On Use of Theory in Computing Education Research

Greg L. Nelson
University of Washington
Allen School, DUB Group
Seattle, Washington
glnelson@uw.edu

Andrew J. Ko
University of Washington
The Information School, DUB Group
Seattle, Washington
ajko@uw.edu

ABSTRACT

A primary goal of computing education research is to discover designs that produce better learning of computing. In this pursuit, we have increasingly drawn upon theories from learning science and education research, recognizing the potential benefits of optimizing our search for better designs by leveraging the predictions of general theories of learning. In this paper, we contribute an argument that theory can also inhibit our community's search for better designs. We present three inhibitions: 1) our desire to both advance explanatory theory and advance design splits our attention, which prevents us from excelling at both; 2) our emphasis on applying and refining general theories of learning is done at the expense of domain-specific theories of computer science knowledge, and 3) our use of theory as a critical lens in peer review prevents the publication of designs that may accelerate design progress. We present several recommendations for how to improve our use of theory, viewing it as just one of many sources of design insight in pursuit of improving learning of computing.

CCS CONCEPTS

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

KEYWORDS

Theory, cognitive load theory, design, design science, domain-specific theory, peer review, publication bias

ACM Reference Format:
Greg L. Nelson and Andrew J. Ko. 2018. On Use of Theory in Computing Education Research. In *Proceedings of 2018 International Computing Education Research Conference (ICER '18).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3230977.3230992

1 INTRODUCTION

As a field of inquiry, computing education is broadly concerned with searching for designs that improve both the learning of computing and broader effects of that learning. Our community investigates a wide range of design ideas including programming IDEs [25], tutorials [28], learning trajectories [52], curricula [4], pedagogy [51], teacher training [15] and more. Our community also considers diverse broader effects of learning, from gaining knowledge measured by tests [60], developing identity [23], improving task performance [31], and gaining expertise [29], to more humanistic and social concerns such as motivation [33], self-efficacy [10], improved quality of life [20], and equity in outcomes and society [47].

To achieve these design goals, our field is increasingly using theory, both to help find better designs and to help interpret their effects on the world. For example, in the 2017 ACM International Computing Education Research Conference (ICER) proceedings, papers drew upon theories of identity development [24], affect [30], cognitive load theory [34], distributed cognition [9], and collaborative learning [19]. This rich body of theoretical work from other disciplines has offered a foundation upon which to derive hypotheses about the classes we teach, the tools we build, and the explanations of computing we form, and to help us interpret what happens when we share them with learners. In addition, in principle, if a theory correctly predicts that a design will be poor, we can avoid building and evaluating it, and instead pursue designs predicted to be better.

Simultaneously, we are also increasing the role of theory in peer reviewing of our research. The reviewing guidelines for most of the major computing education research conferences and journals (for example, ICER, ACM Transactions on Computing Education, and the Computer Science Education journal) explicitly assess the application of theory in both empirical studies and designs. As criterion, the incorporation of theory in our field's design inquiry has great potential to increase the scientific rigor of our discoveries.

In this paper, we argue that while theory *can* accelerate our field's progress and increase its rigor, if not used carefully, it can also inhibit progress in subtle but important ways. We will focus on three ways that theory can inhibit design progress:

- By using theory to explain learning phenomena *and* pursue better designs, we may create tensions between breadth and depth that limit our impact in both pursuits.
- By focusing on general theories of learning, we may overlook the need for domain-specific theory about the content of computing that is so critical to accelerating design progress.
- By using theory as a critical lens in peer review of designs, we may create a publication bias that inhibits both theory evaluation and our search for better designs.

In the rest of this paper, we will outline what theory is, how we use it in computing education for explanation and design, and then examine our three critiques of its use in detail. Throughout, we will use Cognitive Load Theory [57] as an example of a theory that our field has used to provide clear benefits to our search for

ICER '18, August 13–15, 2018, Espoo, Finland

ACM ISBN 978-1-4503-5628-2/18/08.
https://doi.org/10.1145/3230977.3230992

better problem solving instruction, but that may also be impeding progress. We end with a series of concrete recommendations for how to better use theory[1] in computing education research in our shared pursuit of better learning.

2 USES OF THEORY IN COMPUTING EDUCATION

In academic research broadly, a theory may include terms, framing, causal mechanisms [18], arguments (a collection of related claims), and a body of evidence that supports or contradicts those arguments [53]. For this paper we define a *descriptive* theory as including terms that name and describe phenomena (called "analysis" theory by [16] and [7]). We also define an *operational* theory as one with procedures for measurement of some terms as well as arguments and evidence for their validity. For example, Danielsiek et al. [10] operationalizes self-efficacy for an algorithms class by creating a survey and studying the survey to validate that it measures self-efficacy.

In computing education research, we use both kinds of theory for design inquiry to describe, generate, and predict the effects of designs. Theory can help *describe* designs; for example, we might summarize the role and content of a UI element in a design by calling it a "sub-goal label" and referencing the worked examples and cognitive load theory literature that created that term [2, 56]. Theory can help characterize *design spaces* that describe many possible designs, as in how Kafai and Resnick use constructionism to illustrate a whole landscape of learning settings and tools [22]. Theories can be also be used to *generate* designs and *predict* their effects. For example, in recent years, computing education researchers have leveraged Cognitive Load Theory (CLT) for genres of design and their effects [2]. For example, Morrison et al. adapted worked examples from other domains (such as physics [2, 56]) to worked examples for computing, adding sub-goal labels such as "initialize variables" and "determine loop condition" to convey programming problem solving [42]. Figure 1 summarizes how we may use theory during design inquiry to describe and generate designs.

Predictions based on theory are also used to provide *rationale* for design, by arguing for some design choices based on a (usually qualitative) prediction of the design's effects. For example, the design of the Gidget coding tutorial drew upon social psychological theories of in-group relation to reason that manipulating animal elements would engage learners more compared to inanimate objects; this provided a rationale for that design choice [27].

