SIGCSE 2017

Notes by JHD

8–11 March 2017

Abstract

The following key points occurred to JHD.

1. Jeanette Wing's "next big idea" (JHD's expression): section 2.2.2.
2. If all you use is an autograder, then the message is that all that counts is the autograder, and plagiarism is fine. See "Post" on page 22.
3. The vast majority of papers were "we tried this, and the results were . . . ".

Contents

1 Aligning to the ACM Cybersecurity-infused Computer Science Transfer Curriculum **3**

2 March 9, 2017 **5**
2.1 Welcome 5
2.2 Embracing Uncertainty: Jeannette Wing 5
2.2.1 Data Science Workflow 7
2.2.2 Conclusions 7
2.3 CS Education Research Knowledge Forum 7
2.4 The Educator Identity and its Impact 8

3 Performance Analytics **9**
3.1 Using Learning Analytics to Investigate Patterns of Performance and Engagement in Large Classes 9
3.2 Automatically Classifying Students in Need of Support by Detecting Changes in Programming Behaviour 10
3.3 Evaluating Neural Networks as a Method for Identifying Students in Need of Assistance 11

4 Afternoon Thursday 9th March **13**
4.1 Generating Hints and Feedback for Hilbert-style Axiomatic Proofs 13
4.2 In-Lab Programming Tests in a Data Structures Course in C for Non-Specialists 14
4.2.1 Conclusion 14
4.3 Interactions of Individual and Pair Programmers with an Intelligent Tutoring System for Computer Science 15
4.4 BoF: Practical Systems Programming in Computer Science Education 15
4.5 BoF: Improving Effectiveness of CS Teacher Professional Development 17

5 Friday 10 March **20**
5.1 Breakfast with BlueJay/Greenfoot 20
5.1.1 Greenfoot 20

5.1.2 Stride . . . 20
5.1.3 BlueJay . . . 21
5.1.4 Conclusion . . . 21
5.2 General Business . . . 21
5.3 Gail Chapman . . . 22
5.3.1 Who we teach . . . 22
5.4 Innovative Pedagogical Approaches to a Capstone Laboratory Course in Cyber Operations . . . 22
5.5 Micro-Classes: A Structure for Improving Student Experience in Large Classes . . . 23
5.6 Impact of Class Size on Student Evaluations for Traditional and Peer Instruction Classrooms . . . 24
5.7 My Digital Hand: A Tool for Scaling Up One-to-One Peer Teaching in Support of Computer Science Learning . . . 24
5.8 A Pedagogical Analysis of Online Coding Tutorials . . . 25
5.9 Lessons Learned in the Design and Delivery of an Introductory Programming MOOC . . . 26
5.10 Employing Retention of Flow to Improve Online Tutorials . . . 27
5.11 How to Collect, Analyze and Act on Learning Data in Computer Science Courses . . . 27
5.11.1 Introduction . . . 27
5.11.2 Collect . . . 28
5.11.3 Analyse . . . 28
5.11.4 Discussion and Action Items . . . 28

6 Saturday 11 March **29**
6.1 Codio (Supporter's session) . . . 29
6.2 On the (Mis) Understanding of the "this" Reference . . . 30
6.3 Assessing and Teaching Scope, Mutation, and Aliasing in Upper-Level Undergraduates . . . 31
6.4 Multiple Levels of Abstraction in Algorithmic Problem Solving . . . 32

7 Closing **33**
7.1 Introduction . . . 33
7.2 Talk: Fulfilling Papert's Dream (Mitchel Resnick) . . . 33
7.3 Q&A . . . 34

Chapter 1

Aligning to the ACM Cybersecurity-infused Computer Science Transfer Curriculum

Organised by the Committee for Computing Education in Community Colleges. See `ccecc.acm.org`. Current status is pre-release final draft. Note that one target is CS transfer degrees. There is a BoK (Body of Knowledge) with a three-tiered assessment rubric — Bloom's Revised Taxonomy is used.

Note half of all students in US HE are at community colleges, and also half of those at four-year colleges have attended community colleges. This effort is based on CS2013 [ACM13]. Note that this is about to have a CyberSecurity Appendix (and a Data Science one is coming). CS 2013 had a new Knowledge Area[1] on Information Assurance and Security, and elements "sprinkled throughout" the curriculum.

We divided CS 2013 KAs into three clusters. KA and KUs, along with Bloom levels. CS-Transfer includes all KAs from CS2013 except "Intelligent Systems". Also renamed "Information Assurance and Security" as "CyberSecurity" (just has more traction). Also "Platform-Based Development" has no LOs (in CS2013 or here).

Bloom: Remembering, Understanding, Applying, Analyzing, Evaluating, Creating: list of verbs associated with each level. Note that CS students create a lot! Each LO has three levels, "Emerging Standard", "Developed Standard" and "Highly Developed Standard" (which might be at different Bloom levels). 41% of our LOs are at "Apply". Out of 216 LOs for Associate-Degree Transfer Programs, 26 are CyberSecurity. Note that there are also 22 "Social Issues and Professional Practice".

[1] Reminder: KA $\neq$ course.

Q How does this compare with four-year guidelines?

A 17 of 18 KAs from [ACM13] are present, but only parts of them. Some KAs had all KUs incorporated (Discrete Structures, Software Development Fundamentals), the rest some. All written in terms of Bloom verbs.

Q What does ACM do to tell (state) departments of education that there is a standard?

A Good question.

Q How does IT and CIS relate?

A IT (as a two-year degree subject) is intended as a route directly to a job. CIS isn't an ACM term — degrees called CIS might be IS or IT. In my college, CIS is moving into the Business School.

There's a mapping process (tool on the website) for fitting your program into our KAs/LOs. Header information, then (repeatedly) "add course" (title and description, typically cut/paste of course), then (repeatedly) tick the box for which LOs are met by this course. Then review it (which tells you how many LOs), and then "submit", which mails the Committee to review (filtering for stupidity). There's no check as to every LO is mapped: your needs are different from mine. This is primarily a sharing tool.

Portland's non-credit side, which has done a lot of preparation for certifications, is now running a code camp in collaboration with LaunchCode. One participant had built a CS adaptation for LaunchCode: LaunchCode has a very high rate — they have employment partners. At my college, we use this for pre-study guidance. The LaunchCode guy quotes 20–30% female, but couldn't remember other diversity figures. There are intensive women-only tracks in some locations.

Chapter 2

March 9, 2017

2.1 Welcome

Sponsorship Was worth $210.45 per attendee.

