerging knowledge areas in a rapidly changing field are included seems a reason for concern.

2.2 Graduate Education

Most M.S. programs in the US require students to take a set of core courses and select a set of electives from a list that can be—depending on the institution—quite substantial. Table 2, for example, shows the list of graduate electives for an M.S. degree in computer engineering with a specialization in hardware and computer architecture at Boston University. As one can see, emerging topics are, once again, absent from this list of courses.

3 THE EMERGENCE AND EVOLUTION OF SUB-FIELDS

Now, you may argue that it is not the goal of an EE/ECE undergraduate or graduate degree to equip students with skills at the frontiers of computing because (1) such skills can be learned on the job or (2) students can acquire the skills on their own. After all, we attempt to train them to become lifelong learners and there will most likely be lots of things they will have to learn throughout their careers anyway. Yet, the problem is not as simple. Being up to snuff with emerging computing trends often means becoming familiar with entirely new sub-fields and understanding their foundations. For example, a traditional computer engineer may never have been thoroughly exposed to analog electronics, but with the advent of memristors, computation with analog and non-linear devices suddenly became a thing. The same applies to designing accelerators for deep neural networks. Not only does this require a deep understanding of the neural networks, but a student may also have to learn about memristors because they are often used to store synaptic weights. Or, to design neuromorphic spintronics (magnetoelectroncis), a student needs to have a deep understanding of the underlying device physics, but also of neural networks and biology in order to efficiently use magnetic tunnel junctions as synapses and neurons. And it goes on and on.

In our current era of computer design, where do the foundations end, and where do the frontiers begin? Clearly, that boundary has shifted dramatically in the last decade. There seems to be an increasingly gap between the "old" and the "new" way of designing computers. For that purpose, let's look at some recent emerging trends.

Elsevier's Engineering Village thesaurus allows searching for standardized vocabulary terms that are used to index articles in their Inspec database. Figure 1 shows the number of records found for the "computer architecture" thesaurus term by year. As one can see, the field had its peak somewhere around 1985. Figure 2 shows the number of records over the years that were found for selected thesaurus terms that describe emerging technologies. For comparison, the "computer architecture" term is included again. As one can see, all emerging terms show exponential growth since their appearance. "Neural nets" produced more than an order of magnitude more records than "computer architecture" in its heyday. In fact, all emerging terms shown have surpassed "computer architecture" in 2020 (or earlier).

What seems clear is that the way we design computers has been disrupted in the last decade more than ever before. This is not to say that we have or will stop designing computer in the "old" way, quite the contrary. We—and most people in the field—do not see traditional computers disappearing, at least not any time soon. Yet, more than ever before, emerging trends seem to have gained critical mass to the extent that they cannot simply be ignored and or dismissed as yet another short-lived fad. And that is precisely the reason why we believe that a crisis in computing education not just looms around the corner, but it's happening as we write—so to speak.

To gain some additional insight, let's look at physics because the field went through a similar evolution: "There was a time when polymaths like Galileo knew all the physics that was there to be known. Over the centuries, however, the body of knowledge spanned by physics exploded, encompassing topics as diverse as gravitational waves, graphene, or network science" [1]. How did physics address this challenge? We'll answer that question in the next section.

4 ARE THERE WAYS TO AVERT THE CRISIS?

First, let's recap why the problem outlined so far is complex: (1) computing foundations are not obsolete, students still need to learn those, so we can't easily drop topics; (2) technological progress has accelerated as in no other field; (3) emerging computing sub-areas tend to involve complex subject matter and interact with various other disciplines, which makes specialization hard(er); (4) computing education has not kept up with the blistering pace of recent emerging computing models, paradigms, and hardware; and (5) the job market is increasingly expecting recent graduates to be knowledgeable in emerging areas, such as covered by this conference.

So, what might be possible ways to avert a full-fledged crisis at this point?

Table 1: Knowledge areas that contain material that should be included in all computer engineering curricula. Data source: [3].

