Development and Validation of a Parallel and Distributed Computing Exam for Early Computing Students & Research*

1st Chris Bourke
School of Computing
University of Nebraska–Lincoln
Lincoln, NE USA
chris.bourke@unl.edu
0000-0002-1626-054X

2nd David P. Bunde
Computer Science
Knox College
Galesburg, IL USA
dbunde@knox.edu
0000-0001-6334-356X

3rd Charlotte Gruner
Computer Science
Casper College
Casper, WY USA
charlotte.gruner@caspercollege.edu
0009-0007-0522-8839

4th Mary Smith
Computer Science and Engineering
Hawai'i Pacific University
Honolulu, HI USA
mlsmith@hpu.edu
0000-0003-3912-0061

5th Xiaoyuan Suo
Computer and Information Sciences
Webster University
St. Louis, MO USA
xiaoyuansuo51@webster.edu
0000-0003-2473-0705

***Abstract*—**
Significant research has been dedicated to integrating parallel and distributed computing (PDC) concepts into early computing curricula. However, there has yet to be a validated psychometric tool to objectively assess students' understanding of PDC topics. Validated instruments are essential for ensuring the reliability and validity of research findings, enabling rigorous evaluations of educational interventions. In this work-in-progress, we present the development of such an assessment. Additionally, we outline our strategies for validating this exam, with the goal of using it as a research instrument for evaluating interventions in introductory computing courses. We hope that sharing our initial efforts will spark interest within the PDC education research community and invite constructive feedback.

***Index Terms*—Early Computing Curriculum, CS1, Validated Instrument, Parallel Computing, Distributed Computing**

I. Introduction

Multicore architecture has been the de facto hardware paradigm for the last twenty years. However, few computing curricula have been updated to reflect this reality. The past decade researchers have advocated that students should be exposed to Parallel and Distributed Computing (PDC) as early as possible so that they consider it a natural computing paradigm rather than an advanced or rarely used concept [1]. In response, numerous research studies have investigated how to effectively introduce PDC topics in early computing curriculum [2].

Though many exercises, lessons, and other interventions have been reported in the literature, much of this research has used surveys, student self-reported assessments, or other ad-hoc measures to quantify their efficacy. In fact, the vast majority of research papers have not even quantified learning gains at all. In a systematic review of literature 2008–2019 85% of papers were essentially experience reports which included instructor perspectives but little to no theoretical educational research results [2].

While well-designed and *validated* surveys can reliably capture students' *perceptions* and related constructs such as self-efficacy, engagement, satisfaction, etc. these are generally *affective* outcomes that deal with the feelings, values, attitudes, motivations, and dispositions that students develop as a result of their learning experiences. Often these instruments ask for subjective matters (ex. "Did you *feel* you learned from this module" or "Do you have high confidence in your ability after this module"). Surveys are common because they are easy to administer, efficient, scalable and minimally intrusive. While surveys are useful for providing context, they do not directly measure actual learning gains or intervention efficacy.

Indeed, surveys primarily measure *perceived* learning rather than actual knowledge or skill acquisition. Multiple studies find that self-reported learning gains often do not correlate strongly with actual knowledge or performance gains measured by objective tests, especially for specific content knowledge [3], [4], [5]. In addition, surveys are subject to biases such as overconfidence, response styles, inattentive responding, and memory errors, which can distort results [6]. Perceptions of learning may reflect confidence or satisfaction rather than true learning; students might feel an intervention helped, but test scores or observations might show otherwise [7], [8], [9]. Survey data should be interpreted cautiously and ideally triangulated with direct measures. The current state-of-the practice of education research is to use mixed

Supported by NSF 2321015

methods which combine surveys with a more accurate and comprehensive evaluation of learning gains.

High-quality rigorous education research relies on *validated* instruments: measurement tools that have been shown through systematic empirical evidence to accurately and reliably measure what it purports to measure. "Validated assessments are important for researchers to reduce experimental error due to flawed assessments and to allow for comparisons between different experiments" [10]. Decker & McGill [11] observe the need for formal evaluations using standardized and validated instruments while noting that researchers often fail to do so, instead creating their own or relying on ad-hoc measures. They systematically identified 47 such validated instruments in the context of computing, none of which include PDC topics.

While there are validated instruments for computing education broadly [12], [13], [14], [15], [16], [17], [10], there is a notable gap in validated tools for PDC-specific learning outcomes. No widely recognized, validated assessment instruments exist that are designed specifically for PDC interventions in computer science education. This current work seeks to fill this gap.

In this work-in-progress report, we detail the development of a multiple choice exam instrument for measuring knowledge of PDC concepts at the early computing level (CS0/1/2). We plan to administer this exam across multiple early computing courses at multiple institutions and to validate the exam using Item Response Theory (IRT) a widely used methodology in educational testing and psychology. IRT models the probability that a person with a certain ability or trait will answer a specific item correctly. It characterizes each question (item) by parameters such as difficulty and discrimination (how well it differentiates between students of different ability levels). Our hope is that by reporting on our progress so far, we will spark community interest in this project, solicit feedback on our exam content and validation methodology, and eventually effect widespread adoption and adaptation of our instrument. PDC research approaches that combine our instrument with surveys may result in more rigorous and higher quality studies, "triangulating" findings of student attitudes and understanding in a broader context.

