Iyer, S. et al. (Eds.) (2022). Proceedings of the 30th International Conference on Computers in Education.
Asia-Pacific Society for Computers in Education

Learning Computational Thinking Efficiently with Block-based Parsons Puzzles

Jeff BENDER [a*], Alex DZIENA[a] & Gail KAISER[a]
[a]*Programming Systems Laboratory, Columbia University, USA*
*jeffrey.bender@columbia.edu

Abstract: To investigate learning system elements and progressions that affect computational thinking (CT) learning in block-based environments, we developed a Parsons Programming Puzzle (PPP) module within Scratch with scaffolding customized via a novel Blockly grammar. By varying the presentation and types of feedback encountered between- and within-subjects in a study of 579 adults, we identified features and scaffolding strategies that yield manageable cognitive load (CL), improved CT learning efficiency, and increased motivation, for a general populace. Findings indicate: 1) PPPs with feedback induce lowest CL; 2) an isolated palette, correctness feedback, and fading correctness feedback increase learning efficiency; 3) fading scaffolding can increase CT motivation. We analyze 12 conditions to provide insight to those developing block-based PPP systems with the aim to advance equitable CT education for all.

Keywords: computational thinking, parsons programming puzzles, scratch, motivation, cognitive load

1. Introduction

Increasingly, government and education leaders position computer science (CS) as a foundation useful for learning other disciplines, for empowering active citizenship, and for addressing inequalities (Blickstein, 2018). Computational thinking (CT) (Wing, 2006), whether characterized broadly as inclusive of the material and social elements of participatory computational literacy, or narrowly as the cognitive skills and knowledge necessary to reason effectively with a machine, has mobilized expanded access to CS (Proctor et al., 2018). Internationally, many schools have introduced compulsory coding (e.g. Whitehouse.gov, 2016; Hamilton-smith, 2016), and the CS exposure movement, inclusive of Code.org's Hour of Code (Code.org, n.d.) and many more (e.g. Girls Who Code, n.d., Black Girls Code, n.d.), reach students from schools that have not. This traction, while important, can limit focus on the sustained activity necessary to learn CT effectively (Blickstein et al., 2019).

The emerging discipline of learning engineering, which combines theory with data-driven analysis to develop educational methodologies that produce high-quality learning, offers the CS community an architecture to investigate efficient CT learning (Baker et al., 2022). This field encourages the instrumentation of learning environments, development of student learning models, and rich personalization supportive of equity. It guides us to design human-computer systems that harness and leverage the codependences between learning system, student, and teacher, to operationalize CT learning for all. This harmony among roles is important during an era of generational interdependency in which K-12 students and teachers alike are learning CT. In the U.S., 20% of CS teachers describe themselves as overwhelmed and 30% as under-qualified (Schaffhauser, 2018). The limited CT understanding can produce gaps in teacher pedagogical content knowledge, and testing-tuned curricula constrain time afforded to CT study (Celepkolu et al., 2020). While opportunities exist to infuse CT across curricula (c.f. Campos et al., 2017; Pollock et al., 2019), especially in STEM subjects (Cao et al., 2020), efficient CT learning remains crucial for universal access initiatives to succeed.

Though learning CT and coding can be difficult for novices (Anderson, 1982; Pea, 1986; Winslow, 1996), an approach offering efficiencies involves Parsons Programming Puzzles (PPPs), which are program completion exercises that facilitate learners practicing CT by assembling into correct order programming constructs that comprise examples of well-written code, typically focused on a single concept (Parsons et al., 2006). While PPPs originated in text-based formats (Half-Baked

Software, n.d.; Ihantola et al., 2010), researchers have introduced them to block-based environments, like Looking Glass and Scratch (Harms et al., 2016; Bender et al., 2021). Scratch, independently, has had success in attracting a community of over one million active users (MIT Media Lab, n.d.), but it is often used as a minimally guided entry to CT, leaving learners with misconceptions regarding sequences, Booleans, loops, and events (Franklin et al., 2020; Grover et al, 2015, 2017), and with inconsistent capability to demonstrate increased skill over time (Scaffidi et al., 2012). By balancing the agency Scratch offers learners, the integration of PPPs provides structure that can focus learning more narrowly, but efficiently, on the cognitive skill and knowledge elements of CT (Bender et al., 2022).

This approach does not offer the learner experience in the broader aspects of participatory computational literacy, however. Like the strategy in (Brusilovsky et al., 2018), in which learners extend their skills from code comprehension to construction by reading examples, viewing animations, and addressing misconceptions before trying PPPs, we propose learners might benefit from PPP practice in which scaffolding fades as CT concept familiarity develops. By decrementing supports provided by puzzle presentation and feedback, we can offer learners direct instruction to start (Kim et. al, 2005), then provide pathways toward open-ended CT via creation of personally meaningful artifacts, in alignment with Scratch's roots in constructionism (Brennan, 2013; Garner et. al, 2019).

To investigate this approach, we ran an online study targeting adults who were randomly selected into 1 of 12 conditions to learn the CT concept *looping* via 7 PPPs. Between conditions, we varied the presentation of programming constructs, the feedback types, and for control conditions, feedback activation or the CT concept. In 3 conditions, we faded scaffolding as learners progressed by suppressing correctness feedback and/or by presenting additional palette and block options. We investigate 3 research questions: after at most 56 minutes of puzzle solving, what are the effects of PPP variation on adult learner: **R1**) cognitive load (CL)?; **R2**); learning efficiency?; **R3**) CT motivation? Findings from 579 participants indicate **F1**) PPPs with feedback induce lowest CL; **F2**) an isolated palette, correctness feedback, and fading correctness feedback increases learning efficiency; **F3**) fading scaffolding can increase CT motivation. We first review related work and the software developed, then the study purpose, summative evaluation, and results, before previewing future work.