As Figure 2 depicts, theories can also help us explore design alternatives. If theories can make reliable predictions about designs we have not yet fully built, they can help us evaluate designs without having to test them empirically. They provide this value by providing some explanation of the *mechanisms* behind a phenomenon that are useful for a design. For example, CLT is valuable in principle because it helps a designer reason about different types of

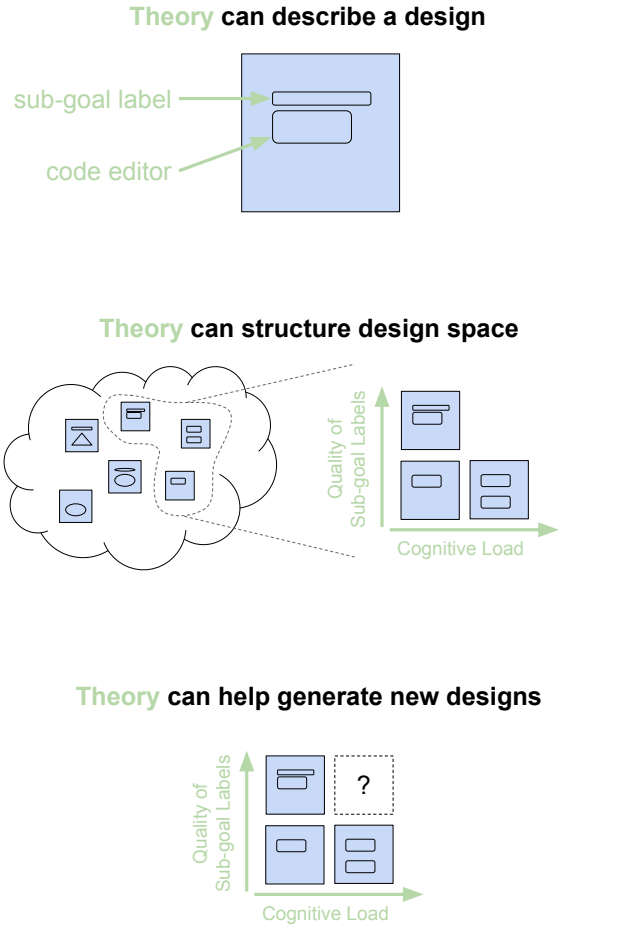

Figure 1: Theory aids design inquiry by helping us describe designs, structure the design space, and generate new designs.

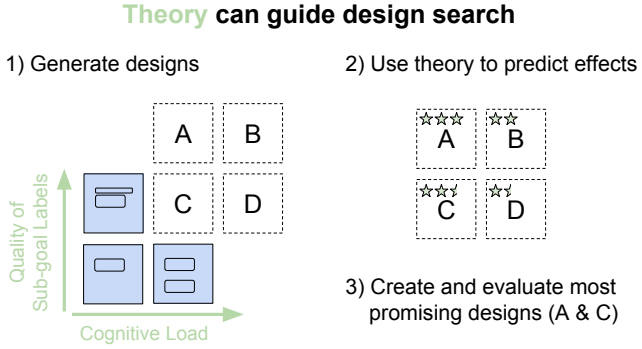

Figure 2: Our field may speed up our design search by prioritizing which designs to build and test based on theoretical predictions of their effects (if the predictions are accurate).

[1]We are writing this paper to the philosophy of science prior knowledge we expect most readers to have. This overlooks important and varied philosophical and epistemelogical issues; for example, Hedstrom et al. review the mechanism-based approach to explanatory theory [18], which is a major but only one among several approaches. We believe considering these issues would lead to stronger critical views on use of theory and higher standards for use of explanatory theory in our field (i.e. beyond the "we should consider falsifiability when choosing theories" we later argue).

load (e.g., intrinsic, extraneous, and germane) and make predictions about which type of load a design is imposing. If CLT's predictions are correct in general, and a designer's interpretation of a design's various kinds of load are sound with respect to the theory, a designer may predict their design's effects on load without empirically testing their design's effects. Thus, a designer should be able to more rapidly refine the instructional materials they design, as they don't need to wait for empirical testing as they iterate. Recent applications of CLT do exactly this, proposing more rapid methods of instructional design through theory [39].

Theories can also be useful to interpret why a design has failed. For example, CLT provides the concepts necessary to explain why the split attention effect [6] (in which multiple types of information presented in the same modality) interferes with learning. In the same way, when we test new designs for worked examples that have mixed benefits, CLT can offer useful concepts in interpreting why those benefits were mixed.

Within computing education, the use of CLT has offered all of these benefits to our collective search for effective worked examples of programming. It has helped us name and generate new points in a design space of worked examples for computing (e.g., [17, 35]) and it has helped us explain the outcomes of evaluations of worked example designs[2] (e.g., [13, 42]). In these respects, it has focused our community's design efforts, it has helped us build upon our own work, and it has helped us build upon more general theories of learning and education.

3 HOW THEORY CAN INHIBIT DESIGN PROGRESS

While theory has clearly been useful in computing education research, we have observed three ways in which it may also inhibit our search for effective designs for learning. In this section, we present these three forms of inhibition, then in the next section, provide concrete recommendations for how to avoid them.

3.1 Tensions between explanation and design

Ultimately, the goal of design exploration is *breadth* of investigation: by considering many design alternatives, and their numerous tradeoffs, designers are better positioned to find optimal ways for people to learn computing. In contrast, the pursuit of explanatory theory such as CLT promotes *depth* of investigation of design, identifying and measuring the causal mechanisms that explain learning outcomes and broader effects[3] of learning. Achieving this depth of explanation essentially requires holding designs more fixed, so we may deeply understand the mechanisms behind their benefits. Our community's joint interest in breadth and depth creates tensions and trade-offs in research. We will consider these tensions from both perspectives.

3.1.1 Design goals undermine explanation goals. To make high-quality tests of explanations, we need to make high-quality validated measures for them. The time we spend broadly exploring innovative designs takes resources away from building these validated measures. Consider, for example, the series of papers by Morrison et al. on worked examples. This work began with some investment in measurement depth by building a validated measure of cognitive load [44], but it was largely followed by broad design explorations [37, 42], at the expense of addressing the measurement issues that may have led to some of the inconclusive explanatory results reported in their papers (learning outcomes varied for the designs but their cognitive load had little to no difference). This balance of design and explanation that Morrison et al. chose is not inherently good or bad; rather, it is an example of the inherent tradeoff between pursuing deep explanations and innovative designs.

This tradeoff can also lead to problematic split attention. To properly build upon the bodies of evidence related to design and explanation, we must apply limited resources to reading papers about both domain-specific design innovations we publish in our own community, but also papers on the general theories being refined in learning and education research. Because our time to read is limited, every paper we read about design is a paper we do not read about the explanatory theories that inform design. While as researchers we might hope we take time to read deeply, the fact that we need to read two bodies of knowledge can change what we even perceive as sufficient depth, given the reality of our limited resources.

One concrete manifestation of this split attention on reading is that we miss ongoing critical debates about the theories we choose to use. For example, there is active debate about CLT in the learning science community about CLT's falsifiability. (A theory is falsifiable if someone can conduct an experiment and, in principle, observe outcomes that contradict predictions by the theory, thus demonstrating the causal mechanisms cannot explain those outcomes [50]). Recent work in the learning science community argues that cognitive load theory is actually not falsifiable[4] [11, 41]. If a theory is unfalsifiable, experiments with explanatory goals may always appear to validate the theory, because an unfalsifiable theory cannot be disproved (at most, results might appear inconclusive). Falsifiability should thus be an important factor in choosing which theories we use[5]. We have found no papers from our community that cite or discuss CLT's falsifiability debate among the 22 papers published

[2] Ericson et al. [13] is also an example of design-based research, a research method that tries to balance iterative design exploration and theory.