In the next section we discuss related work. In section III we discuss the development of the exam and in section IV we discuss the final content we developed and its relation to well-established PDC curricula. In section V we detail our exam administration and validation plans using IRT.

II. Related Work

As parallel and distributed computing (PDC) becomes central to modern computing practice, undergraduate programs have increasingly introduced PDC topics in early courses, such as CS0, CS1, and CS2. Yet no psychometrically validated instrument exists to measure students' conceptual understanding of PDC. This absence limits rigorous evaluation of instructional interventions, including unplugged activities, visualization tools, and introductory parallel-programming modules, and hinders researchers' ability to identify misconceptions related to synchronization, communication, nondeterminism, or task decomposition. Developing a validated PDC assessment requires drawing from established STEM assessment methods, validated computing education instruments, concept inventory (CI) design methodologies, and current work on PDC learning objectives and conceptual challenges.

A. Validated Assessments in STEM

Validated assessments are well established in STEM fields and provide methodological grounding for the development of psychometrically sound tests. General science assessments [18], calculus-based physics exams [19], and mathematics conceptual assessments [20] demonstrate how standardized items can reliably measure domain-specific understanding and reveal misconceptions. These efforts illustrate how stable, validated instruments support evidence-based instruction and large-scale educational research.

Computing education likewise includes multiple validated or standardized tools. The AP CS A exam [12] undergoes continuous revalidation, while additional instruments measure aptitude [13], computing self-efficacy [14], and computational thinking skills [15][16]. Knowledge-focused assessments include the Foundational CS1 (FCS1) [17] and Second CS1 Assessment (SCS1) [10]. A recent review identifies 47 validated instruments across computing education [11], but none address PDC. These instruments show that rigorous assessment design is feasible in computing, but remains unexplored for PDC [21].

Relatedly, concept inventories (CI) aim to uncover persistent misconceptions by using carefully designed distractors grounded in empirical data and expert validation. Foundational examples include the Force Concept Inventory [22] and the Conceptual Survey of Electricity and Magnetism [23]. In computing, validated or systematically developed CIs cover topics such as digital logic [24], discrete structures and programming fundamentals [25], and dynamic programming [26]. A systematic review synthesizes CI development across computing [27]. Psychometric frameworks, Classical Test Theory (CTT) [28] and Item Response Theory (IRT)[29], support these instruments by quantifying item performance, modeling latent ability, and enabling iterative refinement. IRT, used in large-scale assessments nationally (NAEP) and internationally (PISA), provides additional analytic rigor that is essential for validated CI design [30].

B. PDC Instruction and Existing Assessments

The undergraduate PDC curricula emphasize coordination, communication, synchronization, decomposition, and performance evaluation [31], with proposals to integrate PDC throughout the major [32]. Although instructional materials and activities have expanded, PDC assessment remains limited. Existing work includes survey-based measures—for example, studies using ASPECT [33] to evaluate engagement in unplugged PDC activities [34] and a range of ad-hoc pre/post tests [35], passing-rate analyses [36], and short sets

of exam questions [37]. More recent efforts employ open-ended prompts but lack formal psychometric validation [38]. No existing assessment systematically targets core PDC misconceptions using validated CI methodology.

C. The Assessment Gap

Across STEM and computing education, validated assessments and concept inventories have proven essential for measuring conceptual understanding, diagnosing misconceptions, and evaluating curricular impact. Despite strong curricular mandates and well-documented conceptual challenges in PDC, no validated assessment exists for early undergraduate PDC learning. Clearly, there is a need for a psychometrically rigorous PDC concept inventory grounded validated methodology, aligned with ACM learning objectives, and capable of supporting research and instructional improvement.

III. Development

A. Learning Outcomes & Design Goals

The overall goal of designing and implementing this PDC exam is to create an instrument that measures an introductory computing student's basic knowledge of PDC topics. Our intention is that researchers adopt this instrument as a pre/post test instrument to measure the efficacy of PDC interventions in early computing courses. We have made the instrument general enough, avoiding specific programming languages, frameworks, etc., that it can be used in a variety of interventions. This instrument is intended as a starting point for researchers who may adapt and customize it to their specific intervention. If the intervention does not include asynchronous computing for example, questions tagged with that topic may be omitted.

We designed the exam to be administered to lower-level introductory computing courses (CS0, CS1, CS2). It focuses on introductory topics in a more abstract manner. For example, a student is expected to differentiate between sequential and parallel computing, recognize the speedup benefits, and understand the limitations. However, we are not expecting a student to be able to write a `pthread` to solve a specific problem.