2. Related Work

Since PPP emerged as a new form of program completion problem in 2006, (Ericson et al., 2018) has assembled research results, (Du et al., 2020) has presented the variants in the literature, and (Bender et al., 2021) has documented strengths and weaknesses. Generally, PPPs offer learning efficiency gains because scaffolded support enables students to train in shorter time than via code writing exercises, while performing similarly on transfer tasks using either approach. PPPs also have led to more learning gain than alternative formats such as animations and annotated examples (Brusilovsky et al., 2018).

Our PPP integration with Scratch differs from previous PPP implementations by its embrace of economical game-based learning (GBL) instructional design, the variability of the scaffolding presented to the learner, and its embedding in a popular block-based environment. By leveraging game thinking (Kim, 2018), a blend of game design (Salen et al, 2003) with design thinking (Plattner, 2013), we facilitate the construction of engaging learning experiences with limited content development investment compared to serious games, as recommended in (Dickey, 2006). Educators can design motivating learning progressions that increase in difficulty while fading scaffolding and assessing implicitly, as in stealth assessment (Shute et al., 2021), as the learner masters individual CT concepts.

Like most PPPs, our implementation provides feedback as the learner positions programming constructs. We extend this with the configurability of per-construct points, a gameful scoring algorithm inspired by the longest common subsequence strategy in (Karavirta et al., 2012), target minimal moves needed to solve each puzzle alongside a move counter, and the inclusion of multiple feedback types. In addition to correctness feedback, we enable the customization of messages and actions triggered when certain puzzle solution conditions are (or are not) satisfied. This facilitates the design of intermediary objectives within puzzles, similar to subgoal labeling shown to help students solve PPPs (Morrison et al., 2016). The objectives constructed operate as test cases, like the execution-based approach described in (Helminen et al., 2013) and the auto-grader methodology in (Haldeman et al., 2018), though we perform static rather than dynamic analysis as the learner positions each construct, and provide dynamic

execution feedback concurrently via auto-execution of gameful animations in the Scratch stage. Using the feedback classification documented in (Raubenheimer et al., 2021), we view this feedback as constructivist because it is problem- and instance-oriented, which is a category of assistance associated with significantly lower student failure rates than alternatives, like solution- and theory-oriented.

We fortify the GBL features with those enabling the instructor to vary scaffolding, similar to the text/example/bug modulation in the BOTS study in (Zhi et al., 2018). Like text-based PPPs, the instructor can assemble constructs in an isolated palette for use during puzzle play. Alternatively, s/he can design a PPP requiring navigation through existing Scratch palettes (e.g. Control, Operators). Since the optionality might overwhelm, in a manner historically similar to sp/k (Holt et al., 1977) and more recently like (Cazzola et al., 2015; Rose et al., 2018), the instructor can select which blocks to enable in each palette, and further, which palettes to enable. This configurability facilitates the design of progressions in which learners encounter an isolated palette before transitioning to increasing numbers of palettes with increasing numbers of blocks enabled. This scaffolding fading can occur in the feedback dimension separately or simultaneously, as the instructor can also vary the correctness and objective feedback activation. As in (Dicheva et al., 2018), in which PPPs are used in exercises 1-2 and code writing is used in 3-4, the scaffolding variability allows for the introduction of CT concepts with constraints before relaxing those as learning proceeds to afford learners increasing agency.

Since the PPPs are in Scratch, learners who develop cognitive CT mastery as scaffolding fades become familiarized with the UI, enabling them to advance to epistemologically pluralistic elements of the curriculum in which they can use CT for open-ended creation (Turkle et al., 1990). Results from a related previous study indicated that the constraint of an isolated palette can increase learning efficiency and motivation, and the inclusion of correctness feedback can increase motivation (Bender et al., 2022). In the current study, we introduce new instructional design and feedback capabilities, and explore how CL, performance, learning efficiency, and motivation are affected by scaffolding variation.

3. Software Development

To investigate **R1-R3**, we developed an instructional design module using Blockly (Google for Education, 2018), and integrated the functionality with earlier modifications to Scratch that enable the design, play, and assessment of PPPs. While previous work introduced block points, scoring, progress bars, configurable blocks and palettes, and correctness feedback, the learning system could not fade feedback. Though feedback increased motivation in earlier studies, some participants noted anxiety, stress, worry, and anger when informed of mistakes move-by-move (Bender et al., 2022). To bridge the gap between correctness feedback and none except the animation, we defined a novel grammar and an objective editor for instructors to devise objectives with associated feedback messages and actions for each puzzle. Grammar details are available in this article's supplement site: https://bit.ly/3wdCowW.

Since teachers learn CT as they infuse it across subjects (Committee for the Workshops on CT, 2010), we designed the objective editor to be simple to use and include it in the concrete architecture in (Sulaiman et al., 2019) as a module exposed only to instructors. We frame it as a tool situated within teachers' practice intended to promote teacher learning, in the spirit of educative curriculum materials (Ball et al., 1996). The teacher should use pedagogical content knowledge to design useful objectives, such as the timely identification of repeating sequences and loop sentinels.

For example, consider a case in which a teacher introduces the CT concept *looping* to her students, after they have mastered *sequences*. If the puzzle designed requires a sprite to move in a path of a square, a student might begin by selecting a move-steps block, with the intention to later include three additional move-steps blocks, and four turn-left blocks, as shown in Figure 1a. However, the teacher can configure the puzzle to exclude the move-steps block initially, and use the objective editor to author guidance for the student to identify a repeating pattern s/he could embed in a control construct. Figure 1b presents an objective that might instead lead to the solution depicted in Figure 1c. First, the objective in 1b specifies that until the student uses the repeat block, she will receive feedback guiding her toward pattern identification; second, it records the feedback she will receive once she uses the repeat block; third, it designates three actions that will occur once the repeat is used: the move-steps and turn-left blocks will become enabled, and five points will be added to the student's objective score. An example of objective feedback during puzzle play in this study is show in the top-right of Figure 1d.

