SIGCSE 2019

Summary by J.H.Davenport

27 February - 2 March 2019

Contents

Part I

27 February

Chapter 1

CyberSecurity Workshop

See `https://clark.center` [Apparently "CyberSecurity Labs And Resource Knowledgebase"].

"C5" materials were based on NSA support trying to fit CS AP Principles. We looked at `https://clark.center/details/blairt/Intro%20to%20Cybersecurity%20Syllabus`. So far there's only the syllabus, but there will be modules etc. It was noted that there was currently no recommendation for a book.

- Some aspects, e.g. crypto, might be hard, given no prerequisites stated.
- Very broad
- Pretty much fits my software.
- A bit more on forensics would be good.
- I use a CISCO curriculum (which is free, apparently), which might help.
- What resources are the lab: Massive cyber range, or a python script, or where?

A A wide variety should be there. Every Learning bject must have some hands-on activity.

A We don't put actual assessments into Clark (as the students can also access it). Occasionally there are assessments without answers.

Q I'll be teaching in Europe in two years: can I use it.

A It's open access.

1.1 Contributions

We are trying to esnure quality at the data entry end as well as through peer review.

We really want to see the Learning Outcomes. CSEC and ABET have Learning Outcomes and there should be easy mapping. The interface for creating units on Clark requires these, and connections between the assignments and the Learning Outcomes. Pick which Bloom's level you are at: the opening words of the Learning Outcomes are aligned to this, e g. "analyze".

`cis.towson.edu/cyber4all/` apparently has resources not yet on Clark.

Part II

28 February

Chapter 2

Opening

Opening video from 'Peter" (North Carolina) the Chair of the first SIGCSE in 1970 — one day, 143 attendees, 18 papers and no parallel sessions.

2.1 SIGCSE Chair: Amber Settle

There's more than this. ACM's third largest SIG. Last year was the SIG's 50th.

2.2 Keynote: Dr Marie Desjardins: CS Education, Inclusivity and Memebrship

Last-minute standin for her boss, Freeman A. Hrabowski III. University of Maryland Baltimore County. Regards diverity as a moral imperative. Note he was founding[1] (?Faculty,?President) of the first racially integrated state university in Maryland. This is #1 producer of African–American students who ??.

He spoke by video: 9th grade (but aged 12) kid in Birmingham Alabama who heard MLK, and persuaded his parents to lot him go on Alabama march. MLK said "you will influence children who are not yet born": 2/3 of today's America weren't born then. Also noted that it isn't just African–American who don't do well — many disadvantaged kids suffer.

Marie commented on pace of change "Facebook was born at the same time as my daughter" etc. 1984 was the peak of women CS majors, 37%. women.

We don't have enough professors now for the students we have, and we still can't produce enough students. Also, all other subjects will need CS. Data Analyst was new to Glassdoor's 2018 list, coming straight in at 38th.

Claims that 1984 was the launch of the Macintosh (clearly true), and that these were bought for boys, not girls. Look also at image of the Dilbert strip. Claims that the personality tests actually selected for anti-sociability etc.

[1] No - I think the point is that is was funded at time of Birmingham march.

Chapter 3

JHD looks at Co

Chapter 4

Autograders

Karen ?? (Manchester NH) chaired

4.1 Margaret Ellis: Approaches for coordinating eTextbooks, Online Programming Practice, Autograders etc.

2 lectures + lab per week.

WebCat is autograder.

Canvas is MS.

iClicker is the active response system

OpenDSA eTextbook with visualisation etc.

...

Senior TA thought there could be too many tools but wheren't there yet. "I came with no CS background, and I hadn't realised how many tools I was learning".

4.1.1 Issues

Content

Differences of vocabularity etc.

Student account management.

Transfer manual spreadsheet transfer is error-prone.

LTI integration saves a great deal of time. Self-enrolment in tools can lead to chaos. You have to choose and enforce a common key, e.g. e-mail. I assign a lot of small pieces of work, especially from OpenDSA, so I make this a separate course with its own gradebook..

4.1.2 Q&A

Q How important is LTI?

A I used OpenDSA etc. before LTI was there, so very important.

Q This tool gets superceded?

A That"s life.

Q Different OS etc?

A All except Eclispe are online for the CS2; other courses it can be an issue.

4.2 Evan Macius (RPI) Autograding Distributed Network assigments using Containers

Submitty is RPI's own autograder. It's a submission and Autograding system, whch is growing. Config is a JSON file. Contains command and validation method.

4.2.1 Q&A

Q How customised are all these graders?

A There's always some tweaking.

4.3 Peveler (RPI) Sandboxes or

Also Submitty. 4.5 years, 2500 users, 12–15 courses/term. This

1. Parallelism
2. Students shouldn't see other students' code, or solution
3. Students not affect the host system.
4. Shouldn't consume excessive resources,
5. Manage dependencies; Python/Pip; Ruby/Bundler etc.

4.3.1 Jailed Sandbox

Workflow.

1. Student submits
2. Daemon copies this plus any instructor files into a fresh user/group in a new location. Trouble is when different instructors when different sets.
3. Run with rlimit etc. and `seccomp` to restrict system calls, e.g. `fork`.
4. Copy key files back; then wipe and reinstall.