1435 Attendees: highest ever.

SIGCSE Third largest SIG.

2.2 Embracing Uncertainty: Jeannette Wing

"It's been a tremendous 10 years for me", or even more since [**?**]. But I felt I had a very cool reception when I spoke at SIGCSE 2007 in Kentucky. We (i.e. you) have made tremendous efforts since them. I just wanted to change "Intro to CS", and am impressed by the change to K–12. This was pushed by NSF. I was dismayed: 10,000 school districts. The time I really thought it was real was Obama's 2016 State of the Union. He requested 100M dollars, but, having been at NSF I know what that means.

We've solved (well, actually, no, but that's because majors are growing) the CS Faculty shortage: the K-12 CS shortage will be solved too. Let's worry about what happens next.

Sources of Uncertainty: I will be promoting probability and statistics. Traditional sources of uncertainty include faults: systems and networks. Hence distributed consensus protocols etc. Sensors on 'phones, say, have uncertainty. Graph of uncertainly of someone (intern) walking round campus. We are going to have self-driving cars in the future: get used to it! Human beings (image of girl on bicycle darting into street) are great sources of uncertainty. There's also the malicious attacker.

Example of machine learning, face/not face, and there's a somewhat fuzzy boundary. We're use to this. Recommendation systems are another source of uncertainty: 10M users and 10K movies.

Example 1 *Shows her training a movie system with "likes". Starts with "Pretty woman". each like/dislike moves the diagram of like/dislike. After four, the diagram begins to separate. Choosing a movie near the "like" line doesn't change the graph much. The 'data' that she likes it doesn't provide more 'information'.*

We need to represent uncertainly with probabilities: probabilistic state machines etc. 15 years ago this was an interesting intellectual exercise: now it's important.

Acknowledge Chris Bishop for next bit. Used to fair, and even unfair coins.

Example 2 *Sampling (with replacement) from one of two jars A and B. $p(X) = p(X, A) + P(X, B)$. Hence joint and conditional probabilities.*

Theorem 1 (Bayes)

$$\underbrace{p(y|x)}_{posterior} = \frac{\overbrace{p(x|y)}^{likelihood}\ \overbrace{p(y)}^{prior}}{p(x)}$$

Example 3 (Simpson's Paradox) *See example in Wikipedia, especially Berkeley admissions. My example is kidney stones. Treatment A (major) and B (minor). Small stones: 93% = 81/87 major, and 87% (234/270) minor. Large stones 73% (192/263) versus 69% (55/80). But for both: 78% (273/350) or 83% (289/350) says B wins.*

See also Prosecutor's Fallacy.

Example 4

Example 5 *Project has five parts in parallel, each has a 50% of chance being on time. Overall project 3%. Uncertainly ± 1 week, uniform. What's the 50% date: 19.2 days, and 90% 20+ days.*

Example 6 (Election) *Pollsters all gave Trump a significantly non-zero chance.*

Example 7 (Image) *A man falling into a crevice in a ford, labeled "average depth 3 feet".*

Therefore we need to know distributions, not just means.

Example 8 (Valentine's Day, ex Bishop) *Cost \$1.47, price \$1.99, and a past sales history (6 samples). Based on average, order 1M cards, profit \$500K, based on distributions, order 850K, profit \$360K.*

Observation 1 (JHD) *Since the lost profit when you under-order is \$0.52, but when you over-order is \$1.47, there's no reason to believe that a linear process (means) is correct. JMW's conclusion was "follow the distribution". But we don't know the distribution, merely six samples. Is it:*

1. *each of these six values, with probability 1/6*
2. *uniform across* $[\min, \max]$
3. $N(\mu, \sigma^2)$ *based on these samples?*

These will surely give different answers.

Example 9 (Walking) *Compute speed by difference of GPS reception. But a fitness app claimed the intern was doing 59mph. Real programmers use filtering etc., and its 1000s of lines of code. Will new programming languages help: can we get the level of abstraction right? We have one called* `Uncertain<T>`*, with an* `uncertain` *declaration, leading to distributions being used. There was a DARPA programme here about 4 years ago.*

Claims this needs three concepts: Random, Observe and Infer (the posterior distribution).

2.2.1 Data Science Workflow

This also has implications for what we teach. The ML step is just one of many, and probably not the most important — teach the whole workflow. The question is what are the errors (loss). Typical flow is Image → Deep Neural Network → decision, but should output a probability.

2.2.2 Conclusions

We need to elevate the status of "data" and "distributions" to the same status as algorithms. We should require probability/statistics in all CS courses. We should also offer an ML/DS to all students, not just CS majors.

2.3 CS Education Research Knowledge Forum

Led by CSNYC. NYC has 1.1M schoolchildren, and CSNYC's aim is that every NYC child *must* (2015 Mayor's mandate) learn CS. Just offering elective AP doesn't count — e.g. "every 9th grade must take a course". The CSNYC Knowledge Forum was launched last year. We have a $5M external evaluation fund as part of the budget. NYU+others just won that contract. The 2016 forum had 60 individuals. Our goals here are to replicate the 2016 activities and add to our data.

What followed was basically a brain-storming exercise in groups. JHD's group ranked teacher professional development (both in-service and new) as the top priority. Access/equity (e.g. blind students) was alas rated bottom.

2.4 The Educator Identity and its Impact

This was the SIGCSE Life Service Award, presented to Mats Daniels (Uppsala). Especially supporting the international network.

. It helps me that my wife is also a colleague. Coming to these conferences has also changed our son. I wear a different Hard Rock Cafe T-shirt to every lecture: I feel comfortable like that. Part of my identity is professional competence, in me and in my students. Collaboration is a must: how do we assess this? Competence with different cultures is important — how to assess and encourage. Time zone differences exacerbate this. How do the students, and we, deal with different contributions.

Also Faculty Identity: faculty as role models. A challenge is that students optimise their time, against course descriptions amongst other targets.

`matsdaniels.wordpress.com`.

Students who "just want to get the marks", ignoring the context, are not being professional.

Chapter 3

Performance Analytics

3.1 Using Learning Analytics to Investigate Patterns of Performance and Engagement in Large Classes

Hassan Khosravi[1]; Kendra Cooper.

Efforts to identify subpopulations (high, medium, at risk of failing) tend to focus on summative assessment. But in fact the reasons for "at risk" vary. How do you help in a large class. Aha — I should use my Machine Learning background to help with this.

