Codeless Modules for Parallel and Distributed Computing in Early Computing Curriculum

Chris Bourke
chris.bourke@unl.edu
University of Nebraska–Lincoln
Lincoln, Nebraska, USA

Abstract

We report on the results of a year-long intervention aimed at introducing parallel and distributed computing (PDC) concepts in introductory computing courses. In contrast to other efforts, our approach uses "codeless" modules that incorporate visualizations, simulations, and demonstrations, all of which do not require a programming background, making them accessible to novice computing students. To measure the effectiveness of our strategy, we designed and administered a pre/posttest to an intervention group ($n_2 = 75$) and compared it with a control group ($n_1 = 87$). A suite of nonparametric tests demonstrate that our codeless modules have a substantial positive effect on teaching PDC concepts to introductory computing students.

CCS Concepts

• **Social and professional topics** → **CS1**; • **Theory of computation** → *Parallel computing models*; *Distributed computing models*.

Keywords

Early Computing Education, CS1, Parallel and Distributed Computing

ACM Reference Format:
Chris Bourke. 2026. Codeless Modules for Parallel and Distributed Computing in Early Computing Curriculum. In *Proceedings of the 57th ACM Technical Symposium on Computer Science Education V.1 (SIGCSE TS 2026), February 18–21, 2026, St. Louis, MO, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3770762.3772500

1 Introduction

For more than a decade, "single-core" processors have essentially been legacy technology. Most processors manufactured today are multi- or manycore, yet introductory computing curricula have not generally adapted to this new reality. Instead, Parallel and Distributed Computing (PDC) remains primarily in upper-level computing curricula or even as *optional* elective courses (which are also among the least popular [3]).

There have been concerted efforts to infuse PDC topics into early computing curricula [5, 18, 29]. However, the main focus of these efforts has been on intermediate and upper-level elective courses [7, 27, 39]. PDC curriculum guidelines [18] themselves only identify a small number of topics appropriate for a typical Computer Science I (CS1) course.

This approach is understandable because the practical application of PDC knowledge requires a relatively high level of programming maturity and understanding of system operations—concepts which early computing students are unlikely to have encountered in their first year.

We believe there is an opportunity to introduce PDC concepts early in the computing curriculum (CS1) and even to non-computing majors. The challenge is to introduce these concepts in a manner that is accessible to novice computing students. To this end, we have developed a series of "codeless" PDC modules that do not require students to write or even understand any code. Instead, the modules focus on web-based interactive visualizations and simulations, making them accessible regardless of programming knowledge or computing background. Our approach follows a long line of highly successful research demonstrating that visualizations substantially improve student learning in computing [32, 35, 40].

To evaluate the effectiveness of our modules in introducing PDC concepts, we designed and administered a pre/posttest PDC evaluation over the course of an academic year in several CS1 courses. This study is a substantial extension of a prior pilot study [2]. The central research questions of this study are as follows:

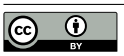
- Q1: Are our codeless modules effective in introducing PDC concepts to introductory computing students?
- Q2: Are students able to accurately gauge their own understanding of PDC concepts?

2 Related Work

For the last 15 years, there has been a concerted effort to introduce PDC topics throughout the computing curriculum and not just as one or two technical electives [5, 16, 25]. Most notably, the CDER Center has held dozens of workshops at several major conferences involving hundreds of computing faculty. A full PDC computing curriculum has been developed [18] as a companion to the ACM/IEEE Computing Curricula.

A number of publications have documented challenges and successes teaching PDC in undergraduate curricula (see the survey papers [7, 27, 39] for a broad overview). However, most efforts have focused on designing assignments and strategies for teaching PDC to more advanced students [23, 34]. Others have advocated for embedding smaller PDC modules throughout the entire computing curriculum [4, 20].

Some researchers have identified methods and benefits of introducing PDC concepts earlier in the computing curriculum, as well as for non-computing majors. Rague [31] developed an early visualization tool to reinforce parallel thinking for CS1 students.



This work is licensed under a Creative Commons Attribution 4.0 International License.
SIGCSE TS 2026, St. Louis, MO, USA.
© 2026 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-2256-1/26/02
https://doi.org/10.1145/3770762.3772500

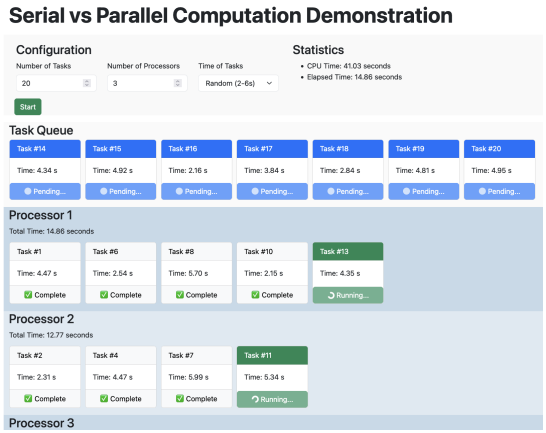


Figure 1: Screenshot of the parallel computing simulation

Gil [17] developed a data science curriculum intended for non-computing majors that did not require programming skills, using a visual programming tool to manage data workflows.