4.3.2 Docker

OS-level virtualisation (containerisation). Shares resources with host kernel and userland. Hence we can specify (per course, per instructor etc.) a base image plus a series of bash commands to install packages. But there problems with containers and resource limits: Java8 and other sees host properties. By default Docker runs everything as root, which is not a good idea. Even so the user has sudo access. Docker seems to add 2.5 seconds (1.5 seconds container creation; 1.0 destruction) to turnround time.

We measures resources and turnround time. Took busiest slice at mid-Fall 2017: CS1/Python and CS2/C++ at same time. And replayed this hour at various speeds. The graphs of Docker/Sandboxes were practically indistinguishable when run at 4× real. We had a limit of 10 graders, and when this was hit, docker had wait times up to 25 seconds. At 16× dockers was definitely hitting limits (especially RAM).

The database/OS container comunities have pool approaches that we should look it.

4.3.3 Q&A

Q There are other solutions, such as a VM per course. Also `chroot`.

A `chroot` has known issues.

Q Experiment?

A A single machine: 32GB RAM, 20 core processor.

Q Student view on 25 second delay?

A–student It's normally 5 seconds: 25 is OK.

Chapter 5

Databases

5.1 Basit/Chen (U. Virginia) A learning platform for SQL Injection

Professor (teaching databases) and final year student. I teach a security course: lots of nuggets but no integrated resource. "Affects any data driven SQL database application". Note that Microsoft's response to [sar98] was "don't worry about it". The Fortnite game exploit was SQL injection (on an inactive game). Showed [XKC07]. Apparently there's a company at UK Companies house called `; DROP COMPANIES` (apparently missing open quote).

LAMP stack, MariaDB. 12 challenges at different levels. Each database has a MariaDB user with only SELECT permissions. But it is important to teach students about access control first. Have an exercise "select a user, select a query, see if it works" to work out permissions each user has.

Actual tools has 12 levels:

1 Display all rows

10 Password matching with `admin' AND password LIKE 'a%'` (guessing each character at a time).

12 Password retrieval via timing attack (only executed if password matches)

`http://biy.ly/SQLInjectionTool`. There's also a lot of tutorial material. Or (?same) `databases.cs.vigina.edu`.

5.1.1 Q&A

Q Is this for database students or security students?

A Primarily database.

JHD to self — surely this shows the misguidedness of the whole area.

Q How long does this take.

A I give them 30 minutes in a lab, then the rest is own time. Should be a few hours.

5.2 Taylor (Oberlin) '); DROP TABLE textbooks;–:

An Argument for SQL Injection Coverage in Database Textbooks.

Co-author did all the database book reading!

Claims Ashley Madison (which led to suicides) was SQL[1], and a Wordpress one. Equifax (the information hasn't surfaced, so probably a nation state), and [FBI16]. There's a YouTube video of a three-year old doing it (Author is "Hunt" apparently).

Dynamic SQL queries (string concatenation) are the fundamental flaw. Prevention via parameterized queries. Easy to trash Stack Overflow, but are our textbooks better?

RQ1 Does the book teach SQL Injection. Only 2 out of 7. 5 of these have security chapters! The same 5 have web application chapters.

RQ2 Does it teach parameterized queries? 5 out of 7 do demonstrate them, and the other two didn't use user input at all (!?).

RQ3 Does it have vulnerable examples. 2 out of 7 (have 2 each, presented as correct thing to do). A really blatant PHP example.

Note that instructors do use textbooks to guide coverage in the class. Also students retain textbooks.

- Choose good textbooks
- If you can't, get the students to cross the bad parts out: they'll enjoy doing this, and remember it.

5.2.1

Q What about Security books.

A Good question: next project!

[1] JHD: but see `https://www.wired.com/2015/08/ashley-madison-hack-everything-you-need-to-know-your-questions-explained/`.

5.3 Toni Taipalus (Jyväskalä): What to expect and what to Focus on in SQL Query Teaching

Co-author doing a PhD on CyberSecurity in IS Setting.

What errors do students make, and are they unfixable (persistent). Syntax errors are generally non-persistent. But logic errors (e.g. $\neq$ rather than $=$) are harder.

744 sudents in 3 cohorts with no prior SQL experience. Each cohort took 15 (different) questions. The correct answers were given to the students. Any internet etc. was available. So 1230,000 queries, 8700 were the "final" ones. [TSV18] categorises errors: 4 classes, 18 categories and 105 error types. "function errors" (persistent) show up massively but only in a few queries. For example, the exercise has to require a join to force join errors.

Hence we need to identify the query concepts behind each exercise to see what we're testing, then identify the corresponding persistent errors, and then fix the exercise table so that the persistent errors give different results.

5.3.1 Q&A

Q How did you come up with this classification?

A Lots of manual work!

Chapter 6

Testing

6.1 Kazerouni (Virginia Tech): Assessing Incremental Testing Practices and Their Impact on Project Outcomes

157 students with 4 assignments, each 1500 lines of code. Aim: give students better feedback on their programming practices. We wanted to track the evolution of the project, even "procrastination" — prior work. Analyze work session by coding/testing (actually test writing). Then break down by method as well. Evaluate $\underset{\text{sessions}}{\text{median}} \frac{T_s}{T_s + S_s}$, and do the same for methods. Also looked at whether test code was written before or after the last edit to the code of the method.