Definition 1 *Learning analytics is the measurement, collection, analysis and reporting of data about learners and their contexts, for purposes of understanding and optimizing learning and the environment in which it occurs.*

Originally, it was felt to be easy, like movie recommendation systems, but it's much harder.

Introductory C: 1000 students and 70 TAs. 78 available raw scores are organised into a vector of nine features in three dimensions (Summative, Formative[2] and Behavioural[3]). For example, S_2 is first midterm grade. Used $k$means as clustering technique. Choosing k is hand: use the elbow method (the point at which "within cluster sum of squares" stops seeing big gains". Ran 100 times with different initialisation for each k.

C1 H,H,M — strongly achieving

C2 H,M,L — prior exposure. Need enrichment.

C3 M,H,H — underachieving, probably lacking soft skills;

[1] University of British Columbia (but now at Queensland)
[2] attending lectures etc.
[3] Out of class behaviour.

C4 L,M,M.

C5 VL,VL,L — at risk of dropping out.

E1 H,H,H — strongly achieving

E2 L/L/VH — lacking in soft skills, trying (but failing) to make up with online materials etc.

E3 ?

Also show students a dashboard, which shows their performance in all features against the cohort. Students don't often get this.

Q How long does it take the professor to go through 1000 graphs?

A On-demand (only office hours attendees).

Q Do you encourage students to come to office hours?

A Not currently

Q Validation of this method?

A Hard.

Q When did you do this?

A After second midterm.

Q Just looking at the video, or deep learning?

A Good question. There's also privacy issues.

3.2 Automatically Classifying Students in Need of Support by Detecting Changes in Programming Behaviour

Anthony Estey, University of Victoria; Hieke Keuning, Open University of the Netherlands; Yvonne Coady, University of Victoria.

33% fail CS1, often because they fall behind and can't catch up [references]. Code reading and code writing questions. Hint features. Didn't affect formal grades.

Question 1 *Can analysis patterns in interaction data help us to understand how to detect and measure learning in CS1.*

Look at hints, repeated hints, compilation attempts, and solutions. We find that we can identify patterns within two weeks of the course start. Low hints and high good compiles implies strength, but the opposite isn't true: the weak students are there, but also a lot of False Positives. Moved from a static analysis to looking at trajectory: can they now answer a question they previously needed hints for? We can perhaps distinguish transient problems from deeper ones? Still unsolved questions: is success the same as learning? How should we best guide students? Why do students exhibit ineffective study behaviour?

Q Many students didn't use it — self-selecting?

A Many of these were people who dropped the course. The rest doesn't seem significantly different.

Q Questions?

A Ordered by difficulty within a given topic.

Q What about Java, where there isn't "compile"?

A I was trying to distinguish phases of student activity. Not sure how this would translate.

3.3 Evaluating Neural Networks as a Method for Identifying Students in Need of Assistance

Karo Castro-Wunsch, University of Toronto Mississauga; Alireza Ahadi, University of Technology Sydney; Andrew Petersen, University of Toronto Mississauga.

The problem is the high drop/fail rate: 30%. So the task is identifying student sin need of assistance. As early as possible. But want to avoid catastrophic negative feedback, e.g. failed midterm. Good prior work (Vihavainen's data) using Weka.

Q1 Can we reproduce?

Q2 Can we use Neural Nets?

Q3 Are some models less sensitive to context

Input (X) was programming exercises in Java (Helsinki, N=263 and 109; 116 assignments) or Python (Toronto, N=897 and 519; 30 assignments). The prediction (Y) was the grades (Midterm etc.). The features used were Steps, number of submissions, and Correctness, percentage of tests passed. Other features we had didn't seem to add to predictability.

Of the seven models, three are definitely better than the other four, but the top three were better at predicting different things. Original to Full term

could get 86% (Random Forest) or 85% (Bayes): both from Weka. But early prediction wasn't nearly as good.

Our N baseline model structure is rectified liner in the internal layers, 200 nodes/layer, fully connected layers, cross-entropy cost function. NN is actually the best method (better than any Weka) at predicting fail.

Chapter 4

Afternoon Thursday 9th March

4.1 Generating Hints and Feedback for Hilbert-style Axiomatic Proofs

Josje Lodder, Bastiaan Heeren, Open University Netherlands; Johan Jeuring, Open University Netherlands and Utrecht University.

Context: logic for CS students. Hilbert-style = "a sequence of proof lines: axiom, assumption or one of Modus Ponens or Deduction Theorem". $q \to r \vdash (p \to q) \to (p \to r)$: various possible proofs. When a wrong attempt is given, the tool explains why the line is invalid. Student can also ask for a hint, or even a complete proof. For these, we need a solution strategy. Alternative solutions implies a Directed Acyclic Multigraph. Based on an algorithm by Bolotov (?JHD [BBGS05]). We think that mistakes consist of

- oversights
- conceptual mistakes
- creative rule-adaptation

So we define "buggy rules" [BB78] in LogAx (11 for MP and 6 for Deduction). Compare with MathMath's proofs. We compared 24 theorems: 22 were equal up to order, and 2 we were shorter. For exam-1 we were very similar to student answers, not for exam-2 and homework. Small pilot (5 students) showed very few syntactic errors, LogAx recognised most of the rule mistakes, but templates introduced a new kind of bug. The initial results were not conclusive. `ideas.cs.uu.nl/logax`.

Q Dead ends?

A There are ignorable lines in Hilbert, but not dead ends as such.

Q Computers are good at "what" but not "why"

A Yes, that's why we need richer hints — future work.

4.2 In-Lab Programming Tests in a Data Structures Course in C for Non-Specialists

Edwin Knorr, University of British Columbia; Christopher Thompson, British Columbia Institute of Technology.

A terminal (2nd)[1] course in C for non-CS EE majors.

2012 Initial (small) year.

2013 No in-lab tests

2014 In-lab programming tests Individual rater than pair. 5 test at 2% each. Average 68.2% SD=23.6. The results were actually worse than previous year. But in fact 2013 TAs were more lenient. Linked list question mean was identical in both years. However, student confidence (survey) improves significantly. Survey on "what did you/partner" contribute leads to asymmetric result. Did a survey of *same* students two years later. 81% of those who replied said that the tests improved their ability. Also asked a question about paper/computer programming exams. Slightly more thought the computer would give better marks, but "I'd waste time on syntax etc." was about evenly split. Also "Engineering is a zero-sum game: time just gets switched between courses" — 74% agreed. 44% said that this (terminal) cours ehelped them get a job (part/full/coop) in computing.