[3] See introduction paragraph 1, e.g. identity development, equity, self-efficacy.

[4] CLT (which can only measure *total* cognitive load with independent validity) is unfalsifiable because it can explain any observed outcome (see [11, 14, 41] for fuller arguments and other difficulties). To illustrate, if an experiment compares two designs and observes higher cognitive load and higher learning outcomes for one design, a researcher can suggest that there was higher intrinsic load to explain outcomes. If the observation was higher load and lower outcomes, one can suggest there was higher extraneous load to explain outcomes. For lower load and lower outcomes, one suggests there was lower intrinsic load, and for lower load and higher outcomes, one suggests there was less extraneous load. Thus, CLT can explain any observable outcomes of total cognitive load and learning outcomes. If a measure could distinguish between the components, the theory could be falsified; however, all validated ways of measuring cognitive load components assume their relationship with learning outcomes during their development and validation [14].

[5] Popper's notion of falsifiability is seen as problematic among most present-day philosophers of science; for example, the Duhem-Quine thesis argues that even if an experiment's results contradict a theory's prediction, one cannot deduce the theory is flawed (perhaps the design used was not analyzed properly with the theory, perhaps the experimental design was flawed, etc.). Beyond falsification, more nuanced sociocultural perspectives have richer notions of scientific practice [26]. We leaves as important future work deeper engagement with these philosophical issues (including explanatory theory and issues with naive empiricism), which is critical for informing recommendations on appropriate theory use in our field. For an example in another community, the papers criticizing CLT engage with these issues and use more modern perspectives [11, 14, 41]. On the limits of falsifiability, see [21] as an accessible example.

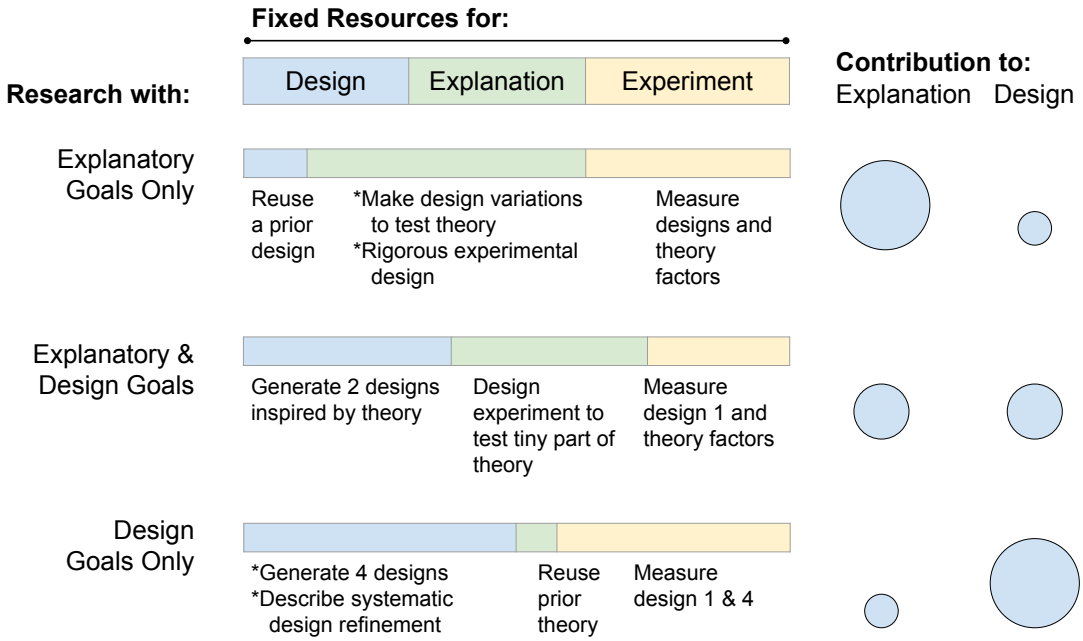

Figure 3: Splitting attention between design goals and explanation goals may lead to work with shallower contributions and less overall contribution.

in our community on CLT (counted via a search of the ACM Digital Library under SIGCSE's conferences for "cognitive load").

Splitting our attention between design and explanation goals may also lead to missed opportunities to do deeper explanatory work, as our community also has not engaged in such debates. Compared to the 22 papers published in our community on CLT, we only found two papers by computing education researchers who published in the learning science communities about CLT [35, 37] and two in the educational psychology community[36, 38]. This engagement has come mostly in the form of confirming CLT, and not in sharing contradictory evidence with the broader learning science or educational psychology community to inform the broader falsifiability debate. For example, CLT generally explains differences in learning outcomes via differences in cognitive load; it generally predicts higher cognitive load will reduce learning. However, in our community, three studies measured learning outcomes and cognitive load with a validated instrument for different designs [44]. They saw differences in learning gains with little to no difference in cognitive load, and reported those to our community, with some potential theoretical explanations framed within CLT [34, 42, 43]. We are unaware of work sharing these conflicting results with the broader learning sciences community, nor the educational psychology community.

Again, we are not arguing that these authors' individual choices of how and whether to engage in this debate are inherently good or bad. Rather, it reveals a fundamental tension between the goals of design and explanation, and how that tension leads to missed opportunities for deeper explanatory work on CLT.

3.1.2 Explanation goals undermine design goals. Taking the opposite view, when we pursue deeper theoretical explanations, we tend to limit our design exploration. This occurs for a few reasons. First, because the pursuit of deeper theoretical accounts of a design's effects can be done with *sufficiently effective* designs, there is no incentive to pursue *optimally effective* designs. For example, the community's work on CLT *can* be done with the worked example designs originally published by Morrison et al. [42], and so it largely has been, at the expense of more radical interpretations of what constitutes a worked example for programming problems.

A second way that explanatory goals inhibit design exploration is that we may not explore designs because it is unclear theoretically why they might work. This deters the exploration of designs that do not yet have a theoretical justification, even though those designs might be objectively better in a way we cannot yet explain.

In summary, our community's joint interest in breadth for design and depth for explanation creates tensions and trade-offs in research, given our fixed resources. Figure 3 illustrates these tensions by contrasting hypothetical research examples with explanatory or design goals only, and a mix of both.