Students use Eclipse, and we have a grabbing plug-in. Outcomes were correctness, and code coverage of student's own test suites. Use WebCat, hence incremental feedback (on correctness and coverage). A worksession is bracketed by $\geq$ 1hour inactivity.

6.1.1 Q&A

Q Were the tests correct?

A No — that's another question.

6.2 Aniche (TU Delft): Pragmatic Software Testing Education

Teaching to year 1 CS students. Why? Society wants code that works, industry wants developers with experience. But there is little attention paid: not many teaching tools, and no curriculum. IEEE: "Worlds Apart: Industrial and Academic Focus Areas in Software Testing". 4th quarter of first year; 140 hours.

Testing a 3kLOC Pacman-like game. 3 hours lecture and 4 labs/week. We have 300/450/750 students in the last three years (TU Delft CS now teaches in English). Our paper has 9 key elements:

1. Apply theory (e.g. state-based testing)
2. Pragmatic discussions (what trade-offs). Should I mock this class?
3. Building a testing mindset. Guest lectures.
4. Software testing automation (very common in industry) Junit; Mockito; Selenium/Cucumber.
5. Hands-on lab work. This is open source `https://github.com/serg-...`.
6. Test code quality matters: we train TAs (students from previous year) to be picky.
7. Design for testability — lectures for this. JPacman is a good example.
8. Theoretical and Practical Books. ISTQ8 *Foundations of Software Testing* and *Pragmatic Unit Testing in Java 8 with JUnit*.
9. Guest practitioners, e.g. Air Traffic Control.

RQ1 What common mistakes? Test coverage. Maintainability of test code (lots of duplication rather than refactoring). Boundary testing

RQ2 which topics are easy/hard. JUnit etc. easy. Structural etsting is easy, but MC/Dc difficult.

RQ3 Best teaching methods. They don't like the books. Scalability is an issue.

6.2.1 Q&A

Q How do you train TAs.

A They are ex-course. Extensive roadmaps for feedback.

Q Did you seed faults in JPacman.

A It does have errors, but that's not the point: we want to see techniques applied systematically.

Q Then how do you grade?

A For state machines we know how many, otherwise branch coverage etc.

6.3 Garcia: Software Testing in Introductory Programming Courses

RQ1 What topics? 9091 papers; select down to 229, then snowball to 293. Classified down to nine topics. Questions like programming practice. Test quality.

RQ2 Pro/con. Should improve programming Objective assessment. But workload Students do poorly and are reluctant.

Chapter 7

CS Accreditation: What you should know

A workshop on accreditation led by three senior ABET people.

"ABET" is a four-letter word. But seriously "ABET is a specialist accreditor in applied science ...". A federation of 25+ societies. It is volunteer-driven, with 2200+ volunteers. Four-tier system: Board of Directors/ Board of Delegates (from member societies)/ Commissioners (who lead campus visits)/ Program Evaluators. 100% of accreditation decisions are made by volunteers. Various commissions, but, say, Criterion 1 is the same across all. We are CAC = Computing Accreditation Commission. The Commission reviews education programs under their urview and make final decisions. Continuous review and enhancement of the commission's criteria, policies and procedurs Changes require ABET Board approval,

Why accreditation? Ensures that a program maintains quality over time. Verifies quality. Makes sure program is committed to self-improvement and accountability. Ensure graduates have the educational requirements necessary to enter their chosen profession. Note that EAC does Computer Engineering and Software Engineering. CAC does Computer Science (includes such as Game Design, Networking), also specilisms in CyberSecurity, information Systems etc. Look at `http://www.abet.org`.

7.1 Program Education Objectives

These are the statement of where students are expected to be in a few years time. Student Outcomes state what graduates should know and be able to do by graduation. Must prepare students to achieve the PEOs. Ther are general criteria (CAC and EAC), and also Program Criteria for specialist disciplines.

1. Students
2. PEO

3. Outcomes – vary
4. CQI (Continuous Quality Improvement?)
5. Curriculum – vary. Must lead to a career, further study *and continuous professional development*. Now needs 30 semester credit hours[1] of Cybersecurity — can be one course or embedded. Fundamental and advanced topics must demonstrate breadth and depth.
6. Faculty – vary. Some facuty members must have PhDs in computer science. Unchanged since last time, but being studied.
7. Facilities
8. Support

Our criteria are based on [ACM13]. This introduces Information Assurance and Security. But these only come into effect 2019–20. So collect materials during 2018–19 year; self-study reports due 1 July 2019. Visits in fall 2019. draft reports in winter 2019–20, due process reports. Final reports in late 2020. EAC similar.

7.2 Changes to General Criteria

Now five required SOs:

1. Analyze a complex problem;
2. design and implement a computer-based solution;
3. ...
4.
5.

and the program can add more. Currently in transition, so the plan has to be there, but not necessarily fully in place. There is no check on the requirement that each graduate demonstrates all outcomes, but there must be a pathway such that every student can.

7.3 Program Criteria Changes: Computer Science

There's a specialist session on CyberSecurity requirements. Note that we looked at [ACM13], with a view to making them feasible across sizes of departments. Note that this is about areas, not specific courses. General Criteria 1–5 and

[1] Federal definition

6: produce a solution. Project can be individual or group, and needs to be "at least several weeks". If the project is tied into student outcomes, then the project has to be assessed.