1) Learning Outcomes: This instrument is designed to measure the following learning outcomes. A student would be a considered high-ability student with respect to this exam if they are able to:

1) understand what Parallel Computing (PC) is
2) define basic terminology (threads, cores, etc.)
3) recognize the benefits (speed up) of PC
4) identify scenarios (data parallel, embarrassingly parallel) that may benefit from PC
5) understand the limitations of PC
6) understand what Distributed Computing (DC) is
7) understand what Asynchronous Computing (AC) is
8) distinguish between sequential, parallel, distributed, and asynchronous computing

The learning outcomes generally fall on the lower end of Bloom's taxonomy [39]. Specifically, 3 outcomes align with the lowest level (Remember) while 5 align with the 2nd level (Understand). This is appropriate as an introduction to a topic aims to build basic familiarity with terminology and concepts, congruent with the lower two tiers. Even if an intervention aims to cover higher tiers of Bloom's, this instrument can still provide a baseline for measurement as (generally) the intervention should implicitly cover these lower tiers.

To keep the exam general, we avoided lower-level details and topics and terminology (Flynn's taxonomy, race conditions, starvation, etc.) and made it language and framework agnostic.

B. Development Process

To develop the exam, a team of experienced computing education researchers who all had expertise in developing and delivering PDC content in early computing courses contributed to a question bank initially consisting of 76 multiple choice questions. We opted for a multiple choice format for ease of delivery, scalability and assessment and so we could utilize well-established validation tools and techniques.

Following the initial development, questions and answers were randomized and provided to each author who took the exam as a student would (without answers available). Questions that were answered incorrectly were particularly flagged as being potentially misleading or not well-formed. Each author also provided additional feedback on each question: a) they provided a likert-scale rank on the quality and appropriateness (1 = low quality/least appropriate, 5 = high quality, very appropriate) and b) additional written feedback to provide context for the rating as well as identify any other issues such as typos, poor phrasing, etc. and improvements.

After this initial evaluation, each question was reevaluated, taking the reviews into account and generating summary statistics (mean, median, low/high/range) on the quality/appropriateness. Any question that had a low rating, lacked a consensus (high range) or was too low-level (Flynn's taxonomy, mutexes, written for a specific language or framework) was removed. Other questions that had high ratings and consensus were marked as "retain" while those with high-to-medium ratings but less consensus were marked as "maybe". In the end, we retained the highest rated $I = 31$ questions ("maybe" questions were retained for future revisions).

The final initial version was then formatted into JSON for easy processing and randomization (of questions and responses where appropriate). Scripts were developed for translation to other formats (including common LMS delivery systems).

IV. Content

A. Questions & Topic Coverage

Our initial design consists of $I = 31$ total multiple choice questions. All questions (save one) have 4 possible answers with only a single correct answer. One question is a NOTA (None of the Above) with 6 possible answers. Three questions have been tagged as "ordered" (the responses should be presented in a particular order) while all others may have their responses randomized. The questions themselves are

TABLE I
PDC EXAM TOPICS & SUBTOPICS. SOME QUESTIONS MAY BE TAGGED WITH MULTIPLE TOPICS/SUBTOPICS SO THE TOTAL DOES NOT MATCH THE TOTAL NUMBER OF QUESTIONS.

Topic/SubTopic	Count
Parallel Computing	15
Performance	3
Performance Limits	1
Distributed Computing	9
Asynchronous Computing	4
Event Handling	1
Concurrent Computing	1
Synchronization	1
Sequential Computing	2
PDC Vocabulary	3

Which of the following best illustrates **Parallel Computing**

a. Task A runs on computer X while Task B runs on computer Y, once both have completed, they report their results back to computer Z
b. Task B runs after Task A has been completed
c. Task A runs for a while, then pauses and allows Task B to execute, then resumes once Task B has completed
d. Task A runs on Processor X at the same time as Task B runs on Processor Y

Fig. 1. A sample question on parallel computing.

also designed to be independent so they can be (optionally) randomly ordered in an actual exam.

Each question is tagged with a unique identifier and (potentially multiple) topics/subtopics. The list of topics as well as the number of questions for each can be found in Table I. The ID and topic tags are for analysis and not intended to be displayed to the examinee. An example question can be found in Figure 1.

The full repository of questions is available in several formats (JSON, markdown, exported CMS modules including Moodle and Canvas) at https://github.com/cbourke/ValidatedPDCExam-public

B. Curricular Alignment

In the next two sections, we analyze our exam with respect to two well known undergraduate curriculum standards for PDC.

1) PDC2020: The NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing (PDC2020) is a standard and widely used undergraduate curriculum guideline for PDC. We analyzed our exam using the most recent version (II-beta) [40]. PDC2020 defines four broad "pervasive" topics that cut across all levels of the curriculum and interrelate to each other. We mapped our learning outcomes (see section III-A1) to these four topics (Table II). With the exception of locality, all pervasive topics are well-covered by multiple learning outcomes. Locality is a much lower-level topic that requires system and architecture knowledge that is not typically covered in introductory computing courses.

The PDC2020 also identifies four topic areas; Architecture, Programming, Algorithms, and Emerging topics. Each topic area has dozens of topics and subtopics and maps them to (multiple) courses where it is recommended they be covered to an appropriate degree.