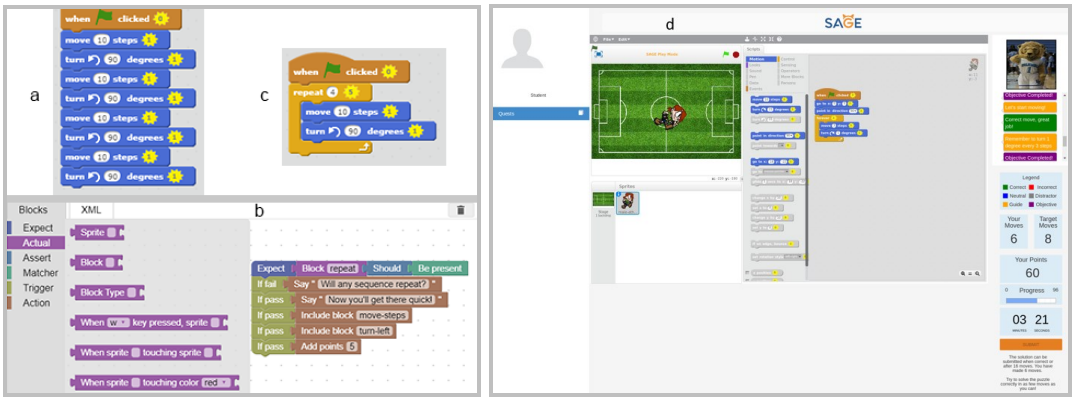

Figure 1. Examples of objective configuration and puzzle play experience.

4. Study Purpose

The software developed prepared us to investigate scaffolding variation. One study purpose was to identify which scaffolding approaches produce the lowest CL and most efficient CT learning. Given the limited skill increases over time and misconceptions developed by Scratchers discussed in section 1, the argument regarding the difficulties with minimally guided instruction in (Kirschner et al., 2006) appears applicable to CT learning in Scratch. By augmenting that learning experience with presentation and feedback modifications and comparing the outcomes, we aimed to contribute insight into the types of guidance that might help accelerate the proliferation of CT learning for the general populace.

A second study purpose was to explore the motivational impact of CT learning. In an era in which negative perceptions of CS persist (Schulte et al., 2005), self-directed learning in older adults is rising (Morrison et al., 2019), and deficits in adult CS knowledge limit parental involvement in children's CT learning (Bresnihan et al., 2019), insight into the attitudinal change provoked by a short CT learning intervention could inform future outreach. (Wang et al., 2015) emphasizes the crucial role family plays in encouraging women to participate in computing, (Ohland et al., 2019) offers arguments for educating parents on how to support CT learning, and (Sepúlveda-Díaz et al., 2020) offers guidance for supporting parents homeschooling their children in CT. We extend this work by examining whether the strategy of fading scaffolding leads to higher CT motivation. While a common approach, high-to-low scaffolding has not always yielded optimal outcomes, particularly in research on preparation for future learning (Chin et al., 2010). For example, (Blickstein et. al., 2018) found that open-ended instruction followed by detailed guidance led to higher learner activation in subsequent activities than the converse, indicating a carryover effect referred to as epistemological persistence.

We include 11 conditions on the CT concept *looping*, plus one concept control on *sequences*, and analyze the variation of presentation and feedback in 7 puzzles, each time-boxed for 8 minutes, that connect via a narrative with explicit goals related to playful animations. Based on the results in section 2, our hypotheses associated with **R1-R3** were: **H1**) PPPs with feedback yield lowest CL; **H2**) an isolated palette yields highest learning efficiency; **H3**) fading scaffolding yields highest CT motivation.

5. Summative Evaluation

5.1 Study Design

To test these hypotheses, we developed a quantitative experiment between-subjects, with some within-subject measurements. A study of perceived preferences for game elements in learning environments (Denden et al., 2018), as well as the review in (Santos et al., 2018) of the use of game elements for learning programming, inspired us to conduct a similar review of PPP elements in the PPP literature. However, we found limited research distinguishing which PPP elements prove most useful for learning (c.f. Ericson et.al, 2018, Bender et al., 2022). Consequently, we devised 12 study conditions detailed in Table 1 to ensure we could granularly analyze the effect of various PPP elements on learning outcomes

and help fill this gap in the literature. As documented in Table 2, participants followed a 10-step protocol which in part required them to respond to a validated: 1) CS CL component survey (CS CLCS) (Morrison et al., 2014); 2) intrinsic motivation Task Evaluation Questionnaire (TEQ) (SDT, 2022); 3) Computing Attitudes Survey (CAS) (Bockmon et al., 2020) (details at https://bit.ly/3wdCowW).

Table 1. *PPP scaffolding variation across 12 study conditions*

Cond.	CT Concept	Presentation	Feedback	Fading Scaffolding
C1	Looping	1-palette	Correctness	No
C2	Looping	Multi-palette	Correctness	No
C3	Looping	1-palette	Correctness + Objectives	No
C4	Looping	Multi-palette	Correctness + Objectives	No
C5	Looping	1-palette	Objectives	No
C6	Looping	Multi-palette	Objectives	No
C7	Looping	1-palette	3 Correctness, 2 Correctness + Objectives, 2 Objectives	Correctness feedback faded
C8	Looping	Multi-palette	3 Correctness, 2 Correctness + Objectives, 2 Objectives	Correctness feedback faded
C9	Looping	4: 1-palette, 3: multi-palette	3 Correctness, 2 Correctness + Objectives, 2 Objectives	Correctness feedback & 1-palette faded
C10	Looping	Multi-palette	None	No
C11	Looping	Multi-palette + distractors	None	No
C12	Sequences	1-palette	Correctness	No