We believe in reading code: understand and add functionality. Used pair programming with role switch every 10 minutes. Different self-selected partner for each assignment. One motivation was that this halved the workload.

[Roedigeretal2010a][2] increased, spaced and cumulative testing improves learning.

4.2.1 Conclusion

Moving to in-lab tests doesn't improve exam scores, but confidence and pair programming satisfaction improved.

Maybe, is the examination too different than the programming tests? Note that this is for non-CS majors, so "just passing" might be enough for the students.

[1] All, 1000, engineering student at UBC take a C course in the first year. Apparently this also has in-course programming tests.

[2] JHD couldn't find this, but [RK06] exists, and is powerful.

4.3 Interactions of Individual and Pair Programmers with an Intelligent Tutoring System for Computer Science

Rachel Harsley, University of Illinois at Chicago; Davide Fossati, Emory University; Barbara Di Eugenio, Nick Green, University of Illinois at Chicago.

There's a common belief that programming is learned individually. ChiQat[3] is our tutoring system. Example discussed is linked lists. ChiQat has a graphical representation, and lines of code typed by the student can affect the graphical representation of the linked list. This was designed for individual learning. Pair programming with driver/navigator metaphor. [Williamsetal2000] shows benefits of pair programming for learning. Hence Collab-ChiQat. Seven problems of increasing difficulty. Individual condition in spring 2015, pair in fall 2015. Same teacher etc. 12 minute pre and post tests, 40 minutes with ChiQat. Also survey.

Started from a corpus of 54 tutoring sessions. Gains were 0.14 human, 0.10 individual and 0.11 pair. n average students completed four, leaving three incomplete. Pairs requested significantly less ($P < 0.01$, $t = 4.79$) system-provided examples than individuals. individuals also spent longer on examples (significant four four out of seven) Time: two was significantly less time, less in six of seven.

What about coding efficiency: #operations, #restarts Individuals took 23 operations on average, pairs 13. Individuals switched problems significantly more often. In both conditions students found the tutor helpful, and interesting.

Q Pre/post tests?

A Individual

Q Synchronous or async?

A The two students were sitting next to each other with one screen.

Q You were saying they spent more time on the tutorial: counter-intuitive?

A Yes, we weren't expecting it so hadn't surveyed for it.

4.4 BoF: Practical Systems Programming in Computer Science Education

Convenors: Peter Froehlich, Johns Hopkins University; Borja Sotomayor, University of Chicago. We (convenors) were amazed to find people doing things other than CS1. Are we re-inventing too many wheels?

Q

[3] Available at `digitaltutors.net`.

UPenn student

?? Looking

Nathan I'm the systems guy in a department of 2.

Columbia Intro to sysp in C,OS

? At a community college, but would like to introduce my CS200 to what is coming

?? Montgomery College. I teach CS200, interested in data structure assignments.

Furman U. Chair wants par

Seattlke U I teach OS.

CMU Parallel architecture etc.

Iowa college Teach OS, networks.

CMU I teach intro to systems, online course on cloud computing. Interested in autograders.

PF Been at JHU since 2005, gradually inherited all the systems courses. Talk about two of them.

Compiler Intro They compile a very simple language. First aim is to understand the pieces. I show them a sample C. Assignments are 'scanner', 'symbol tree' etc. Ends up being a capstone assessment. Also SE, as bugs in the scanner have to be dealt with later.

OS I hand them MIT's XV6, basically Unix 6 in modern C. Not always pretty! We sit in class reading the code. Assignments are 'different scheduler', 'add threading'. The first assignments are in user space, then as we read the kernel, we write assignments depending on these bits. In XV6, I sometimes need to say "just believe this for the time being". They will have read scheduler, file system etc.

BS When teaching OS, I used PintOS kernel. You can boot a physical machine with this (in practice a VM). The scaffolding abstracts away the very machine-dependence. In networks, they implement[4] based on RFCs, providing scaffolding. We give the students an Ubuntu VM. This solves all sorts of compatibility issues.

Both XV6 and PintOS are small systems you can read, and the systems just boot in a VM.

Q Is this feasible for students who have only seen Java, and not Linux.

[4] e.g. TCP. But I now give them the circular buffer code, as this used to be a big distraction.

BS Recommends "NAND to Tetris" curriculum: hasn't used it, but heard nothing but good.

PF For our data structures in Java, we don't use a textbook: just pointers to a set.

Q What's the balance between scaffolding and problems. Should a DB course give them an SQL parser?

BS In my databases class, I given them a "text to data structure" SQL parser. It also depends on what they have already learned. We've iterated over these for several years, with student feedback.

Both These assignments take so long to develop, that we can't throw them away. But it does change every year. This means that you can't give a reference solutions. I run MOSS (Stanford, but you have to submit the code set there) on all my past solutions.

BS For architecture, I teach ARM, as that's pretty new at the moment.

Q I use bleeding edge kernel each year, and that means I have to rewrite everything. So your mention of XV6 is impressive.

Q' My students have said they'd rather write from scratch, rather than make modifications to a "real" system.

BS Even SQL-lite is too big for me to grok, never mind students. Hence writing a DB from scratch is far better than modifying.

4.5 BoF: Improving Effectiveness of CS Teacher Professional Development

Karen Parker, Sloan Davis, Chris Stephenson, Jason Ravitz, Google: these were the listed convenors, but more

CS for HS is a global funding program since 2009. In the US this is aimed at working with local communities. So how do we evaluate?

1. Monitoring. Light touch.
2. Scientific curiosity. Research reviews, stakeholder groups etc.
3. Programme evaluation. We ask PD practitioners to complete a report. Also pre-/post- survey of the taught.

2015/16 evaluation 2015 2900 educators. 94% said CS4HS very valuable; 93% of teachers would recommend. We learned four lessons.

1. Tailor content to experience level. The PD practitioners found mixed classes very hard. Some folks had multiple tracks. Peer-led teaching is also important if you have a mixed group.

NY Not actually CS4HS ourselves, many of our partners are. We've had to impose a 2-trip cap. We have multiple offerings.

* There's also middle/high school differentiation, and amount of equipment etc.

? We are teaching engineers to be teachers. Instead of giving strategies, we're giving techniques for developing strategies. Teaching CS isn't fundamentally different from other subjects, but skills need adaptation. Are you giving them the fish you have caught?

Cornell Tech I enjoy diverse groups. I used to be special ed., so create assignments at different levels.

* Mixed-background is the life of teachers, so we need to model it.