3.2 Less effort towards domain-specific theory

One effect of our community grappling with the tensions between explanation and design is that we may contribute less time to advancing our own domain-specific theories of computer science learning. For example, these theories might include theories of what it means to know a programming language, what it means to know how to program, what it means to be an expert software engineer, what it means to have computer science literacy, and numerous other unanswered and yet foundational questions that are specific to computing education. These theories are critical to design progress in our field, because, without them, we cannot

know if we are making progress on our shared goal of helping people acquire and use such knowledge and expertise.

This lack of attention on domain-specific theory is particularly acute because of our community's limited research resources. While we have applied resources to making domain-specific validated measures for constructs from general theories like cognitive load [44], the volume of this work, in past decades, shows we have mostly applied resources towards general theories of learning and identity, while overlooking major gaps in our domain-specific theories of the content of computing knowledge. This trend is clear in Malmi et al.'s literature review of theory use covering 2005-2011, which found that we use external theories frequently [32]: of papers referencing theory, 60% reference theory from Education, Psychology, or other parent disciplines, compared to 16% using theories created from within computing education.

While some use of general theory may help us, it may also divert resources from one of the most critical gaps in our field: domain-specific theories of the knowledge of computing we want people to learn. Little work has proposed such theories; only recently, Nelson et al. have proposed one for program tracing knowledge [45], and Loksa et al. have proposed one for programming problem solving [31]. Some theoretical work (such as [54]) has considered program dynamics as a threshold concept, proposing a mechanism to explain the large variation in outcomes for learning programming; this kind of nascent theory of knowledge may develop with more work into part of a theory relating conceptual knowledge and the ability to learn programming skills. Other partial work towards a theory of knowledge includes work on learning trajectories (paths from initial knowledge to deeper knowledge), though recent work still lacks direct observations and instead is based on aggregating results from the scholarly literature [52].

Not only do we lack theories of computing knowledge, but critical for evaluating designs, we also lack robust, validated measurements of this knowledge. Some work has advanced towards concept inventories for computer science [12, 46], drawing inspiration from work in other domain-specific research communities such as physics [58]. Concept inventories attempt to validly assess core topics rather than covering a broad theory of knowledge in an area, so they might also act as steps towards broader theories of knowledge. Within this genre the FCS1 [12] and SCS1 [46] are major points of progress, as they show some feasibility for language-independent validated assessments. Their validity argument extended a classic approach in education research to model knowledge that builds on existing teaching practices - 1) start with how a subject is taught today (here, CS1 textbooks), 2) synthesize a conceptual theory of the knowledge [59], and 3) validate the theory and a test specification for it with a panel of experts. This socially validates that the FCS1 and SCS1 measure something like what CS1 textbooks cover and CS1 finals and midterms measure, and represents a sophisticated validity argument and methodology that education researchers have taken years to develop.

While our community can make progress by using such methods and conceptual domain-specific theories of CS knowledge, when we build conceptual theory starting from existing practice, we somewhat assume that current practice actually works. However, it might be that some learners "figure out" tacit unsaid knowledge to succeed (for example, in CS1), which is not actually covered in textbooks or the class itself. This can lead to gaps in our domain-specific knowledge theories that are hard to see.

As CER researchers we might lead education research by representing knowledge beyond the conceptual level, as programs or with other formal representations; these more formal domain-specific theories might potentially lead to better assessments and designs for learning. For example, we have tried building a nascent formal theory of program tracing knowledge as knowing all paths through the compiler program for a language [45]; this uses a formal programming language to represent the knowledge. One can argue for validity of a formal knowledge theory by showing a computer can use that representation to perform tasks that require that knowledge (for example, we know that a compiler program can trace programs because we use them all the time to execute programs). This forces the knowledge theory to be complete and potentially more decomposable; it is less clear when a conceptual theory is truly complete. Formal domain-specific theory might inform assessment creation; for example, our nascent formal theory of program tracing knowledge could inform a more complete assessment for program tracing by requiring test questions that cover each path. Formal domain-specific theory may also inform design; for example, we also used our nascent formal theory to gain insight into what knowledge needs to be shown to learners, leading to a new design for teaching tracing that shows more detailed connections between AST tokens and program execution [45]. These examples are just for tracing, and we hope the community will work on others as well.

Ultimately, without domain-specific theories of computing knowledge, we cannot know when we are making progress on design. We need these theories and measurements to compare the designs we create across standard measurements. We need validated measurements to be confident in our results. And we need theories of the knowledge to make clear to both researchers and teachers precisely what knowledge we are helping learners acquire. With our community's limited resources, when we choose to advance non-domain specific theories, we make less progress on our critically necessary domain-specific theories.

3.3 Publication bias in peer review

While theory can bias what work we choose to do and what work we read, it can also bias how we evaluate each others' work. In particular, we argue that when theory is used as a critical lens in reviewing publications about innovative designs, we are at risk of rejecting papers that describe designs that may be valuable but cannot yet be explained theoretically. This ultimately reduces the breadth of design exploration as well as the base of published evidence for building and testing new or alternative explanatory theories.

To substantiate this publication bias, let us consider one case of a paper being rejected for its use of CLT as a theoretical framework. Because these are the critiques of just one case, we are *not* arguing that these peer review criteria are pervasive in our community. Rather we use this case to highlight the potential for more widespread publication bias, if such rationales were to become prevalent. We will show that the negative consequences of such bias are sufficiently severe that a single case of the review and meta-reviewing process failing to catch this is cause for concern and discussion.

In our original submission, we shared excerpts from reviews that an anonymous author had shared with us for use in this analysis; none of the authors of our paper were on or involved in that work. However, the ICER program chairs would not let us list quotes from these reviews or paraphrased quotes out of concerns for the potential impact on reviewers, and so we explain the reviews' arguments here in our own words.

The reviews concern a paper contributing a system that used a design genre from CLT (worked examples), prototyped a new design, and performed two studies: 1) a randomized comparison with a large sample size of the system vs. existing practice materials that had mixed results and 2) a large in-the-wild self-selected "open to students to use" evaluation showing positive effects. This paper was submitted as a CS education research paper to ICER twice, SIGCSE once, and rejected each time. There were not substantive issues with the paper's evaluation, in fact one meta-review noted a consensus among the reviewers that the experimental design was good, with random assignment and a large sample size.

This paper was rejected for several reasons, many of these reasons using CLT as a critical lens. One CLT critique in the reviews was that *a theoretically-framed design evaluation must produce results that support the theory*. More specifically, in this case CLT predicted a design with worked examples should do better, but instead the empirical results of the paper were mixed. Reviews praised the study design and the paper's aims to build on theory, noted the paper would have been a good "reproducing" experiment if it had positive results, and ultimately emphasized the results were only promising and that the system's design seemed different from work that had found positive results.