Others have investigated different ways to actively engage students without significant coding experience. Neeman *et al.* [28] outline a strategy for teaching PDC concepts using natural-language analogies instead of specific technologies. Bogaerts [1] introduced parallel programming through Scratch for younger students. Bruce *et al.* [6] introduced concurrency into a CS1 course in the context of event-based programming through graphics and animation. Others have developed "unplugged" activities which do not require students to use computers at all [11, 37, 38]. However, most of these approaches have not scaled well, nor have many studies taken a rigorous approach to measuring learning outcomes.

3 PDC Modules

Our "codeless" modules consist of three modules for parallel, asynchronous, and distributed computing. Each module is structured with a short introductory reading explaining each concept, followed by a web-based visualization/simulation and activity. The visualizations are interactive applications inspired by the highly successful approaches used to visualize programs [40], data structures and algorithms [35], and automata [32]. They can be run on any web browser without a special environment or software installation.

As an example, the module on parallel computing has a visual simulation that allows users to specify a number of generic computing tasks, a number of processors to execute them on, and how long each task will take. The simulation then queues the tasks for each processor to execute and reports the status of each task. The simulation tracks the total wall-clock time and processor time for the students to compare. Figure 1 contains an example screenshot.[1]

At the end of each module, students are prompted to reflect on the reading and their interaction with the visualizations by answering a series of questions designed to ensure that: 1) they ran a sufficient number of use cases in each simulation; and 2) their simulation experience directly traces back to the reading to reinforce understanding.

[1]See https://go.unl.edu/PDCatUNL for a live demo.

Since the pilot study [2], the modules have been reorganized and renamed into the three current modules to make the mapping of concepts more straightforward. Other minor updates included editing the readings, removal of superfluous activities, and streamlining of the demonstrations for easier execution.

4 Research Design

To measure how well our modules improved student learning and their confidence we designed a test that was administered both at the start of the semester (pretest) and again at the end of the semester (posttest) with the same content. Students were asked to provide free-form text answers to the following questions. For each of the four identified computing/PDC concepts: sequential (S), asynchronous (A), parallel (P), distributed (D) computing, students were asked to provide:

1) a definition (D) of the concept; and
2) an example (E) of the concept.

We refer to these 8 questions as Q-SD, Q-SE, Q-AD, Q-AE, Q-PD, Q-PE, Q-DD, and Q-DE. For example, Q-PD refers to the question: "Provide a definition for parallel computing." In addition, for each concept (pair of questions), students were asked to self-assess their confidence in the answers they gave on a 7-point Likert scale (7 = very confident, …, 4 = neither confident nor unsure, …, 1 = very unsure).

4.1 Assessment

Since student answers were free-form text, they were manually assessed twice by a single rater with 2 months between each assessment using the following 4-point rubric, with half-point fractional values in cases where the answer was judged to be between categories (7 possible discrete scores):

1. Needs Improvement - The student answer did not clearly demonstrate that they had an understanding of the concept.
2. Developing - The student seems to have a beginner's grasp on the concept, but there are key issues or omissions in their answer that indicate they do not yet fully understand the concept.
3. Adequate - The student has a good grasp on the concept. There may be some insufficiencies or omissions from their answer, but overall, they have demonstrated sufficient understanding of the concept.
4. Above Average - The answer goes above basic understanding and indicates a deeper appreciation of the concept.

To measure the reliability of our assessment, we used an intraclass correlation (ICC1, [36]). Initially, the two rating sets had a 0.7851 ICC value (95%CI [0.76, 0.81], F=8.3068, $p < 0.001$), which is interpreted as *good* [21]. To improve this, a third assessment was made on all ratings that differed by ≥ 1.0. After resolving differences, the ICC value improved to 0.9241 (95%CI [0.92, 0.93], F=25.3450, $p < 0.001$), which is interpreted as *excellent* (highest possible). For the results in section 5, we used the first rating post-resolution.

4.2 Administration

We administered our exam in 4 different CS1 courses at a medium-sized, midwestern R1 university over a full academic year, using 2 courses as a control group and 2 as an intervention group. For both groups, participation was required and part of their course grade. To encourage honest feedback, it was made clear that the accuracy of answers would *not* impact their grade, which was based entirely

on completing the pre/posttests. The study had an approved IRB, and students could opt-out of using their data for this study.

For each course, the pretest was administered at the end of the second week before any substantial material had been covered, while the posttest was administered in the final week of the course (all were 15-week courses). All four courses consisted primarily of computing or engineering freshmen.

The *Control Group* consisted of two spring CS1 courses. Each had distinct instructors, and though the content was mostly similar, one used C and the other Python as their programming languages. Neither course had any PDC content. The *Intervention Group* consisted of two fall CS1 courses taught by the same instructor; one used C, and the other was a smaller honors section that covered both C and Java in parallel. Students in the intervention group were required to complete our PDC modules at their own pace between week 2 and 12.

After filtering for incomplete data and students who opted out, the control and intervention groups consisted of $n_1 = 87$ and $n_2 = 75$ participants, respectively.

5 Results

All of our analysis utilizes nonparametric tools [14, 15] primarily because our data is ordinal (Likert scales) and is not normally distributed (we confirmed this using a Shapiro-Wilks test). However, with sample sizes as large as ours, the difference between nonparametric and parametric analysis tends to be minimal [12]. For completeness, we performed another analysis using a suite of parametric tests and found the results were essentially the same with respect to observed learning gains, statistical significance, etc., when compared with the nonparametric analysis presented here. Generally, the risk of using nonparametric analysis is that it is *less* likely to detect statistically significant effects (where parametric tests would otherwise find them). This makes our results all the more compelling.