Table 2. *Study protocol and data collected*

Step	Activity	Content	Data Collected
1	Registration	Credentials creation & condition assignment	Username & password
2	Background info	Demographic	Demographic, programming attitude, CAS, CT perceptions
3	Tutorial	8-minute video on the learning system & *looping*	N/A
4	Pretest (isomorphic)	7 multiple-choice (4-choice) *looping* questions	Pretest responses & score
5	CS CLCS	10 CL questions with 0-10 scaled responses	Pretest CL & IL/EL/GL components
6	Puzzles	7 puzzles on *looping* in 11 of 12 conditions; learning system behavior varies by condition; 7 puzzles on *sequences* in the 12th condition (control)	Per-puzzle time spent, time-stamped block moves & score, correctness, feedback log, self-explanations
7	CS CLCS	10 CL questions with 0-10 scaled responses	Puzzle CL & IL/EL/GL components
8	Posttest (isomorphic)	7 multiple-choice (4-choice) *looping* questions	Posttest responses & score
9	CS CLCS	10 CL questions with 0-10 scaled responses	Posttest CL & IL/EL/GL components
10	Concluding measurements	Motivation, programming attitude, learning system feedback, CT perceptions	TEQ, CAS, & programming attitude, CT perceptions

5.2 Participants

Using Prolific (Prolific, n.d.) we recruited 579 participants with varying degrees (57% high school, 27% undergraduate, 16% graduate), and a variety of self-reported programming experience (low: 50%; medium: 36%; high: 14%). 405 men, 167 women, and 2 non-binaries comprise the population sourced from 28 countries led by Poland (21%), Portugal (14%), and the U.K. (14%). Since the software development involved Scratch 2.0, which depends on Flash, a technology sunset at the start of 2021, learners needed to download a virtual machine we equipped to bypass Flash disablement. While necessary, this requirement introduced a risk to the external validity of the sample general population, as participants included only those sufficiently capable of installing virtualization software.

5.3 Data Collection & Processing

We created 7 surveys and instrumented the learning system to: 1) record puzzle play duration; 2) trace each block moved; 3) calculate score using an algorithm detailed in (Bender et al., 2021) that results in higher scores as construction nears the solution. To quantify the effect on CT learning, we also recorded self-reported CL. For learning to occur effectively, the CL of complex tasks should be reduced, though

the reduction need not occur in all three dimensions of CL (Sweller, 2010). The total number of interacting elements perceived by the learner determines intrinsic load (IL); the sometimes-impeding presentation of the content determines extraneous load (EL); and the instructional features necessary for schema construction determine germane load (GL). For example, conditions with an isolated palette and with blocks that become enabled incrementally as puzzle play unfolds aim to reduce EL otherwise required for interface navigation and block search to free learners' capacity to contend with GL.

To account for learners who compensate for an increase in CL by committing more mental effort, resulting in constant performance while load varies, we calculate instructional and performance efficiency (IE: learning process, PE: learning outcome) (van Gog et al., 2008). Previous studies have found lower PE for PPPs with randomly distributed distractor blocks compared to PPPs (Harms et al., 2016); higher IE for PPPs than for writing code (Bender et al., 2021); and higher IE for PPPs with 1 palette compared with multiple with distractors (Bender et al., 2022). We calculated IE and PE using both time and CL during training and transfer tasks. Since the data did not exhibit Shapiro-Wilk normality ($p<.05$), we used non-parametric statistics, like Kruskal-Wallis H and Mann-Whitney U between-subjects, and Wilcoxon within-subjects. For effect sizes, we used values in (Fritz et al., 2012).

6. Analysis & Results

6.1 Cognitive Load

We did not find significant differences between conditions in self-reported CL during the pre-test. Likewise, the posttest yielded no significant CL differences, suggestive of the acquisition of cognitive structures of equivalent expertise (van Gog et al., 2008). However, we did find significant differences between training conditions with moderate effect for GL ($H(11)=26.08$, $p=.006$, $\epsilon^2=.05$). Using a Bonferroni-adjusted alpha of .004 (.05/12), significant difference remained ($p=.019$) between conditions C4 ($M=4.90$) vs. C11 ($M=5.27$). This indicates participants training with multiple palettes and both correctness and objective feedback required less mental effort to contend with instructional features necessary for schema construction than those who received no feedback while navigating multiple palettes that included distractor blocks not part of the puzzle solution. To isolate the mediating variables, we analyzed treatment condition sets grouped by scaffolding variation and found a significant difference in GL with small effect between sets of conditions that did or did not receive objective feedback ($U(N_{objectives}=359, N_{no\text{-}objectives}=200)=29{,}306.5$, $z=-3.60$, $p<.001$, $r=.15$, $M_{objectives}=4.78$, $M_{no\text{-}objectives}=5.66$). We also found with small effect significant (GL) and moderate (overall CL) differences between those who received any feedback and those who did not ($U(N_{feedback}=461, N_{no\text{-}feedback}=98)=18{,}537.5$, $z=-2.79$, $p=.005$, $r=.12$, $M_{feedback}=4.95$, $M_{no\text{-}feedback}=5.78$; $U(N_{feedback}=461, N_{no\text{-}feedback}=98)=19{,}860.0$, $z=-1.88$, $p=.060$, $r=.08$, $M_{feedback}=4.65$, $M_{no\text{-}feedback}=4.97$). These results offer evidence that training with feedback limits the GL and CL experienced by the learner, supportive of **H1**.