2. Emphasise CoP development as well as straight professional development. Also a plan for sustainability and continued growth. Note that you will have multiple cohorts.

Q What is your definition of CoP? There's a special definition in educational theory.

SS In the UK, we have local Hub meetings, separate from PD opportunity.

? Plug for local CH4HS chapters.

Google How do we structure CoP as lighthouses rather than walls.

? The only way we know to build communities is F2F PD.

Cathy I evaluates CS for Alabama. Then did a week of F2F PD with ongoing local meetings. It was exemplary.

But A CoP tends to require teachers driving somewhere and taking time off other activities. This doesn't scale beyond early adopters. Other CoP take time during Superintendents conference days etc.

NC We do a hangout every 3-4 weeks, which works because we've met F2F. The group grows organically. The guy on the far end of a 2.5 hour ferry can't come in physically!

3. Meet teachers where they are.

? Format/content/schedule are not distinct topics. Temptation is to "go electronic", but it's not the same. Online is a great paradigm, but the content needs to be adjusted. Fixed schedules makes it easier for families.

Code.org Teachers entering the CS community need to get plugged in. Let's not forget social media: Facebook group etc.

4. Tangible pathways for teachers: where do you go next, and what experiences can you offer your students next. How can we provide pathways when standards etc evolve so quickly.

Colorado Teachers need to co-develop.

UT Austin We can't control what's happening, hence we need to give teachers autonomy. I've been doing PD in Natural Science for 20 years. You have to make teachers feel contributing parts of the community.

? We have teachers create the material (maybe with assistance on the content).

Google As CS4HS at UMass Lowell, has an "unconference" with teachers deciding what to learn/discuss. This works for the teachers who know what they need.

Chapter 5

Friday 10 March

5.1 Breakfast with BlueJay/Greenfoot

Note that one member of the team had been denied entry: wrong nationality/religion.

5.1.1 Greenfoot

Standard example of the crab: writes `move(4)` in the relevant method, and pressing 'run' causes the crab to move. Then adds conditionals. In a couple of minutes he has a keyboard-controlled graphical object. Also a foxes/rabbits simulation, and a piano keyboard.

5.1.2 Stride

Has now (just over a year ago) been added to Greenfoot. There are a lot of advantages to block-based languages, but text is a lot more efficient (and readable) for professional programmers. So what should we do? The transition becomes a problem, and the gap is quite large. Alice etc. have a "multiple views" solution, but we want a single view. So in Stride, hitting the 'if' key inserts an 'if' statement with appropriate holes. These frames become first-class citizens in the editor, so we are editing at the statement level. For example, you can't have missing brackets, since we don't have brackets! Again, indentation should be none of your business: when we move the block, the indentation is automatically correct.

Proposition 1 *The single character is the wrong unit of editing for writing programs.*

Q How did you add an `else`?

A Hitting the 'e' key with the 'if' selected. Note there's also a cheat sheet of syntax and shortcuts.

'Method call' is also a piece of syntax. The prompt tells me the name of the formal parameter names. This is a mouse-over like object, unlike, say, Eclipse, which inserts the name, or Netbeans which inserts a name of the right type. Note that both are inserting errors in your program!

5.1.3 BlueJay

We developed Stride for Greenfoot, and were slightly surprised that it was wanted for BlueJay. Now in BlueJay 4. Note that Java is still there! In BlueJay 4, we offer help with finding good 'import' statements, and we fold that section by default. Because Stride is context-sensitive, we can offer appropriate suggestions.

Did a 'cut' from his Java program and 'paste' into Stride, and it's automatically converted. Stride annotates whether or not a piece of code is a method override, and adding parameters will change the annotation appropriately. BlueJay 4 supports JavaFX styling through CSS.

We are well aware that people may want to end up in Java (with either BlueJay or Greenfoot) so there's a "Java preview" option which shows the program in Java syntax. So one can go back and forth (we don't expect rapid switching) as long as not using advanced Java features like `lambda`.

Also added continuous Dropped CVS support (anyone remember that?) but added Git. Also highlighting multiple compile errors rather than just the first.

5.1.4 Conclusion

There is more to come in future versions.

Q Extension API

A Currently unchanged. We will rewrite it from Swing to FX.

NB Our "Blue Room"/"Green Room" are not public, so we check that people who sign up are teachers.

Q When should I wean my students from BlueJay to, say Eclipse.

A Agreed. We use BlueJay only in first year for CS students. But I fundamentally believe that frame-based, as opposed to character-based, editing will come for programming languages in general. It may take 20 years. There is no reason why non-CS students should ever need to move from BlueJay/Stride.

5.2 General Business

Numbers Now 1497

2017 Best education research paper: 'computing with CORGIS'.

2018 Theme 2018 is CS FOR ALL. Baltimore. feb 24-28 2018.

5.3 Gail Chapman

Award winner 2017. Asked how many in the room were K-12: JHD estimates 10–15%, with many more working with K-12 on at least a monthly basis: when she started at SIGCSE it was "fingers of one hand". Noted 'CS for all', but what does that mean in practice?

> I believe that *public* education is a right for all, regardless of race, gender identity etc.

5.3.1 Who we teach

Think about who inspired us: audience to discuss in groups.

Long description I couldn't follow about her growing-up experiences, positive and negative.

She believes we should teach children about

- creative nature of computing
- ability of computing to solve problems
- relevance of computing to today.

All student should have access to AP, but it's not the only thing.

First programming experiences: Summer 1982: Turtle graphics, and PL/C (Cornell's subset of PL/1). Learnt, and forgotten, various languages, including APL. "Every shiny new thing doesn't have a good impact on society".

Why is high school hard? It's designed to sort students. Standard tests. Education is a political football and changes every 4..8 years, even though children are in school for much longer than that.

[Margolisetal "Stuck in the Shallow End"]. Computing Courses have no academic home. Culture of low expectations about prior knowledge etc.

Q From Chicago, class of 200, plagiarism.

A What are you asking the students?

Post Questioner, JHD etc. had useful discussion. He has an autograder, and basically the students are taught that passing this is the goal (not intentionally, but the questioner realised this is the message he is actually conveying.

5.4 Innovative Pedagogical Approaches to a Capstone Laboratory Course in Cyber Operations

Mike O'Leary, Towson University (near Baltimore). I teach this as a course. Second Semester senior year; 25 students. Isolated laboratory Taught for almost

15 years: these are my tricks. About 1/3 of my students end up "working for DoD in the Fort Meade area". I wrote my own book: "Cyber Operations" (Apress).