We also performed several stratified analyses, separating students into (self-reported) high/low engagement (with the PDC modules), high/low course grade (AB/CDF), and each course separately. However, no significant differences were observed, and so for brevity, we report results of the intervention and control groups as a whole respectively.

5.1 Learning Gains

To determine whether our modules had an effect on student learning and understanding of PDC concepts, we compared the pretest and posttest scores for each of the 8 questions for all students in the intervention/control group separately.

To measure learning gains, we report both the pre- and post-results using the median and mode (central tendency), as well as the observed differences. To measure dispersion, we use the (true) interquartile range (IQR), which is the range of data between the first quartile (25th percentile) and the third quartile (75th percentile), giving a range of the middle 50% of the data. These are more appropriate (than mean/standard deviation) for ordinal, non-normal data [15]. Generally, a larger IQR indicates a greater spread, while smaller values indicate a clustering of data.

To measure statistical significance, we use the Wilcoxon Signed-Rank Test ($\alpha = 0.05$). The null hypothesis (H_0) is that the median difference between the paired values (pre- and post-results) is zero (i.e., no effect), while the alternative hypothesis (H_1) is that the median difference between the paired values is not zero (i.e., there is an effect).

In addition, we report on *effect size*, a measure of the strength or magnitude of the effect of an intervention [22]. For parametric analysis, Cohen's d (which uses mean/standard deviations) is often used [9]. However, for nonparametric analysis, we use Cliff's delta (δ), a more robust measure [8, 26]. Reported on the scale [-1, 1], a negative value indicates that the posttest results are greater than the pretest results. The magnitude indicates the effect size. We use the interpretations (negligible, small, medium, large) recommended by [19].

5.1.1 Intervention Group. For the intervention group, all figures are reported in Table 1.

With respect to both median and mode, students showed significant improvement in understanding the three PDC concepts. With only one exception (Q-AE, providing an example of asynchronous computing), every median score increased by at least half a Likert score, with most showing a full point improvement. On a four-point scale, this improvement is substantial.

For all three PDC topics, the IQR *increased*. Typically, this would mean that the post-results had greater variability. However, considering that the baseline (pre-test results) were very low (in terms of median and mode) and were highly clustered (in most cases around 1, the lowest possible score), the increased dispersion is to be expected. The interpretation is that nearly all students had essentially no PDC knowledge or experience prior to the PDC modules. However, after the PDC intervention, a substantial number demonstrated developing or adequate PDC knowledge. This is what was expected; student understanding was not required to be anywhere close to the level of expert or even that of an advanced undergraduate student.

The improvement is further confirmed by the Wilcoxon analysis. For every single question, the difference was statistically significant (True), and we *reject* the null hypothesis and infer that there is an effect between the pre- and post-results.

Finally, the effect size is also substantial. For all three PDC concepts, every question had at least a medium effect size. In fact, three questions (Q-AD, Q-PD, Q-PE) fell short of *large* (0.474) by only a few hundredths of a measure. Question Q-SD (define sequential computing) had the smallest effect size, which is understandable since the baseline was already pretty good (students already had a good understanding of sequential computing in the first weeks of the course, though they couldn't necessarily give a good example, as observed by the performance and subsequent large improvement in Q-SE).

5.1.2 Control Group. For the control group, all figures are reported in Table 2. As expected, the control group showed significant learning gains with respect to sequential computing (Q-SD, Q-SE), which was the focus of all CS1 sections. However, with one exception, the control group showed no significant or even observable improvements with respect to PDC concepts. Neither the median nor mode scores changed from the pre- to post-test. Likewise, the IQR for

Table 1: Learning Gains - Intervention Group. Performance analysis for the intervention group ($n_2 = 75$) for each of the 8 questions. We report pre/post test scores and differences for the median and mode. The interquartile range (IQR) is reported along with the 25th/75th percentiles. The statistical significance (sig?) using the Wilcoxon statistic (W) is reported along with its p-value ($\alpha = 0.05$). Effect Size is reported using Cliff's delta and its interpretation (interp).

	Median			Mode			IQR			Wilcoxon			Effect Size	
Question	Pre	Post	Diff	Pre	Post	Diff	Pre	Post	Diff	W	p-value	sig?	δ	interp
Q-SD	2.0	2.5	0.5	2.0	2.5	0.5	1.75 [1.00, 2.75]	0.50 [2.00, 2.50]	-1.25	551.00	0.027	True	-0.21	small
Q-SE	1.5	2.5	1.0	1.0	2.5	1.5	1.00 [1.00, 2.00]	0.75 [2.00, 2.75]	-0.25	61.00	0.000	True	-0.63	large
Q-AD	1.0	2.0	1.0	1.0	1.0	0.0	0.00 [1.00, 1.00]	2.00 [1.00, 3.00]	+2.00	138.00	0.000	True	-0.47	medium
Q-AE	1.0	1.0	0.0	1.0	1.0	0.0	0.00 [1.00, 1.00]	1.25 [1.00, 2.25]	+1.25	145.00	0.000	True	-0.33	medium
Q-PD	1.5	2.0	0.5	1.0	2.0	1.0	1.00 [1.00, 2.00]	1.25 [1.75, 3.00]	+0.25	130.00	0.000	True	-0.43	medium
Q-PE	1.0	2.0	1.0	1.0	1.0	0.0	0.00 [1.00, 1.00]	2.00 [1.00, 3.00]	+2.00	69.50	0.000	True	-0.46	medium
Q-DD	1.0	3.0	2.0	1.0	4.0	3.0	1.00 [1.00, 2.00]	2.75 [1.25, 4.00]	+1.75	33.50	0.000	True	-0.62	large
Q-DE	1.0	2.0	1.0	1.0	1.0	0.0	0.25 [1.00, 1.25]	2.00 [1.00, 3.00]	+1.75	92.50	0.000	True	-0.38	medium