TABLE II
PDC2020 PERVASIVE TOPICS MAPPED TO OUR LEARNING OUTCOMES.

Pervasive Topic	Mapped Learning Outcomes
Concurrency	1, 2, 8
Asynchrony	2, 6, 7, 8
Locality	NA
Performance	2, 3, 4, 5

Our analysis initially restricted attention to areas and topics that were specifically identified with CS1. No Emerging topics were identified with CS1 and so it is omitted from our analysis. In total, 8 topics were associated with CS1. The guidelines do not identify any topics *exclusive* to CS1, though they do identify 19 other topics associated with CS2. We expanded our analysis and identified 3 of these topics as relevant to our exam coverage and included them since the guidelines recommended a Bloom level (know, comprehend, understand) that aligns with our exam design. For each topic, we mapped our learning outcomes and specific questions. The results can be found in Table III.

TABLE III
PDC2020 TOPIC/SUBTOPIC QUESTION MAPPING. EACH AREA/TOPIC IS IDENTIFIED AS A CS1/CS2 OR CS2 TOPICS; TOPICS EXCLUSIVE TO CS2 ARE DENOTED WITH †. OUR LEARNING OUTCOMES (LOS) AND THE NUMBER OF QUESTIONS (#QS) ASSOCIATED WITH EACH ONE ARE INCLUDED.

Area	Topic	LOs	#Qs
Architecture	Floating Point - Range	NA	-
	Floating Point - Precision	NA	-
	Floating Point - IEEE754	NA	-
	Event Handling†	7	1
Programming	Concurrency and Parallelism	1, 8	13
	Event Driven Execution†	6	10
	Performance metrics†	3, 4, 5	6
Algorithm	Concurrency, Asynchrony, Dependencies, Nondeterminism	7	1
	Dependencies	1,2	2
	Synchronization	1,2	2
	Algorithms for streams	NA	-

All of our learning outcomes are aligned with one or more PDC2020 topics. Conversely, most PDC2020 topics are covered strongly by multiple questions with the exception of 4 CS1 topics. Three of them involve floating point representations which are ancillary to PDC and not a focus of the exam. While a CS1 course would certainly discuss limitations (precision, overflow, etc.) it would not be common to do so in the context of PDC. The fourth uncovered CS1 topic involves algorithms and data structures (bloom filters, distributed hash tables) for streams which is out-of-scope for our exam and not commonly covered in CS1. The three CS2 topics we identified focus generally on parallel, distributed, and asynchronous computing, the primary content of our exam.

TABLE IV
MAPPING OF CC2023 KNOWLEDGE UNITS TO LEARNING OUTCOMES AND QUESTIONS.

Knowledge Unit	LOs	#Qs
PDC-Programs	1, 2, 5, 6, 7, 8	22
PDC-Evaluation	3, 4	7
PDC-Communication	2	2
PDC-Coordination	2	1
PDC-Algorithms	NA	-

We note that there are 16 other CS2-level topics that are not covered by our exam. These include specialized topics (SPMD, SIMD, MapReduce, etc.) or are recommended for a higher bloom level (apply) that is beyond the scope of our exam. For example, our exam would expect students to be able to identify a thread, have a good understanding of parallel work, its benefits and limitations, but they would not be expected to actually apply (i.e. write) multithreaded code (at least not without substantial starter code).

This analysis demonstrates that our exam is well aligned with the PDC2020 standard recommendations in so far that topic coverage and depth is appropriate to the level of our exam design.

2) ACM/IEEE: ACM and IEEE-CS also sponsor joint task forces to publish curricular recommendations for computing, in particular, the Parallel and Distributed Computing (PDC) Knowledge Area (CC2023) [31]. These recommendations divide topics in the discipline into overlapping knowledge areas, each containing a number of knowledge units, which themselves specify topics to be covered and illustrative learning outcomes related to them. These knowledge areas are explicitly allowed to overlap. We examined our learning objectives and questions in relation to the PDC knowledge area, which shares many topics with other areas, including Algorithmic Foundations (AL), Architecture and Organization (AR), Foundations of Programming Languages (FPL), Operating Systems (OS), and Systems Fundamentals (SF), but we present the knowledge units as they appear in the PDC area.

Table IV shows the knowledge units in PDC to which our learning outcomes and questions map. Clearly the strongest overlap for both is with PDC-Programs, which is the most fundamental knowledge unit of the area and includes the definitions of the terms used to classify PDC systems (tasks, dependencies/independence, the "places" of a distributed system, and the way that tasks are scheduled on them. Thus, it covers all the learning outcomes about the types of computing. The next highest level of overlap is with PDC-Evaluation, which is concerned with parallel performance and overlaps with our questions about speedup and the goals of parallel computing.

We found much less overlap with the other units. PDC-Communication is about communication channels (message passing or shared memory) and APIs, with suggested learning outcomes like explaining the difference between sending a message and writing to shared memory or writing a program using a specific kind of API. PDC-Coodination is about using control constructs and design patterns to avoid races, ensure termination, and similar requirements. A sample learning outcome is to fix a race condition in a given program (e.g. by adding a lock). PDC-Algorithms is about expressing and implementing PDC algorithms, including the use of language constructs and APIs.