Rejecting work with this rationale impedes both design and theory progress. First, if peer review requires that a design framed from a CLT perspective must produce positive results in support of CLT, we will never share evidence about designs that provide evidence *against* CLT, losing insight about how the theory might be refined. Second, if we do not publish designs that work poorly, our community risks wasting resources on re-implementing and re-evaluating designs that have already been explored. Third, sharing novel designs with non-optimal effects also may inspire designs with improved effects. Finally, this critique undervalues studies that compare existing designs, even when those comparisons reveal no differences.

A second CLT critique was that *a design should be consistent with the recommendations of the theory*. More specifically, in this case the logic was that, since the design followed some recommendations from CLT, but it had mixed results, the design should be revised to follow all the recommendations and re-evaluated. Reviews noted the system's design lacked sub-goals and self-explanation steps and did not seem to fit recommendations that prior research made for designing worked examples. Reviews went so far as questioning if the system's "worked examples" met theoretical criteria for being a worked example, and raised issues with interpreting the results as related to CLT and worked examples.

If we were to follow this criteria in peer review, we would not publish designs that contradict theories, which would otherwise allow us to refine or reject theories via new evidence. Additionally, rejecting papers that explore new design variations that go beyond a theory's current recommendations prevents us from discovering designs that might be superior to existing ones and from generating new and improved theories. This policy thus shrinks the space of design search to only designs compatible with existing theories, and precludes the refinement of CLT. In fact, the critique that *a design should be consistent with the recommendations of the theory* means that a design must exhaustively meet the recommendations of theory in order for negative results to be publishable (otherwise, the reviews will say to "improve your design and resubmit"); this creates a potentially insurmountable burden of proof for publishing conflicting evidence.

A third critique was that *an evaluation that does not provide a causal account of positive results is not worth sharing*. Reviews generally praised the methodology for the randomized part of the evaluation with mixed results, which had a stronger causal interpretation. In contrast, reviews criticized that the positive results in the observational evaluation came from a self-selected group of learners, while acknowledging the paper highlighted that fact in its discussion. Reviews expressed concerns focusing on potential confounders to causal interpretations of results: 1) learners opted-in to using the system for extra practice, making it vulnerable to self-selection bias, and 2) observed differences might come from the system or other factors, such as from unmeasured individual factors such as prior knowledge or field of study. Reviews also wanted more causal interpretability for the evaluation as a whole, criticizing that the system was only studied at one institution with its own potential contextual factors, the analysis explained little of the overall variation in exam scores, and the potential for a ceiling effect in the evaluation tasks. The potential for misinterpretation of the observational results by readers was also raised in the reviews.

This critique inhibits design exploration in several ways. First, it prevents the sharing of designs that may produce real effects of learning, but that cannot yet be explained by existing theories[6]. This biases our evidence base toward only designs with well-explained effects. Second, only publishing designs with a clear benefit (rather than just promise) creates a harmful incentive to design simpler interventions with measurable but incremental gains, rather than exploring more radical, innovative designs that have the potential to improve learning, but may take multiple publications to refine and systematically measure benefits. Third, many designs may actually be deployed for discretionary use (a self-selected setting) or even designed specifically for discretionary use; devaluing empirical studies of discretionary use creates barriers to building design knowledge on how to improve the effects of learning in discretionary contexts.

These three uses of theory in peer review —reviewers judging that a) a theoretically-framed design evaluation must produce results that support the theory, b) a design should be consistent with the recommendations of the theory, and/or c) an evaluation that does not provide a causal account of positive results is not worth sharing — ultimately inhibit the breadth of design exploration, which, in turn, may inhibit the learning gains and broader effects of learning we can achieve through better design. They also severely limit our ability to falsify theories that are actually inaccurate and

[6] This is particularly an issue for broader effects of learning, see introduction paragraph 1, e.g. identity development, equity, self-efficacy.

impede our ability to evaluate, refine, and build new theory by restricting the available evidence.

Fundamentally, these peer review critiques confuse the relationship between theory and a body of evidence: a body of evidence tells us if a theory is meaningful and useful, not the other way around. Observations of designs and their immediate and broader effects are *inherently* valuable to progress on improved designs, whether they are consistent with existing theories or not.

While we present only one actual paper here (as we cannot sample or search all peer reviews), unless we resist these uses of theory in peer review, they have the potential to seriously restrict and bias the base of evidence used for creating and evaluating theories, as well as our community's search for better designs.

4 IMPLICATIONS

While theory can help our search for effective designs, we have argued that it can also inhibit design progress by creating tensions between design and explanation, lowering investment in domain-specific theories of computing knowledge, and biasing peer review. In this section we make recommendations for how to avoid these problems.

4.1 Focus on design and use theory as a guide

First, we believe our community should wholeheartedly commit to focusing on design and not on refining general theories of learning. First, general learning researchers will be best at refining general theories of learning. Second, our community has carefully cultivated skills and knowledge for creating domain-specific designs that improve computing learning outcomes and broader effects of learning computing. Third, other learning research communities lack these skills, and are therefore unlikely to make design advances in computing. In summary, our skills uniquely position us to make designs to improve effects[7] of learning in our domain; others cannot do this work.

By committing to design, we can focus our use of theory on guiding design. We should use theory as a guide for selecting designs to explore, as just one of many sources of insight to guide design. These sources include: prior designs (even in other domains), theory, intuition, experience, learner and teacher insights, and iterative and qualitative methods. Design-based research methods from the learning sciences use theory as a guide and include iterative testing of design hypotheses [1, 3, 5, 8]; these methods may also help bridge domain-specific education research and broader education research [48]. However, this method can favor starting with existing theory, which may limit radical design exploration and the creation of new theories. As our community explores these and other methods [40, 49, 55], we should not privilege conformance with or explanation via theory, as this reduces the potential positive impact of other ways to guide design. If someone made an argument based on strong evidence that theory performs better as a guide, perhaps we should privilege it relative to other ways to guide design, but not before we consider and debate that argument as a community, and not at the expense of other valuable sources of insight.

[7]Broadly defined, see introduction paragraph 1, e.g. identity development, equity, self-efficacy.

In our use of theory as a guide, we should explicitly read ongoing debates about theory in their communities of origin, so that we use theory with full awareness of its strengths and weaknesses. We should incentivize deeper reading by encouraging papers that take a large body of work on one or more theories and distills it to make it accessible and actionable for our research on designs and to guide design search. We should extend the page lengths of our archival venues to allow for discussion of such nuance.