the three PDC topics was essentially unchanged. The Wilcoxon analysis confirms that none of the changes (save for Q-PD) were statistically significant, and all the effect sizes were negligible. This strongly suggests that the control group showed no significant learning gains with respect to PDC topics.

The one anomaly was the question asking students to define parallel computing (Q-PD). We performed a closer analysis by separating the two course sections and found that the improvements in one section were not statistically significant ($p = 0.113$), while the other showed only slight significance ($p = 0.039$). In both cases, the effect size was still negligible. This suggests that one section may have benefited from some ancillary coverage of parallel computing, but the learning gains were still small compared to the intervention group.

5.1.3 Comparison. To compare the intervention to the control group directly, we applied a Mann-Whitney U Test [24] to the posttest results which are reported in Table 3. We report the effect size using $r = \frac{Z}{\sqrt{N}}$, where Z is the Z-score, as suggested by Rosenthal [33] for nonparametric tests. A negative value indicates that the intervention group's scores were higher than the control group. We use Cohen's [9] interpretations for practical significance.

For all topics, the intervention group showed statistically significantly better scores than the control group, with a substantial effect size. Nearly every PDC question showed medium-to-large effects, with most approaching the "large" threshold of 0.5 suggested by Cohen. The only small effect (Q-AE) was very near the medium threshold (0.3).

The smallest effect was for Q-SD (define sequential computing), which is expected, as both groups had similar exposure and experience in sequential computing from their courses. However, there was a more substantial effect for providing an example of sequential computing. This suggests that although both groups were able to define sequential computing, the intervention group was better able to articulate an example. We interpret this as a side effect of our modules: students in the intervention group had a clearer contrast between PDC and sequential computing, which enabled them to provide better examples of sequential computing.

Together with the previous results, this strongly suggests that our codeless PDC modules had a significant, if not substantial, effect on student learning and understanding of PDC concepts.

5.1.4 Controlled Comparison. The baseline for prior knowledge in both groups was low, which suggests that the improvements (or lack thereof in the control group) were not a result of prior knowledge. To confirm this, we applied an ANCOVA test (Quade's Analysis of Covariance) [30] which is a blend of ANOVA and regression and assesses whether there are significant differences between groups while controlling for the influence of one or more covariates. In this case, we controlled for prior knowledge as evidenced by the pretest scores.

Results are reported in Table 4. A higher F statistic provides stronger evidence that the means of the two groups are different, even when controlling for the pretest. For all topics, the p value indicates that the differences are statistically significant. However, for the question asking students to define sequential computing (Q-SD), the results are not as strong as for the PDC topics. This aligns with the other results: students in both groups have good knowledge of sequential computing, but only the intervention group consistently showed improved knowledge of PDC concepts, *even when controlling for prior knowledge.*

The mean signed ranks further support this interpretation. All of the values for the control group are negative, indicating that most posttest scores were lower than expected (based on the pretest scores), while all the values for the intervention group are positive, indicating that most posttest scores were higher than expected by the model.

5.2 Student Confidence Assessment

In addition to measuring students' performance, we were interested in how student confidence in these topics changed from the pre- to posttest. We were also interested in determining whether students could accurately gauge their own level of knowledge.

To determine how student confidence changed between the pre- and posttest, we used a Wilcoxon analysis. To measure whether students could accurately assess their own knowledge, we used a Spearman rank correlation coefficient [10]. Measured on a symmetric $[-1, 1]$ scale, a high correlation (near +1) indicates that high confidence is associated with more correct answers, and that low confidence is associated with more incorrect answers. A low correlation (near 0) would indicate that students are not able to accurately estimate their own understanding of a topic.

Table 2: Learning Gains - Control Group. Performance analysis for control group ($n_1 = 87$) for each of the 8 questions. We report pre/post test scores and differences for the median and mode. The interquartile range (IQR) is reported along with the 25th/75th percentiles. The statistical significance (sig?) using the Wilcoxon statistic (W) is reported along with its p-value ($\alpha = 0.05$). Effect Size is reported using Cliff's delta and its interpretation (interp).