2004		2016	
CE-ALG	Algorithms		
CE-CAO	Computer Architecture and Organization	CE-CAO	Computer Architecture and Organization
CE-CSE	Computer Systems Engineering		
CE-CSG	Circuits and Signals		
CE-DBS	Database Systems		
CE-DIG	Digital Logic	CE-DIG	Digital Design
CE-DSC	Discrete Structures		
CE-DSP	Digital Signal Processing		
CE-ELE	Electronics		
CE-ESY	Embedded Systems	CE-ESY	Embedded Systems
CE-HCI	Human-Computer Interaction		
CE-NWK	Computer Networks	CE-NWK	Computer Networks
CE-OPS	Operating Systems		
CE-PRF	Programming Fundamentals		
CE-PRS	Probability and Statistics		
CE-SPR	Social and Professional Issues		
CE-SWE	Software Engineering		
CE-VLS	VLSI Design and Fabrication		
		CE-CAE	Circuits and Electronics
		CE-CAL	Computing Algorithms
		CE-PPP	Preparation for Professional Practice
		CE-SEC	Information Security
		CE-SGP	Signal Processing
		CE-SPE	Systems and Project Engineering
		CE-SRM	Systems Resource Management
		CE-SWD	Software Design

Table 2: List of graduate electives for an M.S. degree in computer engineering with a specialization in hardware and computer architecture at Boston University.

EC 513 Computer Architecture
EC 527 High-Performance Programming with Multicore and GPUs
EC 535 Introduction to Embedded Systems
EC 551 Advanced Digital Design with Verilog and FPGA
EC 561 Error-Control Codes
EC 571 VLSI Principles and Applications
EC 580 Modern Active Circuit Design
EC 582 RF/Analog IC Design Fundamentals
EC 713 Parallel Computer Architecture
EC 749 Interconnection Networks for Multicomputers
EC 752 Theory of Computer Hardware Testing
EC 753 Fault-Tolerant Computing
EC 757 Advanced Microprocessor Design
EC 772 VLSI Graduate Design Project

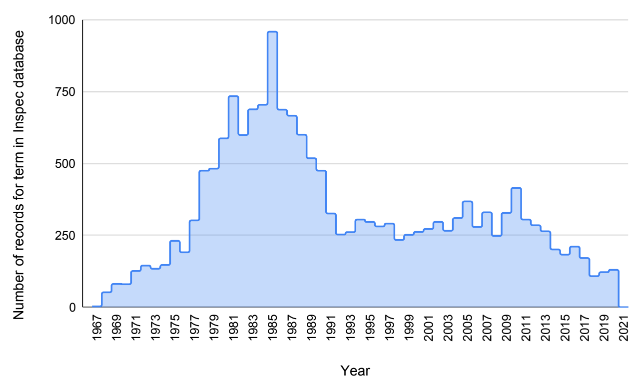

Figure 1: The number of records over the years that were found for the "computer architecture" thesaurus term. The field had its peak around 1985 and has since been in decline. Data source: Elsevier Engineering Village thesaurus.

In a 2017 article [5] we already suggested that "[i]f we want to successfully reboot computing and go beyond Moore's law, we must also reboot the traditional computer architecture, computer engineering, and computer science curricula to train a workforce that can tackle all these challenging problems. Some of these challenges will require not only young, fresh minds that can think outside the box but also educators who can think outside the box to teach the new technologies." What has changed since then? We argue that