4.2 Invest in CER-specific theories and validated measures for effects of learning

In addition to focusing our use of theory as a guide, we should recognize that improved effects of learning are the ultimate guide and indicator of progress for our community's design search. Therefore, we should invest significant resources in building domain-specific theories and measures of those effects (especially computing knowledge) so that we may use them as indicators of progress. These theories include: 1) domain-specific theories and validated measures of what people want to learn about computing, providing a map for the community's design efforts, 2) domain-specific theories and measures of effects of learning like self-efficacy, motivation, identity development, improved quality of life, and equity in outcomes and society (even when they are more difficult to measure and evaluate), and 3) theories of external validity, relating these measures to professional performance and achievement of wider goals that learners have, within and beyond computer science. Better measures of all of these effects (such as learning gains) may even help us gather more resources as a community, providing a return on invested effort.

4.3 Do not use theory or evaluation results as a barrier to publishing novel designs

As we use theory as a guide in design exploration, we should not use theory as a barrier to publication. We should at least publish novel designs with objectively promising benefits, regardless of their theoretical framing. This practice prioritizes building a broad design body of knowledge faster by removing publication barriers. There are three important reasons to do this. First, more radical, innovative designs that have the potential to improve learning may take multiple publications to refine and systematically measure benefits. Second, sharing novel designs with non-optimal effects also may inspire designs with improved effects. Third, if we do not publish designs, our community risks wasting resources on re-implementing and re-evaluating designs that have already been explored.

The same three reasons above actually support publishing *all* novel designs, not just those with measurable benefits. We should trust that our community will not try to game the system and make up novel designs for the sole sake of publication; those strategies will not work well in the long term anyway, as they will not result in convincing letters for tenure cases, job applications, or other career advancement. We might *worry* that such abuse will happen, but we *know* real harms occur from not publishing designs, as we argued above and as we showed in our section on publication bias.

Even more radical than publishing all novel designs, we believe we should not require all designs to have theoretical framing or

rationale. Requiring these actually inhibits both design exploration as well as the development and evaluation of theory, by restricting what designs we share. Moreover, it encourages researchers to generate unhelpful post-hoc design rationale just to satisfy a design rationale requirement. This can also cloud our understanding of how we actually design and what helps design in practice, which limits our ability to improve how we design.

One way to implement these recommendations is to create new tracks for work at publication venues. The publication threshold already varies by publication venue and track; for example, the "Experience Reports and Tools" track at SIGCSE and the "Innovative Practice" track at Frontiers in Education (FIE) invite work without formal experimental evaluations.

However, multiple tracks alone is not a solution to the issues raised in this paper. First, even within a track, the publication threshold may vary in an ad hoc way among reviewers, creating unseen publication bias that may harm our community's design progress. Second, even if we had consistent reviewing within tracks, separate tracks can encourage rejecting work that pushes boundaries or tries to improve an aspect of their contribution without meeting the bar for high quality work in that area; for example, having a non-randomized quantitative evaluation (instead of a "student's liked it" evaluation [61]) may make an experience report seem more like a research paper with a lower-quality evaluation. Third, reviewers may reject work for "being in the wrong track", authors may revise then submit it to another track, only to have it rejected there again for "being in the wrong track". While it is only one case, this actually happened to the worked examples paper we discussed in our peer review section (it was rejected from the Experience Report track at SIGCSE, and ICER twice). In summary, our community should discuss publication thresholds and how to help reviewers achieve consistency pragmatically, in order to improve design progress and build a broader unbiased evidence base.

4.4 Improve support for and audit peer review

We hope that these peer-reviewing biases in the previous section and in section 3.3 are not prevalent in our community, but, upon reflection, our community has little way to know this because peer reviews are not shared widely. Moreover, individual peer reviewers cannot see the aggregate effects of their critiques on our field. Therefore, to gain awareness of biases in our empirical knowledge base, our journal editorial boards and conference steering committees should conduct qualitative studies of the reasons that papers are rejected for publication, using best practices for such studies (such as multiple raters, calculating inter-rater reliability, and blinding each review's author and the paper under review). The results should also be given to individual reviewers for each specific paper they review, to check the results and so they can reflect on their reviewing process. The final anonymized data (and any conflicts) should be published publicly along with a paper interpreting the data.

In summary, we strongly recommend the computing education research community:

(1) Focus on design, domain-specific theories of learning, and validated measures of effects of learning.
(2) Publish work that distills theory from other fields to make it actionable for design and guiding design search.
(3) Publish all novel designs with some promise of improving outcomes.
(4) Publish all novel designs that future work might build on to then actually improve learning.
(5) Publish all novel designs that appear not to work well that other researchers might recreate, to avoid wasted effort.
(6) Conduct a periodic qualitative study of the critiques used in peer review in our community to detect and mitigate bias.

We face an urgent need to scale learning of computing in a changing world. While CLT and other theories have greatly matured computing education research, it is time to view theory as just one powerful way among many to guide our community's design inquiry.

5 ACKNOWLEDGEMENTS

This material is based upon work supported by Microsoft, Google, Adobe, and the National Science Foundation (Grant No. 12566082, 1539179, 1314399, 1314399, and 1153625). We thank the anonymous authors of the rejected worked examples paper and their candor in discussing their paper. We thank the reviewers for their deep engagement and constructive feedback, which we hope to incorporate more of in future work. We thank Briana Morrison and Mark Guzdial for discussions and doing some of the highest quality theory-based work in our field. We recognize the benefit researchers have brought to our community when using theory well and thank them for their contributions. Likewise, we thank you, the reader, for your time and care in reading (and critical discussion).