	Median			Mode			IQR			Wilcoxon			Effect Size	
Question	Pre	Post	Diff	Pre	Post	Diff	Pre	Post	Diff	W	p-value	sig?	δ	interp
Q-SD	1.0	2.0	1.0	1.0	1.0	0.0	1.00 [1.00, 2.00]	1.50 [1.00, 2.50]	+0.50	468.0	0.027	True	-0.19	small
Q-SE	1.0	2.0	1.0	1.0	1.0	0.0	0.00 [1.00, 1.00]	1.00 [1.00, 2.00]	+1.00	153.0	0.000	True	-0.27	small
Q-AD	1.0	1.0	0.0	1.0	1.0	0.0	0.00 [1.00, 1.00]	0.50 [1.00, 1.50]	+0.50	270.5	0.856	False	-0.03	negligible
Q-AE	1.0	1.0	0.0	1.0	1.0	0.0	0.00 [1.00, 1.00]	0.00 [1.00, 1.00]	0.00	91.5	0.610	False	0.01	negligible
Q-PD	1.0	1.0	0.0	1.0	1.0	0.0	0.50 [1.00, 1.50]	0.50 [1.00, 1.50]	0.00	171.0	0.009	True	-0.11	negligible
Q-PE	1.0	1.0	0.0	1.0	1.0	0.0	0.25 [1.00, 1.25]	0.00 [1.00, 1.00]	-0.25	210.0	0.634	False	0.04	negligible
Q-DD	1.0	1.0	0.0	1.0	1.0	0.0	0.50 [1.00, 1.50]	1.00 [1.00, 2.00]	+0.50	194.5	0.191	False	-0.09	negligible
Q-DE	1.0	1.0	0.0	1.0	1.0	0.0	0.00 [1.00, 1.00]	0.00 [1.00, 1.00]	0.0	177.5	0.380	False	0.01	negligible

Table 3: Intervention vs Control Group (posttest). Comparison of performance on the posttest of the intervention versus the control group using a Mann-Whitney U Test. The U statistic is reported along with the p-value and whether or not it was statistically significant ($\alpha = 0.05$). Effect size r is reported along with its practical significance (interp).

Topic	U	p-value	sig?	r	interp
Q-SD	2544.5	0.0132	True	-0.189	small
Q-SE	1589.5	0.0000	True	-0.442	medium
Q-AD	1971.5	0.0000	True	-0.341	medium
Q-AE	2243.5	0.0000	True	-0.269	small
Q-PD	1700.5	0.0000	True	-0.412	medium
Q-PE	1732.5	0.0000	True	-0.404	medium
Q-DD	1520.0	0.0000	True	-0.460	medium
Q-DE	2016.0	0.0000	True	-0.329	medium

Table 4: Intervention vs Control Group - Controlling for Prior Knowledge. Comparison of performance on the posttest of the intervention versus the control group controlling for scores on the pretest using Quade's ANCOVA. The F statistic is reported along with the p-value and whether or not it was statistically significant ($\alpha = 0.05$). The mean signed ranks are reported for each group.

	Quade Statistics			mean signed ranks	
Topic	F	p-value	sig?	control	intervention
Q-SD	5.2149	0.0237	True	-15.4195	18.0067
Q-SE	31.7845	0.0000	True	-29.2011	47.0333
Q-AD	27.1775	0.0000	True	-44.2816	26.1267
Q-AE	19.0292	0.0000	True	-37.5690	21.7267
Q-PD	28.0850	0.0000	True	-42.7989	29.2600
Q-PE	50.3634	0.0000	True	-48.5402	41.9733
Q-DD	58.9563	0.0000	True	-42.3218	54.4933
Q-DE	29.3826	0.0000	True	-48.3391	23.6600

5.2.1 Intervention Group. Results for the intervention group can be found in Tables 5 and 6.

For all four topics, students in the intervention group demonstrated substantial and statistically significant improvements in the confidence of their answers. The biggest improvements were in asynchronous and distributed computing, with median scores improving by 3 full Likert points in both topics. Sequential computing also saw a statistically significant improvement, but the change was more modest since students were already quite confident in their pretest answers.

Table 5: Student Confidence (Intervention Group). Pre/post medians (and differences) of self-reported confidence for each topic pair. Statistical significance (sig?) is reported using the Wilcoxon statistic (W); all p-values were negligible ($\alpha = 0.05$).

	Median			Wilcoxon	
Topic	Pre	Post	Diff	W	sig?
Q-S	5.0	6.0	1.0	144.00	True
Q-A	2.0	5.0	3.0	174.00	True
Q-P	3.0	5.0	2.0	201.50	True
Q-D	2.0	5.0	3.0	206.00	True

Table 6: Student Confidence Correlation (Intervention Group). Pre/post results using Spearman rank correlation coefficient (ρ) with p-values and significance (sig?) ($\alpha = 0.05$).

	Spearman (pre)			Spearman (post)		
Topic	ρ	p-value	sig?	ρ	p-value	sig?
Q-S	0.292	0.011	True	0.128	0.273	False
Q-A	0.274	0.017	True	0.354	0.002	True
Q-P	0.321	0.005	True	0.439	0.000	True
Q-D	0.422	0.000	True	0.285	0.013	True

For both pre- and posttests, the Spearman analysis indicated a moderate positive correlation between students' answers and their confidence. For all topics (except sequential computing in the posttest), the correlation was statistically significant. This suggests that students had a reasonably accurate self-assessment of their own answers, whether correct or not.

Between the pre- and post-test results, topics such as asynchronous and parallel computing saw an increase in the correlation, while sequential and distributed computing showed an inverse relationship. We interpret this to mean that students initially thought they had a good understanding of sequential and distributed computing (due to familiarity with the terms themselves), but after

going through the course and modules, their confidence was tempered by increased awareness of the complexity of these subjects. At the same time, students may have been unfamiliar with the terms "asynchronous" and "parallel" (at least in the context of computing), so their confidence was low to begin with. The modules provided a foundation and context for these concepts, which likely boosted their confidence.