6.2 Performance

During the transfer phase, the treatment population solved significantly more posttest than pretest questions correctly with small effect ($z=3.4$, $p<.001$, $r=.14$, $M_{pretest}=6.0$, $M_{posttest}=6.25$). Like in (Harms et al., 2016; Ericson et al., 2018; Bender et al. 2021, 2022), we did not find transfer performance disparity between PPP conditions. During training, however, we found significant differences between conditions in the time spent solving with relatively strong effect ($H(11)=39.29$, $p<.001$, $\epsilon^2=.20$), and with moderate effect, block moves made ($H(11)=113.8$, $p<.001$, $\epsilon^2=.07$) and puzzles solved ($H(11)=74.46$, $p<.001$, $\epsilon^2=.13$). Notable time-saving conditions, both with an isolated palette and feedback, are C1 ($M=21.6$) and C7 ($M=20.8$), which required significantly less time than C10 ($M=26.7$, C7 vs C10: $p=.033$) and C11 ($M=29.0$, C1 vs C11: $p<.001$) with multiple palettes and no feedback. Examining conditions sets led to significant differences with small effect between sets with an isolated palette and multiple palettes ($U(N_{1\text{-}palette}=211, N_{multi\text{-}palette}=348)=28{,}926$, $z=-4.21$, $p<.001$, $r=.18$, $M_{1\text{-}palette}=23.0$, $M_{multi\text{-}palette}=2.2$), with and without correctness feedback ($U(N_{correctness}=359, N_{no\text{-}correctness}=200)=29{,}188$, $z=-3.67$, $p<.001$, $r=.16$, $M_{correctness}=24.0$, $M_{no\text{-}correctness}=26.7$), and with and without fading scaffolding ($U(N_{fading}=155, N_{no\text{-}fading}=404)=27{,}032$, $z=-2.50$, $p=.012$, $r=.11$, $M_{fading}=23.5$, $M_{no\text{-}fading}=25.5$). This

indicates an isolated palette, correctness feedback, and fading scaffolding lead to less training time.

Condition sets with the lowest number of block moves were those with an isolated palette and with objective feedback ($U(N_{1\text{-palette}}=211, N_{\text{multi-palette}}=348)=31{,}653$, $z=-2.73$, $p=.006$, $r=.12$, $M_{1\text{-palette}}=78$, $M_{\text{multi-palette}}=84$; $U(N_{\text{objectives}}=359, N_{\text{no-obectives}}=200)=20{,}825$, $z=-8.24$, $p<.001$, $r=.35$, $M_{\text{objectives}}=74$, $M_{\text{no-objectives}}=95$). Those without objectives solved significantly more puzzles correctly with small effect, as did those with correctness feedback and fading scaffolding ($U(N_{\text{objectives}}=359, N_{\text{no-obectives}}=200)=28{,}819$, $z=-3.91$, $p<.001$, $r=.17$, $M_{\text{objectives}}=2.6$, $M_{\text{no-objectives}}=3.3$; $U(N_{\text{correctness}}=359, N_{\text{no-correctness}}=200)=42{,}220$, $z=3.49$, $p<.001$, $r=.15$, $M_{\text{correctness}}=3.1$, $M_{\text{no-correctness}}=2.5$; $U(N_{\text{fading}}=155, N_{\text{no-fading}}=404)=37{,}576$, $z=3.70$, $p<.001$, $r=.16$, $M_{\text{fading}}=3.4$, $M_{\text{no-fading}}=2.7$). These results indicate that objective feedback leads to the fewest puzzles solved and correctness feedback and fading scaffolding lead to the most.

6.3 Efficiency

While we did not find significant PE differences between treatment conditions, when using time spent as an estimate of mental effort, we found significant IE differences with moderate effect ($H(10)=27.73$, $p=.002$, $\epsilon^2=.05$), with condition pairs C1 vs. C11 ($p=.034$) and C7 vs. C11 ($p=.002$) remaining significant after Bonferroni adjustment. This indicates that training with an isolated palette with correctness feedback ($M_{C1}=.29$) and more so with an isolated palette with correctness feedback fading ($M_{C7}=.46$) leads to more efficient CT learning than with multiple palettes without feedback and with distractors ($M_{C11}=-.34$). When grouping condition sets by number of palettes, inclusion of correctness feedback, and fading scaffolding, we identified with a small effect significant differences for the first two and a moderate difference for the third ($U(N_{1\text{-palette}}=211, N_{\text{multi-palette}}=348)=41{,}474.0$, $z=2.57$, $p=.010$, $r=.11$, $M_{1\text{-palette}}=.15$, $M_{\text{multi-palette}}=.09$; $U(N_{\text{correctness}}=359, N_{\text{no-correctness}}=200)=40{,}025.0$, $z=2.25$, $p=.024$, $r=.10$, $M_{\text{correctness}}=.06$, $M_{\text{no-correctness}}=.11$; $U(N_{\text{fading}}=155, N_{\text{no-fading}}=404)=34{,}279.0$, $z=1.74$, $p=.082$, $r=06$, $M_{\text{fading}}=.11$, $M_{\text{no-fading}}=.04$). These results support **H2** and reveal additional learning efficiency opportunities via the use and fading of correctness feedback in addition to an isolated palette.

6.4 Motivation

6.4.1 Quantitative Results

To analyze motivation quantitatively, we scored the TEQ, and calculated the within-subject change in CAS scores, programming attitude, and CT perception, from study start to end. Condition sets with significant findings with small effect included those that varied by objective feedback for the interest/enjoyment subscale ($U(N_{\text{objectives}}=359, N_{\text{no-objectives}}=200)=94{,}948.0$, $z=-3.13$, $p=.002$, $r=.13$, $M_{\text{objectives}}=3.73$, $M_{\text{no-objectives}}=4.2$) and by objective and correctness feedback for the perceived competence subscale ($U(N_{\text{objectives}}=359, N_{\text{no-objectives}}=200)=32{,}801.5$, $z=-1.79$, $p=.074$, $r=.08$, $M_{\text{objectives}}=2.80$, $M_{\text{no-objectives}}=3.15$; $U(N_{\text{correctness}}=359, N_{\text{correctness}}=200)=40{,}229.0$, $z=2.30$, $p=0.21$, $r=.10$, $M_{\text{correctness}}=3.06$, $M_{\text{no-correctness}}=2.70$). Though narrowly missing significance, the objective feedback condition sets also showed noteworthy differences in the pressure/tension subscale ($U(N_{\text{objectives}}=359, N_{\text{no-objectives}}=200)=39{,}638.5$, $z=1.94$, $p=.052$, $r=.08$, $M_{\text{objectives}}=3.74$, $M_{\text{no-objectives}}=3.46$). These results indicate correctness feedback increases perceived competence, while objective feedback reduces interest/enjoyment, lowers perceived competence, and increases pressure/tension.