We do not take our lack of coverage of topics in these three units as an issue with our exam. While CC2023 does not identify levels at which to cover specific topics or recommend Bloom levels at which to cover them, the examples indicate that the topics are more advanced than the introductory courses we are targeting. In addition, many of the topics in these units are also tied to implementations, making them relatively inaccessible for a language independent instrument such as ours.

V. VALIDATION PLAN

A. Data Collection

During 2026 we plan on administering our exam in a variety of introductory computing courses at multiple institutions. Our goal is to collect 100-200 responses.

Given the diversity of courses and institutions, some examinees may have had prior PDC experience. We will also collect data indicating the probable level of prior PDC knowledge (from the instructor and/or the student). This will allow us to perform a stratified analysis. For those without prior experience, our analysis will act as a *pretest* and our expectations would be that this population is of *low ability*. For those with some or significant prior experience with PDC, the population will be assumed to be *high ability*. We will also perform a combined analysis.

B. Validation & Item Response Theory

We will use Item Response Theory to to determine each question's *difficulty* and *fit*. Difficulty is determined by how many students correctly answer the question. The fewer students who successfully answer the question the more difficult it is. Fit is a measure of how well the question distinguishes between a high-ability student and a low-ability student. Ideally, a good test should have multiple questions at different scales of difficulty, but all questions should have good fit. This gives evidence that the exam is a *reliable* and *valid* instrument that measures examinees' performance across a large range of skills.

To measure difficulty, we'll use conditional maximum likelihood (CML) estimation, a measure on the scale [-3, 3] ranging from easy (-3) to average (0) to difficult (3). [41]. We'll graph each question using an item characteristic curve (ICC). ICC is a visualization of each questions's difficulty. The x-axis represents a *latent trait* (θ). In our case, it represents a student's basic PDC knowledge. The y-axis represents the probability of a correct response ([0,1]). A mock example can be found in Figure 2.

We plan to (initially) use a 3-parameter logistic model (3PL) which incorporates three parameters: *difficulty* (b), *discrimination* (a) and *guessing* (c). The characteristics of the curve

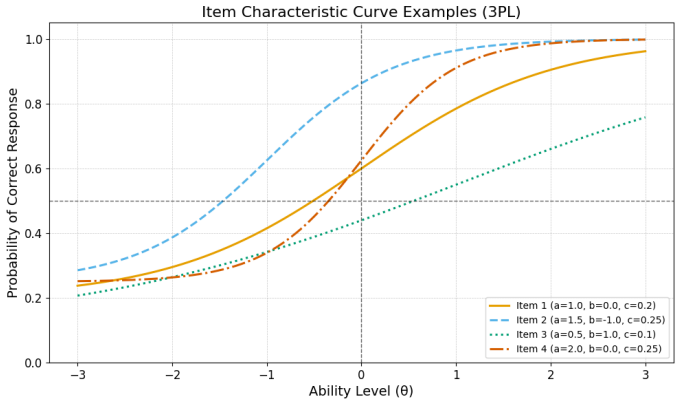

Fig. 2. Item Characteristic Curve Examples

indicates each question's parameter. For each question, the shape of the curve is generally a sigmoid curve centered at $x = b$, the difficulty level. This represents the ability level at which the $y = .5$ (examinees have a 50% probability of answering correctly). The steepness of the curve indicates its discrimination. Generally, the steeper the curve, the better differentiation between individuals near that difficulty level. Conversely, a shallow curve would indicate that the question is less reliable in differentiating ability (and thus is less valid).

Guessing is a third parameter that incorporates the probability that the examinee correctly answers the question purely by guessing. This affects where the curve starts. For example, if a question has 4 possible choices, a low-ability student who guesses has a theoretical lower bound of $c = .25 = 25\%$ probability of guessing correctly and the curve would start at about $y = .25$. Empirically, if $c < .25$ it suggests that the question distractors may be strong (there is a lower than expected probability of guessing) while $c > .25$ may indicate that distractors are weak (there are obviously wrong or irrelevant answer choices) or certain choices are "clues" (obviously correct answers).

In Figure 2 in which we've graphed several different items with a variety of characteristics.

- Item 1's curve represents an item that has medium difficulty (centered at $b = 0$) and moderate discrimination ($a = 1$) and higher guessing (steeper initial curve)
- Item 2's curve represents an easier item ($b = -1$) with high discrimination $a = 1.5$
- Item 3's curve represents a harder ($b = 1$) item but with low discrimination ($a = 0.5$) and lower guessing (flatter initial curve)
- Item 4 represents a near-ideal item: it has moderate difficulty ($b = 0$) with very high discrimination ($a = 2.0$) and low guessing

The goal of a reliable and valid exam is to have multiple questions at each difficulty level (easy $b \in [-2, -1]$, medium $b \in [-0.5, 0.5]$, and hard $b \in [1, 2]$ questions). All questions should have good discrimination (steep curves with gradients in [.08, 2.5]). This makes the item more informative about the ability of an examinee at that difficulty level. Overly steep curves may indicate overfitting.