None of these results were negatively correlated. We interpret this to mean that participants avoided the Dunning-Kruger effect [13] (i.e., being overly confident in incorrect answers while having low confidence in correct ones).

5.2.2 Control Group. Results for the control group can be found in Tables 7 and 8.

Table 7: Student Confidence (Control Group). Pre/post medians (and differences) of self-reported confidence for each topic pair. Statistical significance (sig?) is reported using the Wilcoxon statistic (W); all p-values were negligible ($\alpha = 0.05$).

	Median			Wilcoxon	
Topic	**Pre**	**Post**	**Diff**	W	**sig?**
Q-S	3.0	5.0	2.0	313.00	True
Q-A	2.0	3.0	1.0	353.00	True
Q-P	2.0	3.0	1.0	400.00	True
Q-D	1.0	3.0	2.0	427.00	True

Table 8: Student Confidence Correlation (Control Group). Pre/post results using Spearman rank correlation coefficient (ρ) with p-values and significance (sig?) ($\alpha = 0.05$).

	Spearman (pre)			Spearman (post)		
Topic	ρ	p-value	sig?	ρ	p-value	sig?
Q-S	0.391	0.000	True	0.421	0.000	True
Q-A	0.328	0.002	True	0.281	0.008	True
Q-P	0.349	0.001	True	0.433	0.000	True
Q-D	0.346	0.001	True	0.359	0.001	True

Similar to the intervention group, the control group demonstrated statistically significant increases in the confidence of their answers, although the changes were not substantial. The Spearman analysis also indicates that the control group had a reasonably accurate self-assessment of their own answers, whether correct or not. Given that there were little to no gains in PDC knowledge, we interpret this to mean that by the end of the course, control students were more confident—and quite accurate in that confidence—that they did *not* know much about PDC topics.

6 Discussion

In answer to our first research question, there is substantial evidence to suggest that our codeless modules are effective in introducing PDC concepts to introductory computing students. The intervention group showed statistically significant improvements in their knowledge from the pretest to the posttest, while the control group did not. In nearly every topic, the intervention group also showed substantial improvements with respect to effect size, while any improvements in the control group were negligible. The same results were demonstrated when comparing the control and intervention groups directly, even when controlling for prior knowledge.

With respect to our second research question, there is also substantial evidence that students are able to accurately gauge their own understanding of PDC concepts. Both the control and intervention groups showed improved confidence in their answers from the pretest to the posttest. Though both improvements were statistically significant, the control group's improvements were more modest. In light of the improved learning gains for the intervention group (and lack of gains for the control group), students in the intervention group were able to accurately gauge their knowledge of PDC, while those in the control group were able to accurately gauge their *lack* of PDC knowledge.

6.1 Limitations & Risks

We acknowledge the limitations of our study and its applicability in the context of the usual factors and risks to validity that affect studies like this. Though our dataset size was substantial, our intervention was limited to one institution. A larger, multi-institutional study would be needed to assess the broader applicability of our findings.

Our study also used a single rater, which may have affected the assessment of student answers. However, we mitigated the risks by performing multiple independent ratings over time and utilized intraclass correlation to verify and improve consistency.

Though we attempted to control for other factors as much as possible, the results may have been adversely affected by students independently accessing other PDC materials outside of our modules. Although there was no incentive to do so, it is possible that students accessed materials while taking the pretest/posttest, which could have impacted the accuracy of the exam as an assessment of their knowledge.

7 Conclusion

While most PDC efforts have focused on programming strategies for upper-level and elective courses, we believe there is value and opportunity in introducing these concepts much earlier in the computing curriculum. Our "codeless" approach offers a simple, accessible, and scalable way to introduce PDC concepts to both computing majors and non-majors in CS1.

Our present study provides significant evidence that our approach is an effective tool for introducing PDC concepts and topics to novice computing students without introducing undue complexity. We hope that other educators will adopt or adapt our modules and replicate our results and success. We intend to continue developing and refining our modules and expanding our study to other institutions.

Acknowledgments

We thank Dr. Neena Thota and everyone at the CDER Center for useful discussions. This work was supported in part by NSF 2321015.

References

[1] Steven Bogaerts. 2013. Hands-On Exploration of Parallelism for Absolute Beginners with Scratch. In *2013 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum*. 1263–1268. https://doi.org/10.1109/IPDPSW.2013.63

[2] Chris Bourke and Justin Firestone. 2024. Codeless PDC Modules for Early Computing Curriculum. *2024 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)* (2024), 357–364. https://api.semanticscholar.org/CorpusID:271040190

[3] Steven Bradley, Miranda C. Parker, Rukiye Altin, Lecia Barker, Sara Hooshangi, Thom Kunkeler, Ruth G. Lennon, Fiona McNeill, Julià Minguillón, Jack Parkinson, Svetlana Peltsverger, and Naaz Sibia. 2023. Modeling Women's Elective Choices in Computing. In *Proceedings of the 2023 Working Group Reports on Innovation and Technology in Computer Science Education* (Turku, Finland) *(ITiCSE-WGR '23)*. Association for Computing Machinery, New York, NY, USA, 196–226. https://doi.org/10.1145/3623762.3633497