1. 40 semester credit hours of CS[2], including a major project that requires integration and application. Also some "exposure to"[3] items, such as operating systems.
2. At least 15 hours of mathematics at least as rigorous as introductory calculus
3. At least 6 hours of natural science. Scientific method and laboratory work.

* Note that it used to be 30 of Math+Science, so there's a reduction here.

7.4 Q&A

Q PEOs vs SOs?

A We no longer expect people to measure PEOs (graduates are no longer your students)

Q We are CAC (Computer Science) and EAC (Software Engineering): different numbers of SOs.

A Life! SE is under EAC because it includes the word "Engineering".

Q Cybersecurity looks vague: what can you point us at?

A CS2017 [Ass17] has knowledge areas. Also best place to start is CS 2013 [ACM13].

Q Depth?

A Has to be shown in the self-study. If CS1 was the only pre-req for everything else, we'd be worried.

7.5 BoF on CyberSecurity Requirements

Led by Larry Jones (ABET), Allen Parrish (USNA—Now Mississippi State), Rajendra Raj (Rochester IT), but used last year's slides. Note (previous session) that ABET has added cybersecurity for all. But the service academies have added specialist degrees, and some community colleges are added specialist degrees. ABET will accredit CyberSecurity programs as well. Did a "round the table" view — NSA, academies, smaller state universities. One explicitly said that she wanted to teach "general programmers to code well".

[2] As opposed to 30 which is the general requirement.
[3] Slide of deer caught in headlights: it's not this!

Industry Demand: see `cyberseek.org`. Missouri commented that their degrees were state-approved, which required indusry support. State/Local Government, but also Boeing. Community Colleges — we only hire people with industry experience, and there's a great deal of demand. "We're very close to NSA, and they take our two-year graduates, whereas most Federa requires four-year graduates". Community colleges: "we're hands on, competency based".

Debate about how much mathematics is required. HLS require a four-year degree because of problem-solving ability.

Part III

1 March

Chapter 8

Mark Guzdial

"My advisor made me read CP Snow: Two Cultures. It's been with me ever since".

Computer Scienec: The study of comuters and ll the phenomena associated with them [NPS67]. George Forsythe invented "Computer Science" in J. Engineering Education. Perlis: everyone should study Computer Science, just as they should study calculius: Calculus is the study of rates, and Computer Science is the study of process. Recall also Papert's robots.

Shows graph of # CS teachers in Scotland (declining) and data from Roehampton. Depressing US data from, say, Indiana: 0.5% of students take CS in school. 43% of Georgia High Schools have never offered CS, and $< 1\%$ take it. Most AP exams have a majority of females taking them: CS is the most male-dominated AP exam.

Quotes RAEng definition of "Engineering Thinking", and claims this is mostly computational.

Most SCratch programs used `forever` loops, very few Booleans etc., so you can do a lot with little. "There is amazing learning power in a small subset of CS". Examples of sound waves and their Fourier transforms.

There are a (very) few CS educators in Education Departments: we ened many more of them. There are also read challenges around Computing for Special Needs students — under researched.

Asking Provosts etc. "more money for our majors", tends not to work: "more money to bring in more diverse students" might, but "more money to teach computational thinking to improve learning and outcomes for *all* students" is a much better pitch.

CS0? No — everyone should learn programming. We live in a literate society: hence teh pressure on schools to tecah reading and writing. But we live in a computing society: hence we should teach everyone about it. Let's start with every university student. A corollary of this would be (in time, JHD adds) every school teacher.

Chapter 9

Testing 2

9.1 Weikle (James Madison U.): Automating Systems Course Unit and Integration Testing: Experience Report

We tell them the differences between Java and C, but don't teach C. They have no prioir experience of memory management etc.

Skill Goal Develop command line/Linux-style C programs

Concept Goal Understand what's happening "under the hood".

Idea — "spec-inspired grading". There are 'D' tests, 'C' tests etc., and you have to pass *all* the tests at a level and all lower ones to get that grade. We have unit tests, integration tests, memory leak tests and unsafe function (done with `grep`) tests.

```
START_TEST (name)
{
  check_assert_int_equall(add_abs(2,3),5)
}
END_TEST
```

9.1.1 Q&A

Q Students submitting tests for credit?

A Great idea.

Q Statistics — do students miss levels.

A Students know what they need to do, and well over 50% get A's.

Q What about a student who fixes an A test aand causes a C test to break.

A They get a D. But ther's a 10% penalty late policy, so this would be the route.

9.2 Fraser(Passau): Gamifying a Software Testing Course with Code Defenders

I've been teaching (struggling to teach) software testing for a number of years. Game "Code Defenders" based on mutation tests.

9.2.1 Q&A

Q What happens if a hacker plays the game "optimally", e.g. if `x\~101`

A There are limitations on what they can introduce into their code.

Q Improved retention?

A No hard data.

Q GRiu formaton.

A Rabdomm allocation.

9.3 Measuring Unit Test Accuracy

What is "good quality" testing? Note that we test students (quizzes, exams etc.)

- Code coverage is misleading and becomes a dangerous goal.
- Actually finding bugs. I used "instructor tests", but also ran against other students' tests. Poor tests were highly correlated with poor implementations. See our paper in CSEE&T2019.

Accuracy = $\frac{TP+TN}{TP+FP+FN+TN}$.