Generally, guessing (c) should be as low as possible. Since this is a multiple choice exam and not all students are expected to be proficient, guessing is expected and represents the probability of guessing the correct answer purely by chance. This is a function of the number of available choices. We designed most questions to have 4 answers and all of them only have one correct answer. Thus, generally, the theoretical lower bound for most of our questions will be $\approx .25$.

In addition to the graphical ICC analysis, we will also apply an Orlando-Thissen S-χ^2 ("S-chi-squared") statistic [42] to determine the *fit* of each item. The fit of an item is how well the observed responses match the predicted responses by the IRT model. Essentially, a good fit means that examinees with a higher ability are more likely to answer the item correctly and those with lower ability have a lower probability of answering the item correctly. S-χ^2 in particular is good for our dichotomous test design (each question only has a right/wrong result), works well with 3PL (and 2PL, see below) and is applicable for the smaller sample size that we anticipate.

C. Addressing Assumptions & Limitations

IRT makes several underlying assumptions that need to be met for the model to be valid. We detail how we'll address each assumption and identify further limitations and threats to validity.

Unidimensionality - IRT assumes that only a single latent trait is being measured by the instrument. In our exam design, we have limited questions only to PDC topics which makes it more niche than (say) an exam broadly covering all CS1 topics. Though there are several subtopics (parallel, distributed), they are all related and one of the exam's goals is to measure students' ability to differentiate between them.

Local Independence - an examinee's responses to different items should be statistically independent from each other and only dependent on their ability θ. If this assumption is violated may indicate redundant items. We will test for this assumption using *Yen's Q3 Statistic*, a commonly used method for tests of our type [43] that measures the correlation of item residuals (differences between observed and expected responses given θ). We will follow up by eliminating, replacing, or redrafting any questions that fail this validation.

Monotonicity - Each ICC should be monotone non-decreasing. If this is violated, it suggests that higher ability examinees have a lower probability of success. This will be verified through visual inspection of the IC curves. Items violating this assumption will also be removed, replaced or redrafted.

Sample Size - Generally, validation is more reliable when the sample size of data and number of items are large. However, there are practical limitations to both of these especially as an initial pilot study. We have designed the exam to be of reasonable length ($I = 31$ questions) with the goal of preserving at least $I \approx 25$ questions after validation (eliminating redundant or questions with poor fit) which is

generally sufficient for ability estimation [44]. IRT guidelines recommend respondent samples size of $N = 500 - 1,000$. However, since our target audience is somewhat niche, our goal is to have $N \approx 200$ respondents which is sufficient for an initial pilot study [45]. However, with a smaller sample size, the guessing parameter (c) may be unstable. We will be cautious with a full 3PL analysis and perform a 2PL analysis as well. This involves removing the guessing parameter and making a uniform assumption based on the number of possible answers.

VI. Conclusion

We have outlined a work-in-progress for developing and validating an exam testing PDC knowledge of early computing students. Upon validation, we will refine and update the exam. We share our early progress with the community in hopes that we can receive early feedback on the content. We anticipate that updating and refining the exam will be a longterm project with additional large scale data collection as its adoption in the PDC education research community grows.

Acknowledgements

This work was supported by NSF 2321015. We thank everyone at the Center for Parallel and Distributed Computing Curriculum Development and Educational Resources for their assistance.

References

[1] Y. Ko, B. Burgstaller, and B. Scholz, "Parallel from the beginning: the case for multicore programming in the computer science undergraduate curriculum," in *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '13. New York, NY, USA: Association for Computing Machinery, 2013, p. 415–420. [Online]. Available: https://doi.org/10.1145/2445196.2445320

[2] Sitsylitsyn, Yuriy, "Methods and tools for teaching parallel and distributed computing in universities: a systematic review of the literature," *SHS Web Conf.*, vol. 75, p. 04017, 2020. [Online]. Available: https://doi.org/10.1051/shsconf/20207504017

[3] S. P. León, E. Panadero, and I. García-Martínez, "How accurate are our students? a meta-analytic systematic review on self-assessment scoring accuracy," *Educational Psychology Review*, vol. 35, no. 106, 2023.

[4] S. R. Porter, "Self-reported learning gains: A theory and test of college student survey response," *Research in Higher Education*, vol. 54, pp. 201–226, 2013.

[5] F. S. Hadi, J. Arlinwibowo, and G. N. Fatima, "A meta-analysis of the relationship between self-assessment and mathematics learning achievement," *Jurnal Penelitian dan Evaluasi Pendidikan*, vol. 27, no. 1, p. 39–51, Jun. 2023. [Online]. Available: https://journal.uny.ac.id/index.php/jpep/article/view/60617

[6] D. Tempelaar, B. Rienties, and Q. Nguyen, "Subjective data, objective data and the role of bias in predictive modelling: Lessons from a dispositional learning analytics application," *PLoS ONE*, vol. 15, 2020.