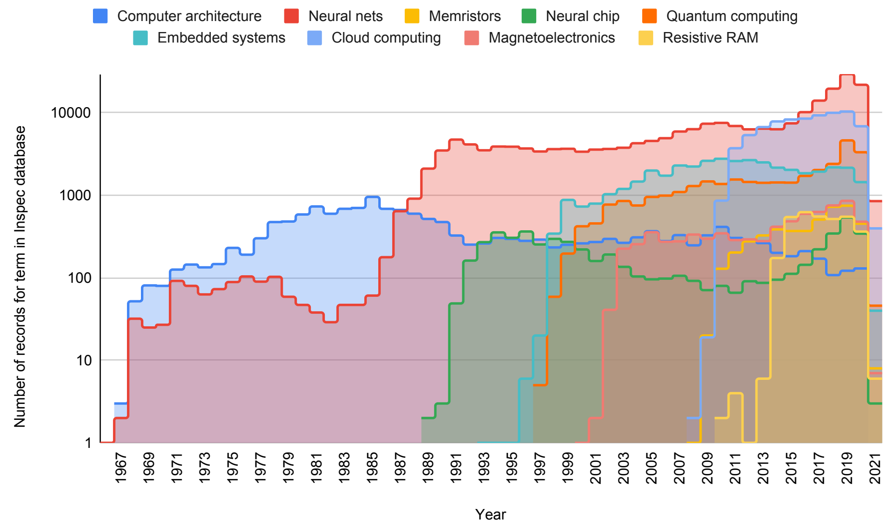

Figure 2: The number of records over the years that were found for a given thesaurus term. "Computer architecture" is included in this log-linear plot for comparison. Data source: Elsevier Engineering Village thesaurus.

the computing frontiers have even more expanded, but alas, the frontiers of computing education have only partially kept up with the fiery pace. The biggest and most obvious educational changes in computing were the addition of new graduate courses in areas such as machine learning, big data, and cybersecurity. For good reasons: ten years ago we removed a graduate neural networks course in our department because there was too little interest. Today, students are knocking down our doors asking for more machine learning courses. Several were added as a result.

But simply adding courses does not address the more fundamental problems. First, adding courses will at some point lead to a higher credit requirement for the degrees. We argue that the pressure to keep college costs down will make it unlikely that degree programs will significantly increase their credit requirements. That also puts the option of educating "computing polymaths," who would have a broad as well as a deep knowledge of all relevant fields, off the table.

Although the field of physics has expanded dramatically since Galileo, that happened over decades, and not as fast as in the computing field. Yet, we can still draw inspiration from how they dealt with that challenge: physicists were forced to specialize as the field expanded in both breadth and depth [1]. "As the aperture of physics widens, the focus of individual physicists narrows, leading progressively to the formation of specialised communities and subfields. [...] We observed that subfields rarely live in isolation but rather tend to overlap, with individual scientists working in multiple subfields and transitioning between fields during their career."

Specialization into sub-fields is—we argue—the most likely solution to the educational crisis the computing frontiers currently face. We can see that trend happening in the following examples: (1) the appearance of new, more specialized graduate programs, e.g., in data science and cybersecurity; (2) the master's degree is becoming the new bachelor's degree, thus allowing for more specialization and deeper knowledge; (3) the appearance of edX MicroMasters degrees (https://www.edx.org/micromasters), which allow students to specifically develop standalone skills for career advancement.

The major online learning platforms have quickly learned to capitalize on important new trends in a fast-changing world. For example, edX offers graduate programs on the Internet of Things (IoT), cybersecurity, and quantum technology. Higher education—known for being notoriously slow and often having a tendency to resist to change—does not only have to adapt to a new (post-)COVID world, it also has to learn to quickly adapt to an accelerating pace of new computing technologies. Otherwise students may turn their backs and rely on alternatives to be successful in their careers.

If we want the computing frontiers to become a real golden age, we need to educate our young talent differently, with the tools they will need for the new kinds of computing. "The golden age has not passed; it lies in the future." — Paul Signac.

5 ACKNOWLEDGMENTS

The author thanks Michael Bowman for the help with the Engineering Village thesaurus and Mark Faust for the feedback.

REFERENCES

[1] F. Battiston, F. Musciotto, D. Wang, A.-L. Barabási, M. Szell, and Roberta Sinatra. 2019. Taking census of physics. *Nat Rev Phys* 1 (2019), 89–97. https://doi.org/10.1038/s42254-018-0005-3

[2] Peter J. Denning. 2009. *The Computing Field: Structure.* American Cancer Society, 615–623. https://doi.org/10.1002/9780470050118.ecse962

[3] Association for Computing Machinery (ACM). 2021. Curricula Recommendations. https://www.acm.org/education/curricula-recommendations. Accessed: 2021-02-02.

[4] P. S. Rosenbloom. 2004. A new framework for computer science and engineering. *Computer* 37, 11 (2004), 23–28. https://doi.org/10.1109/MC.2004.186

[5] C. Teuscher. 2017. The Weird, the Small, and the Uncontrollable: Redefining the Frontiers of Computing. *Computer* 50, 8 (2017), 52–58. https://doi.org/10.1109/MC.2017.3001242