From the CAS scoring, we found significant within-subject change for the factors of: 3) importance, 4) problem solving – strategies, and 6) personal interest for the treatment population (3: $z=-5.3$, $p<.001$, $r=.22$, $M_{\text{start}}=3.94$, $M_{\text{end}}=3.76$; 4: $z=5.62$, $p<.001$, $r=.24$, $M_{\text{start}}=3.44$, $M_{\text{end}}=3.56$; 6: $z=4.28$, $p<.001$, $r=.18$, $M_{\text{start}}=3.42$, $M_{\text{end}}=3.61$). This decrease in importance and increase in problem solving strategies and personal interest largely held across conditions. For the gender equity (2) and gender bias factors (5), however, only the conditions with fading scaffolding yielded significant differences (2: $z=2.35$, $p=.019$, $r=.10$, $M_{\text{start}}=4.47$, $M_{\text{end}}=4.53$; 5: $z=-1.97$, $p=.049$, $r=.08.$, $M_{\text{start}}=2.01$, $M_{\text{end}}= 1.93$), indicating an attitudinal improvement in gender equity and reduction in gender bias.

Additionally, we found many significant programming attitude and CT perception improvements from study start to end per condition set; selected results are presented in Table 3. Overall, the quantitative motivation results provide partial support for **H3**, as fading scaffolding led to improved programming attitude and CT perception scores, but the lack of longitudinal data represents a threat to internal validity, since we cannot confirm the change measured at study conclusion persists.

Table 3. *Within-subject attitude and CT perception changes. *p<0.05, **p<0.01*

-- Progrmaming is… --	1-palette	Correctness Feedback	Fading Scaffolding
fun	M=.71**	M=.65**	M=.64**
enjoyable	M=.82**	M=.69**	M=.53*
easy to start	M=.94**	M=.67**	M=.73**
-- CT Perception --	--	--	--
I would recommend children in my family attend a CT camp or after-school program	no significant change	M=.35**	M=.42*
I would ask an employer for CT training	M=.31*	M=.39**	M=.71**

6.4.2 Qualitative Results

We studied motivation qualitatively by requiring that participants describe their attitude toward programming at study-end, as well as their perspectives on presentation and feedback scaffolding. Those with low self-reported prior programming experience often reflected surprise: 1) "It is perhaps easier than I thought once you get to grips with how it works and the different ideas and ways of doing it, and it is more accessible to the average person than I thought."; 2) "Starting it is much easier than I thought… it can be more fun after this practice I feel more motivated to find some time and finally give it a proper try." Those with higher prior experience advised on use cases: "this format is good for younger people to learn and get introduced to it. I think is needed to be a class in school about it."

Those who received correctness feedback reported the utility: "It helped me know when I was straying from course and re-evaluate my decisions. Very important and useful." Those receiving only objective feedback also perceived value: "The feedback was very motivating for me and each "correct answer" was giving me a lot of joy. I found this very helpful to know if I was heading in the right direction or missing out anything important, and the dynamically enabled blocks helped me make sure I was doing things in the right order." A C1 participant, with an isolated palette and correctness feedback, perceived challenge decreasing as s/he solved puzzles: "At the beginning of the task, I was preoccupied with how to navigate the system, but when I was familiar… I could easily solve the looping puzzles."

Many responses support **H3**, as they indicate fading scaffolding provides motivating challenge across puzzles, while increasing agency. Several participants in C9, in which both presentation and feedback scaffolding faded, reflected inspiration: 1) "When you start by having a lot of support and then it gets reduced, you need to rely on your skills and understanding to keep learning, and when you see that the program is working fine, it is more rewarding."; 2) "the ones with the same palette help at a very early stage to familiarize with the concepts, the ones with a different palette broke this familiarity and helps further by making you think more; 3) "The single palette made it easier to focus on how to organize the blocks, instead of wasting time looking for them. Having them easily accessible at first made it easier to find them and use them in the later tasks when the blocks were spread across multiple palettes." Some with fading feedback, however, noted difficulties: "I found the lack of feedback later on really hard to overcome. It helped me immensely at the beginning. Even though I was more familiar with the concepts and blocks etc, the complexity of puzzles was also increasing so I really struggled." One participant in C8, with multiple palettes and feedback fading, recommended offering learner-configurability: "Create a settings where you can turn on or turn off the feedback for the puzzle. This way, if the student wants to try solve the puzzle without help, they can turn off the feedback. However, if they find it too difficult and needs help, then they can turn on the feedback for guidance."

7. Conclusion & Future Work

To illuminate how learning system elements and progressions affect CT learning in block-based environments, we developed Blockly-scaffolded, configurable PPP functionality within Scratch that enables variation of presentation and feedback types. In a between- and within-subjects study of 579 adults, we enumerated several features and scaffolding strategies suitable for a general populace. PPPs with feedback yield lowest CL; an isolated palette, correctness feedback, and fading correctness feedback results in the highest learning efficiency; and fading scaffolding can increase CT motivation. The analysis offers PPP developers and instructors insight to advance efficient CT education for all.

While these results expose opportunities to advance CT learning via augmentations to popular block-based environments, we caution that developing effective technical interfaces between these systems is non-trivial given current implementation stratification. To echo and apply the call for standardization to advance progress in learning engineering in (Baker et al., 2022), we advocate for CT education data standards supportive of scaling consistent, quality data capture using extendable data models facilitating customization but rooted in commonality. Such anchoring could accelerate efforts to extend functionality in future work, for example by simplifying the way the Blockly objective editor could be integrated with the Scratch PPP evaluation engine to drive execution-based feedback like the js-parsons method described in (Helminen et al., 2013). It could also lead to data-driven generation of concept inventories and misconception maps, an approach proposed by EvoParsons developers in (Bari et al., 2019), that could better equip educators with the operational learning definitions necessary to infuse CT across curricula. With continued study, we aim to identify paths toward reliably efficient, effective, and equitable CT learning that builds bridges between cognitive, situated, and critical CT.