[7] K. Craig, D. Hale, C. Grainger, and M. Stewart, "Evaluating metacognitive self-reports: Systematic reviews of the value of self-report in metacognitive research," *Metacognition and Learning*, vol. 15, no. 2, pp. 155–213, 2020. [Online]. Available: https://doi.org/10.1007/s11409-020-09222-y

[8] K. S. Double, "Survey measures of metacognitive monitoring are often false," *Behavior Research Methods*, vol. 57, 2025. [Online]. Available: https://doi.org/10.3758/s13428-025-02621-6

[9] J. Li and Y. Copur-Gencturk, "Perceptions versus performance: Assessing teacher learning in asynchronous online professional development," *Education and Information Technologies*, vol. 30, pp. 4751–4776, 2025.

[10] M. C. Parker, M. Guzdial, and S. Engleman, "Replication, validation, and use of a language independent CS1 knowledge assessment," in *Proceedings of the 2016 ACM Conference on International Computing Education Research*, ser. ICER '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 93–101. [Online]. Available: https://doi.org/10.1145/2960310.2960316

[11] A. Decker and M. M. McGill, "A topical review of evaluation instruments for computing education," in *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 558–564. [Online]. Available: https://doi.org/10.1145/3287324.3287393

[12] C. Board, "AP computer science A," https://apcentral.collegeboard.org/courses/ap-computer-science-a, 1984, accessed 2025-10-31.

[13] A. Elliott Tew, B. Dorn, and O. Schneider, "Toward a validated computing attitudes survey," in *Proceedings of the Ninth Annual International Conference on International Computing Education Research*, ser. ICER '12. New York, NY, USA: Association for Computing Machinery, 2012, pp. 135–142. [Online]. Available: https://doi.org/10.1145/2361276.2361303

[14] V. Ramalingam and S. Wiedenbeck, "Development and validation of scores on a computer programming self-efficacy scale and group analyses of novice programmer self-efficacy," *Journal of Educational Computing Research*, vol. 19, no. 4, pp. 367–381, 1998. [Online]. Available: https://doi.org/10.2190/C670-Y3C8-LTJ1-CT3P

[15] M. S. Peteranetz, P. M. Morrow, and L.-K. Soh, "Development and validation of the computational thinking concepts and skills test," in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 926–932. [Online]. Available: https://doi.org/10.1145/3328778.3366813

[16] K. H. Koh, A. Basawapatna, H. Nickerson, and A. Repenning, "Real time assessment of computational thinking," in *2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 2014, pp. 49–52.

[17] A. E. Tew and M. Guzdial, "The FCS1: a language independent assessment of CS1 knowledge," in *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '11. New York, NY, USA: Association for Computing Machinery, 2011, pp. 111–116. [Online]. Available: https://doi.org/10.1145/1953163.1953200

[18] K. B. Follette, D. W. McCarthy, E. Dokter, S. Buxner, and E. E. Prather, "The quantitative reasoning for college science (quarcs) assessment, 1: Development and validation," *Numeracy*, vol. 8, no. 2, p. Article 2, 2015. [Online]. Available: https://doi.org/10.5038/1936-4660.8.2.2

[19] T. I. Smith, P. Eaton, S. W. Brahmia, A. Olsho, C. Zimmerman, and A. Boudreaux, "Toward a valid instrument for measuring physics quantitative literacy," in *Physics Education Research Conference 2020*, ser. PER Conference, Virtual Conference, July 22-23 2020, pp. 490–496.

[20] R. M. Suinn and E. H. Winston, "The mathematics anxiety rating scale, a brief version: Psychometric data," *Psychological Reports*, vol. 92, no. 1, pp. 167–173, 2003. [Online]. Available: https://doi.org/10.2466/pr0.2003.92.1.167

[21] K. Huff and B. Plake, "Evidence-centered assessment design in practice," pp. 307–309, 2010.

[22] D. Hestenes and I. Halloun, "Interpreting the force concept inventory," *The Physics Teacher*, vol. 33, no. 8, pp. 504–506, 1995.

[23] D. P. Maloney, T. L. O'Kuma, C. J. Hieggelke, and A. Van Heuvelen, "Surveying students' conceptual knowledge of electricity and magnetism," *American Journal of Physics*, vol. 69, no. S1, pp. S12–S23, 2001.

[24] G. L. Herman, C. Zilles, and M. C. Loui, "A psychometric evaluation of the digital logic concept inventory," *Computer Science Education*, vol. 24, no. 4, pp. 277–303, 2014.

[25] B. Chen, S. Azad, R. Haldar, M. West, and C. Zilles, "A validated scoring rubric for explain-in-plain-english questions," in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 2020, pp. 563–569.

[26] M. Ferland, V. Nagaraj Rao, A. Arora, D. van der Poel, M. Luu, R. Huynh, F. Reiber, S. Ossman, S. Poulsen, and M. Shindler, "Construction and preliminary validation of a dynamic programming concept inventory," in *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1*, 2025, pp. 325–331.