Flipped I don't say anything during the class. They hammer keyboards during the class, asking me when necessary. The responsibility for learning goes on the students. These people have to learn how to be professionals.

***** 12 weeks of "lectures" plus three live exercises. They have to build the network diagram (three subnets, web servers etc.) issued. I have an exercise control system. This issues keys etc. The class has to be both offense and defense. Use older software with known exploits. I have a list of 10K passwords they have to use, which makes brute force feasible.

Speakers I get 4–8 Red Team members, typically graduates from the 1/3rd. Many people from pen testing are willing to volunteer (recruiting!): the question is making clear what the aim is.

Q teams?

A Self-selecting with rules against repeats.

Q System?

A Exercise control I built myself: the students build on VMware.

5.5 Micro-Classes: A Structure for Improving Student Experience in Large Classes

Christine Alvarado, Mia Minnes, Leo Porter, UC San Diego.

Showed picture of a large amphitheatre. Taken from the back, showing "hiders". The largest class in the audience was 800, with 4 at 400, and 4 more (including JHD) at over 300. Good graph of teacher/student headcount. It's harder to hide in a small class, from the teacher and from each other.

We (UCSD) have 20 students/section, with 2 TAs and 5 tutors (undergraduates).

Call a group of 20 a microclass, with empty rows between them in the amphitheatre. During the active learning, the tutors/TAs must ensure that everyone is reached. TAs/tutors keep their assignment to a given microclass. Ran one of these at the same time as a baseline section: same instructor, activities as well. This was a Data Structures class, sophomores/juniors. It's a hard class.

so did it work? Various evaluations, including "classroom community scale". Prior data showed similar data. There was also a control group CTRL2 from a previous semester. But the fraction of women was 23.9% rather than 13.3 and 15.5 in the controls. Microclasses "felt connected" at 50.6, versus 47 and 46.8 in controls. In student evaluations, Micro was slightly better than CTRL2, but

significantly more than CTRL1. Even things like "is instructor well-prepared" scored better with Microclasses. But no difference in performance/ retention.

Conjecture: we didn't see improvement as the baseline is already pretty good. Question: what would have happened if we had tried this with first years.

Q How did you form the microclasses?

A Scheduling based on lab requirements

Q Were there outliers?

A Little pushback, but students could switch microclass.

5.6 Impact of Class Size on Student Evaluations for Traditional and Peer Instruction Classrooms

Soohyun Nam Liao, William Griswold, Leo Porter, University of California at San Diego.

How scalable is Peer Instruction? Unknown. Data taken from standard evaluation data at UCSD

RQ1 Now do PI and non-PI classes compare? Pi came out significantly better for engagement

RQ2 ??

Twitter question, individual vote, then group vote, then classroom discussion. 60 sections doing PI, 105 not. various subjects from CS0 to advanced data structures. 7 instructor-related questions, five course-related. Key question is "would you recommend".

5.7 My Digital Hand: A Tool for Scaling Up One-to-One Peer Teaching in Support of Computer Science Learning

Aaron Smith, University of North Carolina; Kristy Elizabeth Boyer, University of Florida; Jeffrey Forbes, Duke University; Sarah Heckman, North Carolina State University; Ketan Mayer-Patel, University of North Carolina. Increased enrollments are here to stay. especially CS1/2. Various tools, e.g. Blackboard exist to help with large classes. By Peer Teaching we mean experienced students helping new ones. Walk-in office hours. Note that this is scaling-invariant in terms of availability. But what about equity, logistics etc. We wanted transparent measurement of office hours. Student arriving at office hours complete

a form (computer/'phone) to "raise their hand", and then peer teachers would answer. The tool logged the interaction. Then asked for coarse-grained feedback.

Ran at Duke, UNC and NCSU. Duke had a different office hour structures. Identified excessive waiting times (Duke: 40% of students waited over an hour), long interactions (medians over 20 minutes), uneven allocation (5% of students used 50% of office hours time).

Therefore made students describe their problem, try to ensure repeat dates, count down timer for interactions (currently 10 minutes), peer tutor to suggest "road map" to student at end of interaction. `mydigitalhand.org`. We'd be interested in sharing data.

Q Views on this?

A Tutors really like the time-out: no feeling from students yet.

Q Students who need longer?

A The time-out is only a suggestion. Individual instructors can tell their peer tutors how strict to be.

Q Question description?

A Can choose from a dropdown of previous student descriptions

Q Vague descriptions?

A We don't filter automatically, and some students write AAAA.

Q Anonymity?

A None. But students can't see who else is waiting.

5.8 A Pedagogical Analysis of Online Coding Tutorials

Ada S. Kim, Andrew J. Ko, University of Washington.

Various previous studies. But the tutorials studied are mostly research prototypes, and the studies are not focusing on the pedagogy of the tutorials What we need is holistic guidance for teachers, and questions for the research community. We analysed 30 tutorials. 5 genres: interactive tutorials; Web references, MOOCs, Educational Games, Interactive platforms. Personalisation (customisable to meet prior knowledge), utilisation (how learners could leverage their knowledge: good were educational games, conversely W3school had none), contents (W3school and Codecademy), organisation (bottom-up or top-down), context (Codecademy was good), learner actionability (Khan Academy was good, Intro to JS: Drawing and animation; Coursera less good), feedback (poor for web references), meta-cognitive learning (whether it teaches 'when'/'why' to

use some construct, rather than just 'how'), support for learning (additional materials outside the curriculum: Gidget was good and proactive) were the 9 dimensions used. Dense slide: see paper.

Most are bottom-up, with 'how' emphasis. Feedback tends to be shallow. We recommend most educational games, some interactive tutorials; Khan, Codecademy, Gidget.

Note that we were using binary judgments, and possibly biases. This was an analytical assessment, rather than a measurement of outcomes.

5.9 Lessons Learned in the Design and Delivery of an Introductory Programming MOOC

J Michael Fitzpatrick, kos Ldeczi, Gayathri Narasimham, Vanderbilt University; Lee Lafferty, Ral Labrie, Paul T Mielke, Independent Consultant; Aatish Kumar, University of Amsterdam; Katherine A Brady, Vanderbilt University. Authors 4-7 were *ad hoc* mentors who just emerged.