Acknowledgements

The Programming Systems Lab is supported in part by DARPA N6600121C4018, NSF CCF-1815494 and NSF CNS-1563555.

References

Anderson, J. R. (1982). Acquisition of cognitive skill. *Psychological Review*.

Baker, R. S., Boser, U., Snow, E., & McNamara, D. (2022). Learning engineering. *Technology, Mind, Behavior*.

Ball, D., & Cohen, D. (1996). Reform by the book? *Educational Researcher, 25*(9), 6-14.

Bari, A., Gaspar, A., Wiegand, R., Albert, J., Bucci, A., & Kumar, A. (2019). EvoParsons. *GPEM*, 213-244.

Bender, J., Zhao, B., Dziena, A., & Kaiser, G. (2022). Learning computational thinking efficiently. *ACE*.

Bender, J., Zhao, B., Madduri, L., Dziena, A., Liebeskind, A., & Kaiser, G. (2021). Integrating Parsons Puzzles with Scratch. *ICCE.*

Black Girls Code. (n.d.). *Women of color in technology*. Retrieved from https://www.blackgirlscode.com/

Blickstein, P. (2018). *Pre-college computer science education.* Retrieved from https://goo.gl/gmS1Vm

Blickstein, P., & Hejazi, M. S. (2019). Computing education. In *Computing Education Research* (pp. 56-78).

Blikstein, P., Gomes, J., Akiba, H., & Schneider, B. (2017). The effect of highly scaffolded versus general instruction on students' exploratory behavior and arousal. *TKL*, 105-128.

Bockmon, R., Cooper, S., Gratch, J., & Dorodchi, M. (2020). Validating a CS attitudes instrument. *SIGCSE*.

Brennan, K. (2013). *Best of both worlds: issues of structure and agency in computational creation.* MIT.

Bresnihan, N., Srong, G., Fisher, L., & Lynch, Á. (2019). Increasing parental involvement in CS education. *CSE*.

Brusilovsky, P., Malmi, L., Hosseini, R., Guerra, J., Sirkiä, T., & Pollari-Malmi, K. (2018). An integrated practice system for learning programming in Python: design and evaluation. *RPTEL, 13*(1), 1-40.

Buffum, P. S., Lobene, E. V., Frankosky, M. H., Wiebe, E. N., & Lester, J. C. (2015). A practical guide to developing and validating CS knowledge assessments. *SIGCSE*, (pp. 622-627).

Campos, A., Rodrigues, M., Signoretti, A., & Amorim, M. (2017). piBook. *CSE*, (pp. 179-195).

Cao, L., Rorrer, A., Pugalee, D., Maher, M. L., Dorodchi, M., Frye, D., Barnes, T., Wiebe, E. (2020). Work in progress report: a STEM ecosystem approach to CS/CT. *ACM SIGCSE*, (pp. 999-1004).

Cazzola, W., & Olivares, D. M. (2015). Gradually learning programming supported by a growable programming language. *IEEE Transactions on Emerging Topics in Computing*, 404-415.

Celepkolu, M., O'Halloran, E., & Boyer, K. (2020). Upper elementary middle grade teachers' perceptions, concerns, and goals for integrating CS into classrooms. *SIGCSE*, (pp. 965-970).

Charters, P., Lee, M., Ko, A., & Loksa, D. (2014). Challenging stereotypes and changing attitudes. *SIGCSE*.

Chin, D., Dohmen, I., Cheng, B., Oppezzo, M. C., & Schwartzz, D. (2010). Preparing students for future learning with teachable agents. *Educational Technology Research and Development*, 649-669.

Code.org. (n.d.). *Hour of Code*. Retrieved from https://hourofcode.com/

Committee for the Workshops on Computational Thinking. (2010). *Report of a workshop on the scope and nature of computational thinking.* National Academies Press, Washington, D.C.

Denden, M., Tlili, A., Essalmi, F., & Jemni, M. (2018). Does personality affect students' perceived preferences for game elements in gamified learning environments? *Advanced Learning Technologies*, (pp. 111-115).

Dicheva, D., & Hodge, A. (2018). *Active learning through game play in a data structures course.* SIGCSE.

Dickey, M. D. (2006). "Ninja looting" for instructional design. *SIGGRAPH.*

Du, Y., Luxton-Reilly, A., & Denny, P. (2020). A review of research on parsons problems. *ACE*, (pp. 195-202).

Ericson, B., & Rick, J. (2018). Evaluating the efficiency and effectiveness of adaptive parsons problems. *ICER*.