H1 Accuracy correlated with implementation. Very much (75%), beats coverage which beats bug identification

H2

9.3.1 Q&A

Q

A

Chapter 10

Security 1

10.1 Lee (Stanford): Evaluation of Peer Instruction for Cybersecurity Education

Peer Instruction. Various methods — I use this.

1. Mini-lecture (7-8 minutes), then an MCQ with individual votes (recorded) then small group discussion and re-poll. We used this in "Introduction to CyberSecurity" [seniors/graduate students: 33 students] (this paper), and in Introduction to Penetration Testing and Introduction to Forensics.

* Note that these MCQs have a different goal than straight assessment.

* Never grade on correctness.

* "Testing before you teach" is a valid technique here.

    - Deliberate ambiguity
    - Trolling for misconceptions

Example 1 *An attacker deletes files on a system, denying access. Which of C/I/A/more than one?*

Example 2 *What fraction of disc blocks are affected by a format? Answer choice was 100%, 65%, 20%, $< 5\%$. Correct answer is $< 5\%$ for modern FAT discs.*

Great evaluations — see [CuttsetalSIGCSE16]. `http://peerinstruction4cs.org`

10.1.1 Q&A

Q What %age of students do the pre-reading?

A self-reported 60%.

Q How important is it that you show the results.

A I never show the intermediate results (unless it's a tie). Recall that the purpose is discussion, not assessment.

10.2 Womack (Moorhouse College) From Midshipmen to Cyber Pros: Training Minority Naval Reserve Officer Training Corp Students for Cybersecurity

Speaker was a senior student. C-SCoRE : Cyber Spectrum Collaborative Research Environment. It's a multi-university ROTC/NROTC program. This program is three institutions: Female HB, male HB (Moorhouse) and a coed ug/grad HB. 14F/11M, 40% STEM (3 of whcih were CS: 2F1M). Many cohorts elsewhere are all CS. 15 weeks (3 fall; 12 spring). once/week for 2 hours (1 initially, but this didn't work). Individual work, then team projects (3 or 4).

The program was fundamentals of CS (data structures, APIs). Python with IDLE; Maplotlib etc.

1. Basic user profile (given a Twitter handle)
2. customised user profile
3. language analysis
4. sentiment analysis

DoD-funded, so classified in detail. 86% were more interested in CS after this. Students said that the independent assessment and the group projects were pain points. We actually need to teach some these skills as well.

10.2.1 Q&A

Q Longitudinal results?

A Not formally, but it's small and I know it's worked.

Q Classified so hard to repeat!

A APIs are public — I'm looking at Instagram for the next cohort.

Q Credit?

A No, but the students wer compensated.

10.3 Topological Scoring of Concept Maps for Cybersecurity Education

JHD had to leave early.

Chapter 11

Competency-Based Education Workshop

Led by Amardeep Kahlon (Director of "Fast Track Program", and on Board of CBE Network) and Ann Kennedy from Austin Community College. Roundtable of attendees: wide range: a freshman, various school, community college and university teachers, various companies including AWS. One said "I want to develop project-based learning".

Had three months to transfer a semester to CBE. Program ained at veterans, unemployed and under-employed. 1 FTE industry engagement officer. Remarkably powerful backgrounds, (e.g. MD Anaesthesiologist who had to retrain) and great women/African-American figures.

- I evaluate Prior Learning via CBE.

Note that traditional "mid-terms+finals" doesn't help the student who fluffs understanding week 1.

Note that traditional assessment is fixed time and variable outcome, while CBE is converse. This is a fundamental difference, and really contradicts credit-hour based systems (not just for students, but workload assignments etc.).

Part IV

2 March

Chapter 12

Introduction

12.1 Anniversary

This is SIGCSE's 50th Anniversary. There's an award for the best papers in the first 49 SIGCSEs. A committee selected 20 which were put to a public vote. See `https://sigcse.org/sigcse/files/documents/pdfs/Test%20of%20Time%20Award%20nominations.pdf`.

10 Objects First [CDP03]

9 Contributing to success in a introductory ... [WS01]

8 What should we teach — David Gries [Gri74]

7 Using software testing to move students [Edw04]

6 Constructivism in computer science education [BA98]

5 The introductory ... 10 principles [Sch78]

4 A multiinstitutional study of peer instruction (many authors) [PBC+16]

3 Undergradiate women in computer science: experience ... [FMM97]

2 Improvingthe CS1 experience wth pair programming [NWW+03]

1 Identifying student misconceptions of programming [KPEH10]

We're also creating a test-of-time award.

12.2 Introduction

Received over $5M in funding, as a "Clinical Associate Professor"[1] at Towson University.

[1] Essentially what Bath would call a Senior Teaching Fellow.

12.3 Blair Taylor

"Russia/China are poised to bring downthe electric grid for days if not week". "Cybersecurity is a lot like a woman: it takes a lot to figure it out, and then it changes again"

aged 16 Went to college (Johns Hopkins) — lived with parents. No CS degree available at the time.

aged 48 Finished doctorate.

7 years in industry: starting working for American Totalizer. Then EMC Controls. Someone asked "what is Blair's dog's name" over the intercom, as it was a password. They wanted to promote me, but I was pregnant, so the firm created a part-time position for me.

30 years teaching: notionally part time, but .... Changed jobs, started doctorate, but had to drop out. Develop a programme called "Security Injection" to teach secure coding from day 1. Then teachng at Towson. "Girls like to work in pairs". Also noted that girls felt underprepared. Now working on Clark (chapter 1)— designing an "Intro to Cybersecurity" course.