REFERENCES

[1] Terry Anderson and Julie Shattuck. 2012. Design-Based Research: A Decade of Progress in Education Research? *Educational Researcher* 41, 1 (2012), 16–25. https://doi.org/10.3102/0013189X11428813 arXiv:https://doi.org/10.3102/0013189X11428813
[2] Robert K. Atkinson, Sharon J. Derry, Alexander Renkl, and Donald Wortham. 2000. Learning from Examples: Instructional Principles from the Worked Examples Research. *Review of Educational Research* 70, 2 (2000), 181–214.
[3] Sasha Barab and Kurt Squire. 2004. Design-Based Research: Putting a Stake in the Ground. *Journal of the Learning Sciences* 13, 1 (2004), 1–14. https://doi.org/10.1207/s15327809jls1301_1 arXiv:https://doi.org/10.1207/s15327809jls1301_1
[4] Julian M Bass, Roger McDermott, and JT Lalchandani. 2015. Virtual teams and employability in global software engineering education. In *Global Software Engineering (ICGSE), 2015 IEEE 10th International Conference on*. IEEE, 115–124.
[5] Philip Bell. 2004. On the Theoretical Breadth of Design-Based Research in Education. *Educational Psychologist* 39, 4 (2004), 243–253. https://doi.org/10.1207/s15326985ep3904_6 arXiv:https://doi.org/10.1207/s15326985ep3904_6
[6] Paul Chandler and John Sweller. 1992. The split-attention effect as a factor in the design of instruction. *British Journal of Educational Psychology* 62, 2 (1992), 233–246.
[7] Tony Clear. 2011. Doctoral work in computing education research. *ACM Inroads* 4, 2 (jun 2011), 28. https://doi.org/10.1145/2465085.2465092
[8] Allan Collins, Diana Joseph, and Katerine Bielaczyc. 2004. Design Research: Theoretical and Methodological Issues. *Journal of the Learning Sciences* 13, 1 (2004), 15–42. https://doi.org/10.1207/s15327809jls1301_2 arXiv:https://doi.org/10.1207/s15327809jls1301_2
[9] Kathryn Cunningham, Sarah Blanchard, Barbara Ericson, and Mark Guzdial. 2017. Using Tracing and Sketching to Solve Programming Problems: Replicating and Extending an Analysis of What Students Draw. In *2017 ACM Conference on International Computing Education Research*. ACM, 164–172.
[10] Holger Danielsiek, Laura Toma, and Jan Vahrenhold. 2017. An Instrument to Assess Self-Efficacy in Introductory Algorithms Courses. In *2017 ACM Conference on International Computing Education Research*. ACM, 217–225.
[11] Ton de Jong. 2010. Cognitive load theory, educational research, and instructional design: Some food for thought. *Instructional Science* 38, 2 (2010), 105–134.

[12] Allison Elliott Tew. 2010. Assessing fundamental introductory computing concept knowledge in a language independent manner. December 2010 (2010), 147. http://search.proquest.com/docview/873212789?accountid=14696

[13] Barbara J Ericson, Kantwon Rogers, Miranda C Parker, Briana B Morrison, and Mark Guzdial. 2016. Identifying Design Principles for CS Teacher Ebooks through Design-Based Research.. In *ICER*. 191–200.

[14] Peter Gerjets, Katharina Scheiter, and Gabriele Cierniak. 2009. The scientific value of cognitive load theory: A research agenda based on the structuralist view of theories. *Educational Psychology Review* 21, 1 (2009), 43–54. https://doi.org/10.1007/s10648-008-9096-1 arXiv:arXiv:astro-ph/0507464v2

[15] Susannah Go and Brian Dorn. 2016. Thanks for Sharing: CS Pedagogical Content Knowledge Sharing in Online Environments. In *WiPSCE '16 (WiPSCE '16)*. ACM, New York, NY, USA, 27–36. https://doi.org/10.1145/2978249.2978253

[16] Shirley Gregor. 2006. The nature of theory in Information Systems. *MIS Quarterly: Management Information Systems* 30, 3 (2006), 611–642. https://doi.org/10.2307/25148742

[17] Jean M Griffin. 2016. Learning by taking apart: deconstructing code by reading, tracing, and debugging. In *Conference on Information Technology Education*. ACM, 148–153.

[18] Peter Hedström and Petri Ylikoski. 2010. Causal Mechanisms in the Social Sciences. *Annual Review of Sociology* 36, 1 (2010), 49–67. https://doi.org/10.1146/annurev.soc.012809.102632

[19] Maya Israel, Quentin M Wherfel, Saadeddine Shehab, Oliver Melvin, and Todd Lash. 2017. Describing Elementary Students' Interactions in K-5 Puzzle-based Computer Science Environments using the Collaborative Computing Observation Instrument (C-COI). In *2017 ACM Conference on International Computing Education Research*. ACM, 110–117.

[20] Betsy James DiSalvo, Sarita Yardi, Mark Guzdial, Tom McKlin, Charles Meadows, Kenneth Perry, and Amy Bruckman. 2011. African American men constructing computing identity. In *CHI '11*. ACM, 2967–2970.

[21] Ashutosh Jogalekar. 2014. Falsification and its discontents. *Scientific American* (2014). https://blogs.scientificamerican.com/the-curious-wavefunction/falsification-and-its-discontents/

[22] Yasmin B Kafai and Mitchel Resnick. 1996. *Constructionism in practice: Designing, thinking, and learning in a digital world*. Routledge.

[23] Andrew J Ko. 2009. Attitudes and self-efficacy in young adults' computing autobiographies. In *IEEE Symposium on Visual Languages and Human-Centric Computing*. IEEE, IEEE, New Jersey, 67–74.

[24] Andrew J Ko and Katie Davis. 2017. Computing Mentorship in a Software Boomtown: Relationships to Adolescent Interest and Beliefs. In *2017 ACM Conference on International Computing Education Research*. ACM, 236–244.

[25] Michael Kölling. 2010. The greenfoot programming environment. *ACM Transactions on Computing Education (TOCE)* 10, 4 (2010), 14.

[26] T.S. Kuhn. 1962. *The Structure of Scientific Revolutions*. University of Chicago Press. https://books.google.com/books?id=E6MJuAAACAAJ

[27] Michael J Lee and Andrew J Ko. 2012. Investigating the role of purposeful goals on novices' engagement in a programming game. In *Visual Languages and Human-Centric Computing (VL/HCC), 2012 IEEE Symposium on*. IEEE, 163–166.

[28] Michael J Lee, Andrew J Ko, and Irwin Kwan. 2013. In-game assessments increase novice programmers' engagement and level completion speed. In *ICER '13*. ACM, 153–160.

[29] Paul Luo Li, Andrew J Ko, and Jiamin Zhu. 2015. What makes a great software engineer?. In *37th International Conference on Software Engineering-Volume 1*. IEEE Press, 700–710.

[30] Alex Lishinski, Aman Yadav, and Richard Enbody. 2017. Students' Emotional Reactions to Programming Projects in Introduction to Programming: Measurement Approach and Influence on Learning Outcomes. In *2017 ACM Conference on International Computing Education Research*. ACM, 30–38.

[31] Dastyni Loksa, Andrew J Ko, Will Jernigan, Alannah Oleson, Christopher J Mendez, and Margaret M Burnett. 2016. Programming, Problem Solving, and Self-Awareness: Effects of Explicit Guidance. In *2016 CHI Conference on Human Factors in Computing Systems*. ACM, 1449–1461.

[32] Lauri Malmi, Ahmad Taherkhani, Judy Sheard, Roman Bednarik, Juha Helminen, Päivi Kinnunen, Ari Korhonen, Niko Myller, and Juha Sorva. 2014. Theoretical underpinnings of computing education research. *ICER '14* (2014), 27–34. https://doi.org/10.1145/2632320.2632358

[33] Jane Margolis and Allan Fisher. 2003. *Unlocking the clubhouse: Women in computing*. MIT press, Cambridge, MA.