MATLAB (40% of audience identified). Fitzpatrick has taught since 2000, with over 2500 students. This is a 14-week course, but most MOOCs are 4–6. Endded up with 8. Covered most of the regular curse, but left out recusrion. The video lectures are scripted, but doubt sound like it. Jokes, intentional mistakes, but still multiple takes. Excellent lecturer. Videos varies from 30 to 60 minutes. Most MOOcs have minima assessment, which is unsuitable. We had six labs of eigt assessemnts each. Two goals: to ensure passing students can program, and keep better students engaged. I/O auotmated grading, no style etc.

In 2015, 80K active students, 18K watched all videos, 5,600 passed (2.3 of homework). 100K homework sets graded. 2016 was on Coursera, so numbers not comparable. Over 50% of students had a college degree.

- Based on real course.
- Mike gave all the videos (great lecturer)
- Graded assignments
- Mentors (one answered over 2000 questions)

`https://www.cpoursera.org/learn/matlab`.

Q Cheating/plagiarism?

A I really don't care.

Q Why did students drop out?

A Anecdotally, when the assignments start getting hard (lecture 3).

***** This isn't a bad drop-out rate.

Q Expensive parts?

A Videos (four days of Mike editing), assignments.

Q Autograder?

A Ran on own machine. Latest version is MATLAB online, though.

5.10 Employing Retention of Flow to Improve Online Tutorials

Ashok Basawapatna, SUNY College At Old Westbury; Alexander Repenning, University of Applied Sciences and Arts Northwestern Switzerland (Exemplary Paper).

"Hour of code" has massive exposure. 400M students have done one. "Flow" = skills and challenges balanced: balanced between boredom and anxiety. We want to bring back students who are drifting from flow to anxiety: that's our zone for intervention. Think of it as a Markov process. We want probabilities to be the same at each step. We release an activity to a lot of students, and look for deviations from the negative exponential model.

One challenge is common mistakes, but still only made by a minority of students. Just warning against it worries those who didn't make the mistake. We solved this via a critiquing system. Worked slightly better in US than Switzerland (conjecture: multiple languages??).

5.11 How to Collect, Analyze and Act on Learning Data in Computer Science Courses

Ananda Gunawardena, Princeton University. Absent due to a family event. Colleagues from NCSU presented.

5.11.1 Introduction

Change conversation from anecdotal evidence: see for example Section 5.7. Also another paper from the conference. There was a survey, JHD was in the 22% of "very large (> 150) classes". Issues from the room.

- Time (everyone)
- Poor quality data
- Baseline data (with an IRB to make sure you can publish it)
- Data presentation.

Noted that we need a goal, to tell us what to collect (and collect well).

A good research question is: interesting (to us, at least, possibly publishable later), answerable, repeatable (to make a comparison), measurable, appropriately scoped. Use a Goal–Question–Metric (e.g. download at noon on first day of week) methodology.

1. I want to know how long my students are spending on a problem?

A Time on LMS?

Q I can't currently collect that, but maybe I should.

Floor Immediate feedback autograder. Immediate and right; nothing for days, then soon right; really struggling; don't start until very late. Public tests and secret "release" tests. We release three secret tests a day, but early submitters are told how many they have failed.

Staff You don't necessarily have to get the survey etc. right the first time.

5.11.2 Collect

Note IRB is required for K–12. If you're going to publish, you need to start the IRB process *before* you start the collection. Note that BlueJay collects lots of data, possibly via a check-box. For normal education activities, you may be able to get a waiver from informed consent.

Piazza, LMS message boards etc. can provide data. Speaker uses a GitHub+Jenkins system.

5.11.3 Analyse

Distinguish between *a priori* and *post hoc* analyses. Example in R. Note that a Boolean vector can be counted/averaged (TRUE=1). Standard data manipulation and subsetting. Must `install` a library first, then `library` command. Note the `lm` (linear model) function.

5.11.4 Discussion and Action Items

`EmpiricalCSEd.org` is where we work on disseminating CS research techniques. DEER = Designing Empirical Education Research. JHD: in XX10190, consider early returns on CW1.

Telling students what you find out is productive.

Chapter 6

Saturday 11 March

6.1 Codio (Supporter's session)

We developed Codio from the professional point of view, rather than educational. Hence the power of the system. Behind each project is an Ubuntu server (you can have as many as you want: no extra cost). We therefore support anything that runs here. The IDE is used by thousands of schoolkids in the UK.

People assume Codio can't do GUIs, but that's wrong: demo of virtual desktop. OpenGL, JavaFX etc. Demonstrates debugger: Python, C/C++, Java and more.

Demonstrates 'list of class' view: Blackboard, Moodle etc. integration. Can log in as a 'test student' as well as a 'teacher'. Can 'pin' current items to the top of the student view. Lecturer's like the ability to directly open the student project, rather than ask the student for their code (and iterate). When a student marks a project as 'completed', then can program an autograder script to fire.

"Advanced Autograde" allows the lecturer to write his own script: 10–20% of total usage. Example was testing HTML, where the DOM needs to be inspected. Manual grading supports moderation by sample as well. There can be grading rubrics. The lecturer decides when grades are released to the gradebook. Can specify start/end times for a specific assignment.

Shows Codio Guide on an introductory programming course (Python).

Q Can the students run unit tests?

A Yes.

Q–JHD Can one do hybrid manual/automatic grading?

A Yes.

Q Feedback into the students' code.

A Not yet? Feature coming where teacher can highlight code with a comment

Q App?

A Looking at a mobile app for the content side. We also have strong plagiarism detection. Talking to Columbia about looking at development speed as a means of checking for plagiarism.

Q XSLT?

A If you can install it, then fine. No native support from us.

Q Versioning?

A Yes, and warnings that certain structural changes will impact student work?

Q Multiple routes through?

A Not really — we're looking at adaptivity.

Q Can I use my own hardware box? How do my usernames relate to yours?

A Yes, it's a fully web-based setup. You configure the cluster. Also people doing DS let students play with small boxes, and have a larger one for tests.

Q Can you put LaTeX, or PDF, in the guide?

A Not directly, but can link to them.

Q Batch load existing questions?

A No.

Q Plagiarism detection across multiple sections?

A Not yet, and equally it doesn't go back to previous instances of the same class.

Q Cost model?

A Varies. Say 60-80$ per student per year if buying 100student license, which would allow 100 students in each semester.

6.2 On the (Mis) Understanding of the "this" Reference

Noa Ragonis, Beit Berl College; Ronit Shmallo, SCE-Shamoon College of Engineering.

Why "this": difficulties in understanding it, we can program without it, but we are accustomed to using it.

But the use of `this` is mandatory when parameters and attributes share the same name.