Not only is there a shortage of Cybersecurity people, there's a shortage of faculty to teach them.

4 reasons it's a game changer.

- *All* students (not just geeky white men) like Cyber.[2]. Turned up at a conference to given "CyberSecurity – where the boys are", but the organisers changed the title to "CyberSecurity – where are the girls". But which is more attractive to girls?
- CyberSecurity can draw more diverse students
- Cyber creates opportunities for CS faculty to engage all students.
- CyberSecurity makes a difference. This can be a great motivator, especially for girls.

12.3.1 Q&A

Q Industry pays more — is the joy of teaching worth the salary sacrifice.

A We need more CyberSecurity PhD programmes,and also need more industry faculty to help us. But standard part-time pay rates don't work: indutsry need to donate them.

Q Started a High School program, but what to teach?

A We have some resources in Clark, but not perfect.

[2]Daughter went to Cornell to study CS. Included in introductory pack was a (male) deodorant. Stereotypes??

Q How do you get the boys to be allies.

A Need to look at all aspects.

Q What did you mean by "not political"?

A The aim: defence of the country, e.g. the grid, is bi-partisan.

Q I accept that everyone needs to know about CyberSecurity. What about Political Science etc.

A Clark has 20 videos on Cyber warfare which can be shown to a History etc. class.

Q You mentioned a diversity problem. But isn't it really an inclusivity problem. So I want to call you out about your "jokes" etc.

A Sometimes you have to play along to get along.

Q I keep being asked "You're from IBM: please give me a CyberSecurity teacher". But I'd rather we taught teachers.

A Good point.

12.3.2 Postscript: e-mail from Organizer

Dear SIGCSE community, we hear your concerns about this morning's keynote. We acknowledge that the keynote was not representative of the values of our community. The message conveyed while well intentioned, it was expressed in a manner that was not inclusive.

We understand that we need to have more conversations about inclusivity. We also believe that these conversations need to be had in a public forum, in a constructive and respectful manner.

At the same time, we feel that the comments caused members of our community to feel isolated and excluded from the conversation. We must work harder to bring all of us together into an inclusive community where people belong and share common values.

Proudly, we feel that our community expressed their values very respectfully during the Q&A period.

Moving forward, we need to:

- have a more open and candid conversation about these issues,
- continue to recognize the impact that micro aggressions have in members of our society,
- strive to understand why some perspectives persist and how to move away from them.

On Thursday's opening session, we were told that it is the community's responsibility to remind speakers to use the microphone. We also heard that many participants took that statement to heart and acted accordingly.

We hope that this morning's keynote has a similar impact and eventually brings us together to be an even more inclusive community.

Given the travel schedule of the conference organizers and members of the SIGCSE board, we respecfully request a few days to provide a more complete response to this situation.

Chapter 13

Mathematics

13.1 Sigurdsson: Students' perspectives on Mathematics in Computer Science

There's work on the rôle of math, and on students' perceptions, but not much intersection. CS requires an intro mathematical proofs cours,e linear algebar, and calculus, which has three options of varying levels of rigour. 2017 was a N=13 qualiytative study. Then an N=219 quantitaive survey.

2017 learning-oriented[1] students were worried that their Yr1 choice cut off options. Hence 2018 survey, which has big take-up of free-text results. Quoted one (negative: they just gave us the code) example. Another attacking "Theory of proof" — all we need is induction. We've tried a different institution, with no mandatory math. Actually got the same factor analysis.

13.1.1 Q&A

Q Is "Intro to proof" for both mathematicians and CS?

A Yes, taught by mathematicians. The question "who teaches" does seem to be very important.

13.2 Fowler:Impact of Steps, Instruction, and Motivation on Learning Symbolic Reasoning Using an Online Tool

Typically debugging and dry-running uses concrete inputs, numbers. Want to look at a symbolic trace tool. This is very course-oriented ("you ar enow on exercise 7").

[1] The contrast was "industry-oriented".

N=114, first interaction with tool. “Tool is useful, but interface is gross” etc. “Why”. Si we explained the purpose next year. Then N=92. But also “I felt I used logic more than random guessing”.

13.2.1 Q&A

Q How do you know this isn't a cohort effect.

A Similar statistics, but hard to be sure.

Chapter 14

Security

14.1 The Politics Behind Internet Routing Security and Insecurity

Internet Map1982: basically Arpanet + leaves. But the gateways were manually configured, unlike the Arpanet internal dynamic routers. 1978: Strasiser/Perlamn "There must be a method for gateways to authenticate the existence of other gateways ...". This still isn't really solved. misconfiguration within an AS is now sandboxed, bu tthe same problem arises with networks of ASs. The first routing hijack was by Harvard in 1973, blamed on a memory error (distance 0 to everywhere).

"The Net interprets censorship as damage and routes around it" — Gilmore 1993.

"The hidden story of China's BGP Hijacking". [DS18]

14.1.1 Q&A

Chair [1] "The Internet is held together by peanut butter and goblins" *a propos* the recent African BGP hijack.

Q Do you think this "philosophy" is really about the application layer?

A Often. But a lot of the Internet infrastructure models geopolitics.

Q Why is the NO community so slow to adapt?