[4] Richard Brown and Elizabeth Shoop. 2011. Modules in Community: Injecting More Parallelism into Computer Science Curricula. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education* (Dallas, TX, USA) *(SIGCSE '11)*. Association for Computing Machinery, New York, NY, USA, 447–452. https://doi.org/10.1145/1953163.1953293

[5] Richard Brown and Elizabeth Shoop. 2012. CSinParallel and Synergy for Rapid Incremental Addition of PDC Into CS Curricula. In *2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum*. 1329–1334. https://doi.org/10.1109/IPDPSW.2012.166

[6] Kim B. Bruce, Andrea Danyluk, and Thomas Murtagh. 2010. Introducing Concurrency in CS 1. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education* (Milwaukee, Wisconsin, USA) *(SIGCSE '10)*. Association for Computing Machinery, New York, NY, USA, 224–228. https://doi.org/10.1145/1734263.1734341

[7] Jose Aprigio Carneiro Neto, Antonio Jose Alves Neto, and Edward David Moreno. 2022. A Systematic Review on Teaching Parallel Programming. In *Proceedings of the 11th Euro American Conference on Telematics and Information Systems* (Aveiro, Portugal) *(EATIS '22)*. Association for Computing Machinery, New York, NY, USA, Article 14, 4 pages. https://doi.org/10.1145/3544538.3544659

[8] Norman Cliff. 1993. Dominance statistics: Ordinal analyses to answer ordinal questions. *Psychological Bulletin* 3, 114 (1993), 494––509. https://doi.org/10.1037/0033-2909.114.3.494

[9] Jacob Cohen. 1988. *Statistical Power Analysis for the Behavioral Sciences* (2nd ed.). Routledge, New York. https://doi.org/10.4324/9780203771587

[10] W. J. Conover. 1999. *Practical Nonparametric Statistics* (3 ed.). John Wiley & Sons.

[11] Anurag Dasgupta, Venkat Margapuri, Simon Shamoun, Shubbhi Taneja, and Matthew Toups. 2024. Hands-On Learning: Teaching Parallel and Distributed Computing through Unplugged Activities in Undergraduate CS Courses. In *SC24-W: Workshops of the International Conference for High Performance Computing, Networking, Storage and Analysis*. 363–369. https://doi.org/10.1109/SCW63240.2024.00054

[12] Joost de Winter and Dimitra Dodou. 2010. Five-Point Likert Items: t-Test Versus Mann–Whitney–Wilcoxon. *Practical Assessment, Research & Evaluation* 15, 11 (10 2010). https://doi.org/10.7275/bj1p-ts64

[13] David Dunning and Justin Kruger. 1999. Unskilled and unaware of it: How difficulties in recognizing one's own incompetence lead to inflated self-assessments. *Journal of Personality and Social Psychology* 77, 6 (1999), 1121–1134. https://doi.org/10.1037/0022-3514.77.6.1121

[14] Jim Frost. 2024. *Hypothesis Testing: An Intuitive Guide for Making Data Driven Decisions*. Statistics by Jim Publishing. https://books.google.com/books?id=vqLx0AEACAAJ

[15] Jim Frost. 2024. *Introduction to Statistics: An Intuitive Guide for Analyzing Data and Unlocking Discoveries*. Statistics By Jim Publishing. https://books.google.com/books?id=vTPl0AEACAAJ

[16] Sheikh Ghafoor, David W. Brown, and Mike Rogers. 2018. Integrating Parallel Computing in Introductory Programming Classes: An Experience and Lessons Learned. In *Euro-Par 2017: Parallel Processing Workshops*, Dora B. Heras, Luc Bougé, Gabriele Mencagli, Emmanuel Jeannot, Rizos Sakellariou, Rosa M. Badia, Jorge G. Barbosa, Laura Ricci, Stephen L. Scott, Stefan Lankes, and Josef Weidendorfer (Eds.). Springer International Publishing, Cham, 216–226.

[17] Yolanda Gil. 2014. Teaching Parallelism without Programming: A Data Science Curriculum for Non-CS Students. In *Proceedings of the Workshop on Education for High-Performance Computing* (New Orleans, Louisiana) *(EduHPC '14)*. IEEE Press, 42–48. https://doi.org/10.1109/EduHPC.2014.12

[18] TCPP Curriculum Working Group. 2020. NSF/IEEE-TCPP curriculum initiative on parallel and distributed computing – core topics for undergraduates. https://tcpp.cs.gsu.edu/

[19] Melinda Hess and Jeffrey D. Kromrey. 2004. Robust Confidence Intervals for Effect Sizes: A Comparative Study of Cohen's d and Cliff's Delta Under Non-normality and Heterogeneous Variances. https://api.semanticscholar.org/CorpusID:17670117

[20] David J. John and Stan J. Thomas. 2014. Parallel and Distributed Computing across the Computer Science Curriculum. In *2014 IEEE International Parallel & Distributed Processing Symposium Workshops*. 1085–1090. https://doi.org/10.1109/IPDPSW.2014.121

[21] Terry K. Koo and Mae Y. Li. 2016. A Guideline of Selecting and Reporting Intraclass Correlation Coefficients for Reliability Research. *Journal of Chiropractic Medicine* 15, 2 (2016), 155–163. https://doi.org/10.1016/j.jcm.2016.02.012

[22] Mark W Lipsey and David B. Wilson. 1993. The efficacy of psychological, educational, and behavioral treatment. Confirmation from meta-analysis. *The American psychologist* 48, 12 (1993), 1181–1209. https://doi.org/10.1037/0003-066X.48.12.1181

[23] Jie Liu. 2016. 20 Years of Teaching Parallel Processing to Computer Science Seniors. *2016 Workshop on Education for High-Performance Computing (EduHPC)* (2016), 7–13.