Franklin, D., Salac, J., Thomas, C., & Krause, S. (2020). Eliciting student Scratch script understandings. *SIGCSE*.
Fritz, C. O., Morris, P. E., & Richler, J. J. (2012). Effect size estimates. *Experimental Psychology, 141*(1), 2-18.
Garner, J., Denny, P., & Luxton-Reilly, A. (2019). Mastery learning in CS education. *ACE*, (pp. 37-46).
Girls Who Code. (n.d.). *Girls Who Code*. Retrieved from https://girlswhocode.com/
Google for Education. (2018). *Bockly*. Retrieved from https://developers.google.com/blockly/
Grover, S., & Basu, S. (2017). Measuring student learning in introductory block-based programming. *SIGCSE*.
Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a CS course for MS students. *CSE*.
Haldeman, G., Tjang, A., Babes-Vroman, M., Bartos, S., Shah, J., Yucht, D., & Nguyen, T. (2018). Providing meaningful feedback for autograding of programming assignments. *SIGCSE*, (pp. 278-283).
Half-Baked Software. (n.d.). *Hot Potatoes*. Retrieved from http://hotpot.uvic.ca/
Hamilton-smith, L. (2016). *Learning Curve: Coding classes to become mandatory in Queensland schools*. https://www.abc.net.au/news/2016-11-17/coding-classes-in-queensland-schools-mandatory-from-2017/
Harms, K., Chen, J., & Kelleher, C. (2016). Distractors in parsons problems decrease learning efficiency for young novice programmers. *International Computing Education Research*, (pp. 241-250).
Harms, K., Rowlett, N., & Kelleher, C. (2015). Enabling independent learning of programming concepts through programming completion puzzles. *Visual Lanugages and Human-Centric Computing*, (pp. 271-279).
Helminen, J., Ihantola, P., Karavirta, V., & Alaoutinen, S. (2013). How do students solve parsons programming problems? *Learning and Teaching in Computing and Engineering*, (pp. 55-61).
Holt, R. C., Wortman, D. B., Barnard, D. T., & Cordy, J. R. (1977). SP/k. *Comms. of the ACM , 20*(5), 301-309.
Ihantola, P., & Karavirta, V. (2010). Open source widget for parson's puzzles. *Innovation and Technology in CSE.*
Karavirta, V., Helminen, J., & Ihantola, P. (2012). A mobile learning application for parsons problems. *CER*.
Kim, A. J. (2018). *Game thinking.* gamethinking.io.
Kim, T., & Axelrod, S. (2005). Direct instruction. *The Behavior Analyst Today, 6*(2), 111-120.
Kirschner, P., Sweller, J., & Clark, R. (2006). Why minimal guidance during instruction does not work. *Educational Psychologist, 41*(2), 75-86.
MIT Media Lab. (2022). Retrieved from Scratch Statistics: https://scratch.mit.edu/statistics/
Morrison, B., Dorn, B., & Guzdial, M. (2014). Measuring cognitive load in introductory CS. *ICER*, (pp. 131-138).
Morrison, B., Ericson, B., & Guzdial, M. (2016). Subgoals help students solve Parsons problems. *SIGCSE*.
Morrison, D., & McCutheon, J. (2019). Empowering older adults' informal, self-directed learning. *RPTEL*, 1-16.
Ohland, C., Ehsan, H., & Cardella, M. E. (2019). Parental influence on children's CT in an informal setting. *ASEE.*
Parsons, D., & Haden, P. (2006). Parson's programming puzzles. *ACCE.*
Pea, R. D. (1986). Language independent conceptual "bugs" in novice programming. *ECR*, 25-36.
Plattner, H. (2013). *An introduction to design thinking: process guide.* Institute of Design at Stanford.
Pollock, L., Mouza, C., Guidry, K. R., & Pusecker, K. (2019). Infusing CT across disciplines. *SIGCSE*.
Proctor, C., & Blikstein, P. (2018). How broad is computational thinking? *Society of the Learning Sciences.*
Prolific. (n.d.). Online participant recruitment for surveys and market research: https://www.prolific.co/
Raubenheimer, G., Jeffries, B., & Yacef, K. (2021). Pedagogical feedback in programming learning. *ACE*.
Rose, S., Habgood, J., & Jay, T. (2018). *Pirate plunder: game-based CT using scratch blocks.* ACPIL.
Salen, K., & Zimmerman, E. (2003). *Rules of Play: Game Design Fundamentals.* MIT Pess.
Santos, A., Souza, M., Dayrell, M., & Figueiredeo, E. (2018). A systematic mapping study on game elements and serious games for learning programming. *Computer Supported Education*, (pp. 328-356).
Scaffidi, C., & Chambers, C. (2012). Skill progression demonstrated by users in Scratch. *HCI, 28*(6), 383-398.
Schaffhauser, D. (2018). *CS education week kicks off, but som teachers don't feel ready*. https://thejournal.com/articles/cs-education-week-kicks-off-but-some-teachers-dont-feel-ready.aspx
Schulte, C., & Magenheim, J. (2005). Novices' expectations and prior knowledge of soft. dev. *CER*.
SDT. (n.d.). Self-determination Theory: https://selfdeterminationtheory.org/intrinsic-motivation-inventory/
Sepúlveda-Díaz, C., Suardro Rojas, E., Simmonds, J., Guitierrez, F. J., Hitschfeld, N., Casanova, C., & Sotomayor, C. (2020). Lessons learned from introducting preteens in parent-led homeschooling to CT. *SIGCSE*.
Shute, V., Rahimi, S., Smith, G., Ke, F., Almond, R., Dai, C., Kuba, R., Liu, Z.,Yang, X., & Sun, C. (2021). Maximizing learning without sacrificing the fun. *Computer Assisted Learning*, 127-141.
Sulaiman, J., Dziena, A., Bender, J., & Kaiser, G. (2019). SAGE-RA: a reference architecture to advance the teaching and learning of CT. *Embedding AI in Education Policy and Practice for Southeast Asia*.
Sweller, J. (2010). *Cognitive Load Theory: Recent Theoretical Advances.* Cambridge University Press.
Turkle, S., & Papert, S. (1990). Epistemological pluralism. *Women in Culture and Society*, 128-157.
van Gog, T., & Paas, F. (2008). Instructional efficiency. *43*(1), 16-26.
Wang, J., Hong, H., Ravitz, J., & Ivory, M. (2015). Gender differences influencing pursuit of CS. *ITiCSE*.
Weintrop, D., & Wilensky, U. (2017). Between a block and a typeface. *IDC*, (pp. 183-192).
Whitehouse.gov. (2016). *CS4All* https://obamawhitehouse.archives.gov/blog/2016/01/30/computer-science-all
Wing, J. (2006). Computational Thinking. *Communications of the ACM, 49*(3), 33-35.
Winslow, L. E. (1996). Programming pedagogy—a psychological overview. *SIGCSE Bulletin*, 17-22.
Zhi, R., Lytle, N., & Price, T. (2018). Exploring instructional support design for K-12 CE. *ITiCSE*, (pp. 747-752).