There is much research on misconceptions, e.g. object/class. But not directly on `this`. Hence our research: questionnaires for teachers and students. 86 high school students form seven school. 31/45/10 in 10/11/12 grades.

1. A project is presented with a simple class, a composed class and a main class.
   (a) mark where `this` is required in simple class
   (b) and in composed class
   (c) develop a static method replacing an instance method using `this`
   (d) can `this` be used in the main method
2. using `this` as a parameter
3. Using `this` as a method name to call ...
4. Personal taste: four codes, all stated to be correct, to be ranked in order of preference, and state criteria used.
5. Open comprehension: when must `this` be used (45% correct), ..., "what is `this`" (40% current object, 21% referral to attribute, ...).

Used 48 high-school teachers to categorise the students' answers. Many answers, clearly some never taught in school. Common reasons were legibility and convention (given for more that one alternative: ?JHD different schools).

Even though told that all version were correct, some students would say "doesn't work". Note that the teachers weren't always correct (72–83% correct): see paper.

Students tend to believe that `this` must be used, and beliefs about what will happen if it isn't. We therefore recommend that `this` not be used in teaching until a full understanding is achieved. You can get our tool and experiment.

6.3 Assessing and Teaching Scope, Mutation, and Aliasing in Upper-Level Undergraduates

Kathi Fisler, WPI; Shriram Krishnamurthi, Preston Tunnell Wilson, Brown University. Work done at WPI.

```
void m(SomeClass o) {
   int x = 3;
   int y = x;
   x = 4;   # ask students  what y is
   o.f = 5
   }
   p = new SomeClass(8);
   m(p);
```

and similar examples. Variables are not aliased, but objects are. This confuses students. Lots of research about these issues in CS1. But what happens in upper levels, when students have seen several languages? This study was in upper-level course on programming languages and their design. Questions in Racket ($\approx$ Scheme) and Java. They had all had Racket as CS1, Java in CS2 and life. Gave this s a pre- and post- test for the course. showed typical questions, testing aliasing and scope (easier to write in Racket).

Racket: pre 66%, post 88%; Java went 65% to 69% (insignificant). Bizarre, as most of them hadn't seen Racket since CS1. Scope questions went 66% to 86% (mostly Racket), Variable Mutation went 67% to 85%, and Parameter mutation went 62% to 70%.

In the course, all programming was done in purely functional Racket. They had to implement an interpreter for a static scope/closures language. We made them write (and graded) the test suites. Also a black-box, "what is this compiler's rules", question. Bizarrely, aliasing in Racket got better even though wasn't exercised in the course. Bizarrely 5 of the 9 with poor post scores had high interpreter scores.

So how effective are our teaching methods about programming languages? We are *not* getting "writing an interpreter really helps". Aliasing has a big impact on parallelism, which isn't in CS1/2. But we note that all their experience hasn't cleared away the misconceptions. We need new techniques to address misconceptions.

6.4 Multiple Levels of Abstraction in Algorithmic Problem Solving

David Ginat, Yoav Blau, Tel-Aviv University

Example 10 *Given N numbers, find greatest difference. Naive solution was $O(n^2)$ time and $O(n)$ space. Good one (*$\max - \min$*) is $O(n)/O(1)$.*

Example 11 (Queens) *Given K queens on $N \times N$ board, find number of threats. Naive (allocating entire board) is $O(KN)/O(N^2)$. Alternatively, sort queens by rows, then columns then diagonals. $O(K \log K)/O(K)$. But error-prone. But counting by rows/columns/diagonals is $O(N)/O(N)$.*

Chapter 7

Closing

7.1 Introduction

Introduction: Klaus Schwab at Davos predicted that robots would displace many professions, such as lawyers and doctors. What of CS teachers?

Papert was Lego professor at MIT. Resnick is Lego Papert professor at MIT.

7.2 Talk: Fulfilling Papert's Dream (Mitchel Resnick)

Papert passed away last year. This is the 50th anniversary of Logo. Note all this is pre-PC, and Papert believed in letting children use these very expensive computers. Hence they could see, via the turtle, whether they had it right or not. Children could learn about design. This has led to all the work we are doing with Scratch.

Papert's Dream is still unfulfilled. Papert was a strong believer in fluency. Hence we need a computer for every student. I am worried by "CS for all" as it's not about the specific discipline, but about thinking. Children should see themselves as creators. No-one would ridicule "Writing for all", so what's wrong with "Coding for all"?

Showed video of what one child had created. Managed to do scrolling backgrounds. But also led crowd sourcing of drawings etc., so grew as a person. Projects; Passion; Peers; Play are our guiding principles. We want to develop the creative thinker. There's nothing wrong with puzzles, but, just like crossword puzzles, they aren't the whole answer. Showed an interview with Khan, saying that he can change the directions of tens of thousands of children by the wording on a badge. Papert often said "Low floor, high ceiling" (the diagram also showed "wide walls"). Want to move away from transmission model of learning. We launched the Scratch community at the same time as the language. Minsky on Logo: "nice language, pity there's no literature". Video of children experimenting with Lego. Scratch 3.0 will have an API for adding new

sorts of blocks. Shows a voice recognition service. Closing song: “Give P’s a chance”.

7.3 Q&A

Q I’m from Taiwan. Learning English with Scratch. What about people who are being forced to learn programming, as a pathway to jobs.

A kids don’t necessarily like writing essays. Writing is a pathway to jobs for many, but far from all.

Q Will Scratch 3.0 have functions?

A Scratch has procedures. We can’t yet make functions sufficiently intuitive.

Q “beginners need structure and experts need freedom” is often said.

A I totally disagree. False dichotomy. Need an appropriate mix. We have tutorials, but some kids skip them successfully. Educational institutions tend to be too hot on structure.

Q Will Scratch 3.0 have more sharing facilities, e.g. Google Docs.

A Synchronous sharing is technically doable. We have worries about safety issues. In Scratch everything is public.

Q (same) Could teacher accounts have this?

A Verification of ”teacher” internationally is a challenge. We hope some-one else will solve that problem.

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.

[BB78] J.S. Brown and R.R. Burton. Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science*, 2:155–192, 1978.

[BBGS05] A. Bolotov, V. Bocharov, A. Gorchakov, and V. Shangin. Automated First Order Natural Deduction. `https://www.researchgate.net/publication/220888450_Automated_First_Order_Natural_Deduction`, 2005.

[RK06] H.L. Roediger and J.D. Karpicke. The power of testing memory: Basic research and implications for educational practice. *Perspectives on Psychological Science*, 1:181–210, 2006.