[34] Lauren Margulieux and Richard Catrambone. 2017. Using Learners' Self-Explanations of Subgoals to Guide Initial Problem Solving in App Inventor. In *ICER '17*. ACM, 21–29.

[35] Lauren E. Margulieux and Richard Catrambone. 2016. Improving problem solving with subgoal labels in expository text and worked examples. *Learning and Instruction* 42, 2016 (apr 2016), 58–71.

[36] Lauren E. Margulieux, R. Catrambone, and M. Guzdial. 2013. Subgoal labeled worked examples improve K-12 teacher performance in computer programming training. *Proceedings of the 35th Annual Conference of the Cognitive Science Society* (2013), 978–983.

[37] Lauren E. Margulieux, B.B. Morrison, M. Guzdial, and R. Catrambone. 2016. Training Learners to Self-Explain: Designing Instructions and Examples to Improve Problem Solving. *Proceedings of International Conference of the Learning Sciences* 1 (2016), 98–105.

[38] M. Margulieux, L. E., Catrambone, R., & Guzdial. 2012. Subgoals improve performance in computer programming construction tasks. In *Proceedings of the EARLI SIG 6&7 Conference*. 60–62.

[39] Raina Mason, Graham Cooper, Barry Wilks, et al. 2016. Flipping the Assessment of Cognitive Load: Why and How. In *ICER '16*. ACM, 43–52.

[40] Mary J. Melrose. 2001. Maximizing the Rigor of Action Research: Why Would You Want To? How Could You? *Field Methods* 13, 2 (2001), 160–180. https://doi.org/10.1177/1525822X0101300203 arXiv:https://doi.org/10.1177/1525822X0101300203

[41] Roxana Moreno. 2010. Cognitive load theory: more food for thought. *Instructional Science* 38, 2 (2010), 135–141. https://doi.org/10.1007/s11251-009-9122-9

[42] Briana Morrison, Lauren Margulieux, and Mark Guzdial. 2015. Subgoals, Context, and Worked Examples in Learning Computing Problem Solving. In *Proceedings of the eleventh annual International Conference on International Computing Education Research - ICER '15*. ACM Press, 267–268. https://doi.org/10.1145/2787622.2787744

[43] Briana B. Morrison. 2017. Dual Modality Code Explanations for Novices. *ICER '17* (2017), 226–235. https://doi.org/10.1145/3105726.3106191

[44] Briana B. Morrison, Brian Dorn, and Mark Guzdial. 2014. Measuring cognitive load in introductory CS. *ICER '14* (2014), 131–138. https://doi.org/10.1145/2632320.2632348

[45] Greg L Nelson, Benjamin Xie, and Andrew J Ko. 2017. Comprehension First: Evaluating a Novel Pedagogy and Tutoring System for Program Tracing in CS1. *thirteenth International Workshop on Computing Education Research Workshop (ICER '17)* (2017), 42–51.

[46] Miranda C Parker and Mark Guzdial. 2016. Replication, validation, and use of a language independent CS1 knowledge assessment. *ICER* (2016), 93–101. https://doi.org/10.1145/2960310.2960316

[47] Hadi Partovi. 2015. A comprehensive effort to expand access and diversity in computer science. *ACM Inroads* 6, 3 (2015), 67–72.

[48] Melanie Peffer, Maggie Renken, and Debra Tomanek. 2016. Practical Strategies for Collaboration across Discipline-Based Education Research and the Learning Sciences. *CBEâĂŤLife Sciences Education* 15, 4 (2016), es11. https://doi.org/10.1187/cbe.15-12-0252 arXiv:https://doi.org/10.1187/cbe.15-12-0252 PMID: 27881446.

[49] Ken Peffers, Tuure Tuunanen, Marcus Rothenberger, and Samir Chatterjee. 2007. A Design Science Research Methodology for Information Systems Research. *J. Manage. Inf. Syst.* 24, 3 (Dec. 2007), 45–77. https://doi.org/10.2753/MIS0742-1222240302

[50] Karl R Popper. 1959. The logic of scientific discovery. (1959).

[51] Leo Porter, Dennis Bouvier, Quintin Cutts, Scott Grissom, Cynthia Lee, Robert McCartney, Daniel Zingaro, and Beth Simon. 2016. A multi-institutional study of peer instruction in introductory computing. *ACM Inroads* 7, 2 (2016), 76–81.

[52] Kathryn M. Rich, Carla Strickland, T. Andrew Binkowski, Cheryl Moran, and Diana Franklin. 2017. K-8 Learning Trajectories Derived from Research Literature: Sequence, Repetition, Conditionals. In *ICER '17*. ACM, 182–190. https://doi.org/10.1145/3105726.3106166

[53] Yvonne Rogers. 2012. HCI Theory: Classical, Modern, and Contemporary. *Synthesis Lectures on Human-Centered Informatics* 5, 2 (2012), 1–129. https://doi.org/10.2200/S00418ED1V01Y201205HCI014

[54] Juha Sorva. 2010. Reflections on threshold concepts in computer programming and beyond. *Koli Calling '10* (2010), 21–30. https://doi.org/10.1145/1930464.1930467

[55] Daniel Stokols. 2006. Toward a Science of Transdisciplinary Action Research. *American Journal of Community Psychology* 38, 1 (01 Sep 2006), 63–77. https://doi.org/10.1007/s10464-006-9060-5

[56] John Sweller. 2010. Element interactivity and intrinsic, extraneous, and germane cognitive load. *Educational Psychology Review* 22, 2 (2010), 123–138. https://doi.org/10.1007/s10648-010-9128-5 arXiv:arXiv:1002.2562v1

[57] John Sweller, Paul Ayres, and Slava Kalyuga. 2011. *Cognitive load theory*. Vol. 1. Springer.

[58] C. Taylor, D. Zingaro, L. Porter, K.C. Webb, C.B. Lee, and M. Clancy. 2014. Computer science concept inventories: past and future. *Computer Science Education* 24, 4 (2014), 253–276. https://doi.org/10.1080/08993408.2014.970779

[59] Allison Elliott Tew and Mark Guzdial. 2010. Developing a Validated Assessment of Fundamental CS1 Concepts. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education (SIGCSE '10)*. ACM, New York, NY, USA, 97–101. https://doi.org/10.1145/1734263.1734297

[60] Allison Elliott Tew and Mark Guzdial. 2011. The FCS1: a language independent assessment of CS1 knowledge. In *42nd ACM technical symposium on Computer science education*. ACM, 111–116.

[61] David W Valentine. 2004. CS educational research: a meta-analysis of SIGCSE technical symposium proceedings. *ACM SIGCSE Bulletin* 36, 1 (2004), 255–259. https://doi.org/10.1145/1028174.971391