[27] M. Ali, S. Ghosh, P. Rao, R. Dhegaskar, S. Jawort, A. Medler, M. Shi, and S. Dasgupta, "Taking stock of concept inventories in computing education: A systematic literature review," in *Proceedings of the 2023 ACM Conference on International Computing Education Research - ICER '23*. New York, NY, USA: Association for Computing Machinery, 2023, pp. 397–415. [Online]. Available: https://doi.org/10.1145/3568813.3600120

[28] S. Alagumalai and D. D. Curtis, "Classical test theory," in *Applied Rasch measurement: A book of exemplars: Papers in honour of John P. Keeves*. Springer, 2005, pp. 1–14.

[29] L. Cai, K. Choi, M. Hansen, and L. Harrell, "Item response theory," *Annual Review of Statistics and Its Application*, vol. 3, no. 1, pp. 297–321, 2016.

[30] P. W. van Rijn, S. Sinharay, S. J. Haberman, and M. S. Johnson, "Assessment of fit of item response theory models used in large-scale educational survey assessments," *Large-Scale Assessments in Education*, vol. 4, no. 1, p. 10, 2016.

[31] D. Lea, S. G. Aly, M. Oudshoorn, Q. Xiang, D. Grossman, V. Sarkar, M. Herlihy, S. Ghafoor, C. Weems, and S. Burckhardt, "Parallel and distributed computing (pdc) knowledge area – version gamma," https://csed.acm.org/parallel-and-distributed-computing/, August 2023, version Gamma, August 2023.

[32] T. Newhall, A. Danner, and K. C. Webb, "Pervasive parallel and distributed computing in a liberal arts college curriculum," *Journal of Parallel and Distributed Computing*, vol. 105, pp. 53–62, 2017, keeping up with Technology: Teaching Parallel, Distributed and High-Performance Computing. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0743731517300114

[33] B. L. Wiggins, S. L. Eddy, L. Wener-Fligner, K. Freisem, D. Z. Grunspan, E. J. Theobald, J. Timbrook, and A. J. Crowe, "Aspect: A survey to assess student perspective of engagement in an active-learning classroom," *CBE—Life Sciences Education*, vol. 16, no. 2, p. ar32, 2017.

[34] M. Smith and S. Srivastava, "Evaluating student engagement towards integrating parallel and distributed computing (pdc) topics in undergraduate level computer science curriculum," in *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 1269. [Online]. Available: https://doi.org/10.1145/3287324.3293854

[35] S. A. Bogaerts, "One step at a time: Parallelism in an introductory programming course," *Journal of Parallel and Distributed Computing*, vol. 105, pp. 4–17, 2017, keeping up with Technology: Teaching Parallel, Distributed and High-Performance Computing. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0743731517300023

[36] H. Lin, "Teaching parallel and distributed computing using a cluster computing portal," in *2013 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum*, 2013, pp. 1312–1317.

[37] S. V. Moore and S. R. Dunlop, "A flipped classroom approach to teaching concurrency and parallelism," *2016 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pp. 987–995, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:14143467

[38] C. Bourke, "Codeless modules for parallel and distributed computing in early computing curriculum," in *Proceedings of the 57th ACM Technical Symposium on Computer Science Education*, ser. SIGCSE '26. New York, NY, USA: Association for Computing Machinery, 2026, p. To Appear. [Online]. Available: https://doi.org/10.1145/3770762.3772500

[39] L. W. Anderson and D. R. Krathwohl, *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. New York, NY: Longman, 2001.

[40] S. K. Prasad, T. Estrada, S. Ghafoor, A. Gupta, K. Kant, C. Stunkel, A. Sussman, R. Vaidyanathan, C. Weems, K. Agrawal, M. Barnas, D. W. Brown, R. Bryant, D. P. Bunde, C. Busch, D. Deb, E. Freudenthal, J. Jaja, M. Parashar, C. Phillips, B. Robey, A. Rosenberg, E. Saule, and C. Shen, "NSF/IEEE-TCPP Curriculum Initiative on Parallel and Distributed Computing – Core Topics for Undergraduates, Version II-beta," Tech. Rep., Nov. 2020, online. [Online]. Available: http://tcpp.cs.gsu.edu/curriculum/

[41] F. B. Baker and S.-H. Kim, *Item Response Theory: Parameter Estimation Techniques*, 2nd ed. Boca Raton: CRC Press, 2004. [Online]. Available: https://doi.org/10.1201/9781482276725

[42] M. Orlando and D. Thissen, "Likelihood-based item-fit indices for dichotomous item response theory models," *Applied Psychological Measurement*, vol. 24, no. 1, pp. 50–64, 2000.

[43] W. M. Yen, "Effects of local item dependence on the fit and equating performance of the three-parameter logistic model," *Applied Psychological Measurement*, vol. 8, no. 2, pp. 125–145, 1984. [Online]. Available: https://doi.org/10.1177/014662168400800201

[44] R. K. Hambleton, H. Swaminathan, and H. J. Rogers, *Fundamentals of Item Response Theory*, ser. Measurement Methods for the Social Sciences. Newbury Park, CA: Sage Publications, 1991.

[45] R. J. de Ayala, *The Theory and Practice of Item Response Theory*. New York: Guilford Press, 2009.