A I wouldn't blame these. The OSI people were still trying to draft protocols by 1985, while Apranet was on demo deadlines.

Q About this routing, can they read the messages?

[1] Alyssa Moore Senior Policy & Advocacy Advisor, Canadian Internet Registration Authority.

A As Diffie said, there's a lot that can be read into encrypted traffic. Also it can be stored for time and easier decoding. Note also that "end-to-end" is a retrospective justification.

14.2 Dupont: Computer and Network Security: Understanding Communities of Ethical Research

See Spamalytics paper [**?**]. Also [BF15]. This got an "ethics warning" from the PC: now common.

So we're dong a "grounded theory" piece of research. A majority of reviewers are concerned with the community's ethical practices. Papers lacking IRB or ethical consent. Note that IRB is a rare thing outside the USA. Lawyers are becoming more involved. "University Ethics courses" seem to be a significant factor in where people learn.

14.2.1 Q&A

Chair Have you studied the industry side of research?

A Our research is about academic researchers.

Q "You said they were well-equipped".

A I don't think so. They *have* to do it. Menlo Report doesn't seem to help.

Q Internet measurement conference etc.?

A Yes, I look at this. "Active measurement" is problematic as well.

14.3 Rethinking Values in the Design of Security Technologies

So some people want to encode value in the technology, and wrote to IETF as such.

14.3.1 Q&A

Q–Chair In these days of AVs and their biases, do these concerns translate.

A IETF is pretty remote from "I'll sort out snooping on the IoT". After Snowden, IETF came up with "privacy considerations",. Pretty much a show.

Chapter 15

Big Data Analytics with SPARK

I'll be using Scala, but Python is there. Java is more verbose.

1. JHD installed Python
2. JHD installed Pip
3. JHD used pip to install pyspark and numpy
4. `python SimpleAppRDD.py` was tried, and complains that "Java not found", which is absurd as JHD has various Java programs, such as Maple and MATLAB (which use Java for the frontend).

Give up practical work.

15.1 Introduction

For me, "Big Data" is more than one can fit on a single machine. Was born by the Google MapReduce project [DG08a, DG08b]. But if one has many machines, one needs a fault tolerant system. The main implementation of MapReduce is Hadoop, which does a lot of disc saving to ensure fault-tolerance. Spark replaced the distributed computation part of Hadoop. Spark keeps a graph knowing what's being done, so that a single failing machine can be repeated. Because it's uniform, you can do Spark on one machine, which helps debugging. Java, Python and R interfaces. I use Scala: it's the native implementation language (here first), with no translation. Your code runs on oen machine. There are then multiple executors, on the same or different machines.

Two APIs — low level is based on RDD and functional programming. High level is SparkSQL.

RDD = Resilient Distributed Dataset. There are transforms, such as `map` and `filter`. These are lazy. Actions are methods that produce a value on the

master. Note that debugging actions is hard. Actions are `reduce` and `fold`. Note that these are in parallel, so have to be associative and commutative. These operations have the same type, so can't do, say wordcount. Need `aggregate` which takes both a transforming function and a collection function. `collect` brings results back to the master machine.

All the work is done in a `SparkContext`, which we set. Also `setMaster`. The code gets converted to JVM bytecode.

```
txtFileLines.filter(line => line.contains("val")).count()
```

Note that `count` is an action, counting on each executor then adding up the results on the master.

15.2 Example on Census Data

The file (from Kaggle) is basically a csv. SparkRDD doesn't have a csv reader (SQL does) so a hack to remove the header line, then parse each line into a data structure (with some parsing of strings to ints etc.). Do do analysis by race, split into tuples, then use `aggregateByKey`[1] to do the same calculation for each value of the key (race).

To find the median, he sorts, then does `zipWithIndex` (equivalent to `zip` with $(1, 2, 3 \ldots)$) and then picks the right element.

Giving an RDD a name doesn't prevent it being recalculated. By default Spark does the computation for each action. Need to explicitly with `.cache()`. By and large, SparkRDD does what you tell it: clever or dumb.

15.3 SparkSQL

Many readers, e.g. CSV, JSON (but one object/line). Apparently not XML. The fundamental abstraction is the Dataset. `filter` is also called `where` to keep SQL-hackers happy. Build a `SparkSession` rather than a context. SparkSQL has various encoders, whereas RDD relies on JAva serialisation (ouch!).


Writing your own aggregator is the last resort: if you can make SQL do it, great.

Note that a schema will let the builtin csv reader do the parsing. Stll need caching. The (rough) equivalent of `aggregateByKey` is `groupBy`.

It is possible to pass SQL-like strings in, but these are run-time parsed. Problems for error-checking, but the efficiency is about the same.

Note `readToTypedDataset`.

[1] In a special package of `PairRDDFunctions`. There's also `DoubleRDDFunctions` for things like sum on numeric values.

15.4 MLLib

The Machine Learning library for Spark. This is the newer library, on top of SparkSQL. There are several options for missing values, e.g. drop, replace by median etc.

Bibliography

[ACM13] ACM/IEEE-CS Joint Task Force on Computing Curricula. Computer Science Curricula 2013. Technical report, ACM Press and IEEE Computer Society Press, December 2013. URL: http://dx.doi.org/10.1145/2534860, doi:10.1145/2534860.