[24] Henry B. Mann and Donald R. Whitney. 1947. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics* 18, 1 (1947), 50–60. https://doi.org/10.1214/aoms/1177730491

[25] Suzanne J. Matthews. 2020. PDCunplugged: A Free Repository of Unplugged Parallel Distributed Computing Activities. In *2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. 284–291. https://doi.org/10.1109/IPDPSW50202.2020.00060

[26] Kane Meissel and Esther S. Yao. 2024. Using Cliff's Delta as a Non-Parametric Effect Size Measure: An Accessible Web App and R Tutorial. *Practical Assessment, Research, and Evaluation* 29, 2 (2024). https://doi.org/10.7275/pare.1977

[27] Nkundwe Moses Mwasaga and Mike Joy. 2020. Using High-Performance Computing Artifacts as a Learning Intervention: A Systematic Literature Review. In *Proceedings of the 2nd International Conference on Intelligent and Innovative Computing Applications* (Plaine Magnien, Mauritius) *(ICONIC '20)*. Association for Computing Machinery, New York, NY, USA, Article 42, 10 pages. https://doi.org/10.1145/3415088.3415130

[28] Henry Neeman, Lloyd Lee, Julia Mullen, and Gerard Newman. 2006. Analogies for Teaching Parallel Computing to Inexperienced Programmers. In *Working Group Reports on ITiCSE on Innovation and Technology in Computer Science Education* (Bologna, Italy) *(ITiCSE-WGR '06)*. Association for Computing Machinery, New York, NY, USA, 64–67. https://doi.org/10.1145/1189215.1189172

[29] Sushil K. Prasad, Anshul Gupta, Krishna Kant, Andrew Lumsdaine, David Padua, Yves Robert, Arnold Rosenberg, Alan Sussman, and Charles Weems. 2012. Literacy for All in Parallel and Distributed Computing: Guidelines for an Undergraduate Core Curriculum. *CSI Journal of Computing* 1, 2 (2012).

[30] Dana Quade. 1979. Using Weighted Rankings in the Analysis of Complete Blocks with Additive Block Effects. *J. Amer. Statist. Assoc.* 74, 367 (1979), 680–683. https://doi.org/10.1080/01621459.1979.10481670

[31] B. Rague. 2009. Teaching parallel thinking to the next generation of programmers. *Journal of Education, Informatics and Cybernetics* 1, 1 (2009), 43—-48.

[32] Susan Rodger. 2006. Learning Automata and Formal Languages Interactively with JFLAP. *SIGCSE Bull.* 38, 3 (jun 2006), 360. https://doi.org/10.1145/1140123.1140270

[33] Robert Rosenthal. 1991. *Meta-Analytic Procedures for Social Research* (revised ed.). Vol. 6. Sage Publications.

[34] Erik Saule. 2018. Experiences on Teaching Parallel and Distributed Computing for Undergraduates. *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)* (2018), 361–368.

[35] Clifford A. Shaffer, Matthew L. Cooper, Alexander Joel D. Alon, Monika Akbar, Michael Stewart, Sean Ponce, and Stephen H. Edwards. 2010. Algorithm Visualization: The State of the Field. *ACM Trans. Comput. Educ.* 10, 3, Article 9 (aug 2010), 22 pages. https://doi.org/10.1145/1821996.1821997

[36] Patrick E. Shrout and Joseph L. Fleiss. 1979. Intraclass correlations: uses in assessing rater reliability. *Psychological bulletin* 86 2 (1979), 420–8. https://doi.org/10.1037//0033-2909.86.2.420

[37] Mary L. Smith and Srishti Srivastava. 2023. Introducing Parallel and Distributed Computing concepts through the use of Flashcards and a Card Game. In *2023 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. 278–283. https://doi.org/10.1109/IPDPSW59300.2023.00053

[38] Srishti Srivastava, Mary Smith, Amrita Ghimire, and Shaun Gao. 2019. Assessing the Integration of Parallel and Distributed Computing in Early Undergraduate Computer Science Curriculum using Unplugged Activities. *2019 IEEE/ACM Workshop on Education for High-Performance Computing (EduHPC)* (2019), 17–24.

[39] Xiaoyuan Suo, Olga Glebova, David Liu, Alina Lazar, and Doina Bein. 2021. A Survey of Teaching PDC Content in Undergraduate Curriculum. *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)* (2021), 1306–1312.

[40] Jaime Urquiza-Fuentes and J. Ángel Velázquez-Iturbide. 2009. A Survey of Successful Evaluations of Program Visualization and Algorithm Animation Systems. *ACM Trans. Comput. Educ.* 9, 2, Article 9 (jun 2009), 21 pages. https://doi.org/10.1145/1538234.1538236