[Ass17] Association for Computing Machinery/IEEE/AIS SIGSEC/IFIP WG 11.8. Cybersecurity Curricula 2017. https://www.acm.org/binaries/content/assets/education/curricula-recommendations/csec2017.pdf, 2017.

[BA98] Mordechai Ben-Ari. Constructivism in Computer Science Education. In *Proceedings of the Twenty-ninth SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '98, pages 257–261, New York, NY, USA, 1998. ACM. URL: http://doi.acm.org/10.1145/273133.274308, doi:10.1145/273133.274308.

[BF15] S. Burnett and N. Feamster. Encore: Lightweight measurement of web censorship with cross-origin requests. *ACM SIGCOMM Computer Communication Review*, 45:653–667, 2015.

[CDP03] Stephen Cooper, Wanda Dann, and Randy Pausch. Teaching Objects-first in Introductory Computer Science. In *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '03, pages 191–195, New York, NY, USA, 2003. ACM. URL: http://doi.acm.org/10.1145/611892.611966, doi:10.1145/611892.611966.

[DG08a] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. https://static.googleusercontent.com/media/research.google.com/en//archive/mapreduce-osdi04.pdf, 2008.

[DG08b] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters (Slides). http://labs.google.com/papers/mapreduce-osdi04-slides/index.html, 2008.

[DS18] C.C. Demchak and Y. Shavitt. China's Maxim — Leave No Access Point Unexploited: The Hidden Story of China Telecom's BGP Hijacking. *Military Cyber Affairs pp. 7:1-7:11*, 3, 2018.

[Edw04] Stephen H. Edwards. Using Software Testing to Move Students from Trial-and-error to Reflection-in-action. In *Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '04, pages 26–30, New York, NY, USA, 2004. ACM. URL: http://doi.acm.org/10.1145/971300.971312, doi:10.1145/971300.971312.

[FBI16] FBI. FBI Flash T-LD1004-TT (18 August 2016): Targeting Activity Against State Board of Election Systems. https://s.yimg.com/dh/ap/politics/images/boe_flash_aug_2016_final.pdf, 2016.

[FMM97] Allan Fisher, Jane Margolis, and Faye Miller. Undergraduate Women in Computer Science: Experience, Motivation and Culture. In *Proceedings of the Twenty-eighth SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '97, pages 106–110, New York, NY, USA, 1997. ACM. URL: http://doi.acm.org/10.1145/268084.268127, doi:10.1145/268084.268127.

[Gri74] David Gries. What Should We Teach in an Introductory Programming Course? *SIGCSE Bull.*, 6(1):81–89, January 1974. URL: http://doi.acm.org/10.1145/953057.810447, doi:10.1145/953057.810447.

[KPEH10] Lisa C. Kaczmarczyk, Elizabeth R. Petrick, J. Philip East, and Geoffrey L. Herman. Identifying Student Misconceptions of Programming. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, SIGCSE '10, pages 107–111, New York, NY, USA, 2010. ACM. URL: http://doi.acm.org/10.1145/1734263.1734299, doi:10.1145/1734263.1734299.

[NPS67] A. Newell, A.J. Perlis, and H.A. Simon. Computer Science. *Science*, 157:1373–1374, 1967.

[NWW+03] Nachiappan Nagappan, Laurie Williams, Laurie Williams, Miriam Ferzli, Eric Wiebe, Kai Yang, Carol Miller, and Suzanne Balik. Improving the CS1 Experience with Pair Programming. In *Proceedings of the 34th SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '03, pages 359–362, New York, NY, USA, 2003. ACM. URL: http://doi.acm.org/10.1145/611892.612006, doi:10.1145/611892.612006.

[PBC+16] Leo Porter, Dennis Bouvier, Quintin Cutts, Scott Grissom, Cynthia Lee, Robert McCartney, Daniel Zingaro, and Beth Simon. A Multi-institutional Study of Peer Instruction in Introductory Computing.

In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, SIGCSE '16, pages 358–363, New York, NY, USA, 2016. ACM. URL: http://doi.acm.org/10.1145/2839509.2844642, doi:10.1145/2839509.2844642.

[sar98] Jeff Forristal (signing as rain.forest.puppy). NT Web Technology Vulnerabilities. *Phrack Magazine*, 8(54):article 8, 1998. URL: https://web.archive.org/web/20140319065810/http://www.phrack.com/issues.html?issue=54&id=8#article.

[Sch78] G. Michael Schneider. The Introductory Programming Course in Computer Science: Ten Principles. In *Papers of the SIGCSE/CSA Technical Symposium on Computer Science Education*, SIGCSE '78, pages 107–114, New York, NY, USA, 1978. ACM. URL: http://doi.acm.org/10.1145/990555.990598, doi:10.1145/990555.990598.

[TSV18] T. Taipalus, M. Siponen, and T. Vartiainen. Errors and Complications in SQL Query Formulation. *ACM Transactions on Computing Education (TOCE) pp. 3:1-3:15*, 18, 2018.

[WS01] Brenda Cantwell Wilson and Sharon Shrock. Contributing to Success in an Introductory Computer Science Course: A Study of Twelve Factors. In *Proceedings of the Thirty-second SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '01, pages 184–188, New York, NY, USA, 2001. ACM. URL: http://doi.acm.org/10.1145/364447.364581, doi:10.1145/364447.364581.

[XKC07] XKCD. Cartoon 327. https://xkcd.com/327/, 2007.