Student Experiences with Parsons Problems in a First-Year Engineering Course

Tyler James Stump, The Ohio State University

Tyler Stump is a first year Ph.D. student in the Department of Engineering Education at The Ohio State University. Tyler received his B.S.in Biosystems Engineering at Michigan State University in 2022 and received his M.S. from Michigan State University in 2023. His engineering education interests include first-year engineering student experiences, computing education, and how to foster and develop creativity within programming courses.

Abbey Darya Kashani Motlagh, The Ohio State University
Dr. Krista M Kecskemety, The Ohio State University

Krista Kecskemety is an Associate Professor of Practice in the Department of Engineering Education at The Ohio State University and the Director of the Fundamentals of Engineering for Honors Program. Krista received her B.S. in Aerospace Engineering at The Ohio State University in 2006 and received her M.S. from Ohio State in 2007. In 2012, Krista completed her Ph.D. in Aerospace Engineering at Ohio State. Her engineering education research interests include investigating first-year engineering student experiences, faculty experiences, and the research to practice cycle within first-year engineering.

Student Experiences with Parsons Problems in a First-Year Engineering Course

Abstract

Computational skillsets have become ubiquitous in introductory engineering courses to equip the next generation of engineers to solve modern-day problems during the technological age. Computing Education Researchers focused on improving computing curriculum development, assessment mechanisms, and computational activities to support learning in these contexts. Programmers in the elementary stages of development are challenged with disentangling the dense syntactical thinking prescribed in code creation requiring innovative approaches to pedagogical decisions in aligning activities that proactively mitigate these challenges. One such computational activity, Parsons Problems, is an effective tool to support introductory programmers in mitigating frustration associated with syntactical debugging for students, contributing to attrition in beginner programmers. Parsons Problems are grounded in Cognitive Load Theory to reduce extrinsic cognitive load by separating syntactical thinking from code writing, thus supporting a wide range of computing mastery as well, which is often the landscape in first-year engineering programs. Despite the support that Parsons Problems provides they have not been significantly studied to better understand diverse student experiences when engaging with these computational activities. This study, at a First-Year Engineering Program at The Ohio State University, works to identify the features of these activities that impact the student experience and prioritize these features for optimization by understanding the prevalence of student experiences by feature. Using a thematic codebook, eleven unique features of Parsons Problems were identified that impact student experiences when engaging with them including difficulty, group dynamics, and accessibility. Utility value and group dynamics were found to be the most frequent features to positively impact student experiences, while the difficulty was found to be the most consistent critique suggested further investigation into ways to optimize these learning tools to better support student learning in computing concepts.

Introduction

In the technological age, the need to study and understand computation and the scholarship and teaching employed to prepare the next generation of engineers has become a priority for current education researchers. The National Academies of Sciences, Engineering, and Medicine, reported in a 2018 report by stating, "*It is a time for institutions to consider their missions and constituencies they serve and to determine what role computing should play in the experience, knowledge, and skills of its graduates 2025 and beyond,*" [1]. Computing has been identified as a necessary skillset for engineers entering the workforce to employ computational solutions to complex global issues. Computing educational researchers have embarked on the journey to uncover the evidence-based mechanisms to better support student learning and improve the overall nature of computing courses. As a result of this, computation has been integrated into numerous first-year engineering courses to expose students to introductory computing activities to improve student learning early in their post K-12 career. Introductory programming courses such as these first-year engineering courses have been a significant context to study as the challenges associated with novice programmers have been a focus of scholarly work within computing education research both for the students themselves and the instructors [2,3].

The challenges students face in introductory programming has been a focus for computing education and engineering education researchers investigating the contexts of both introductory computer science courses and introductory programming courses for engineers. In which similar challenges are documented across literature for both contexts. Skills such as problem solving, logical reasoning, and data structures have been identified as such difficulties [4]. Similarly, other common skills seen in multiple studies are the complexities that come with dense syntactical understanding embedded within conceptual understanding that impact a student learning by increasing cognitive load [4-6]. Similarly, instructors are often asked to make pedagogical decisions within introductory programming courses but often lack the methods and tools needed for teaching these contexts [4, pg. 83]. Student engagement and motivation have been identified as critical difficulties educators face when teaching introductory programming. Understanding how to better equip educators with tools that can support the challenges faced by students such as disentangling syntactical thinking from conceptual understanding have created unique computational activities that could be used to mitigate such difficulties for instructors and students.

One such computational activity that has shown promise to be an intervention with the potential to mitigate challenges students and educators face in introductory programming courses is Parsons Problems.

Background – Parsons Problems and Cognitive Load Theory

Parsons Problems (PPs) were first introduced in 2006 by Dale Parson and Patricia Haden from New Zealand in their publication, "Parson's Programming Puzzles: A fun and Effective Learning Tool for First Programming Courses." Parson and Haden describe the unique challenges faced in introductory programming courses as students are often asked to engage with complex coding activities [7]. The first of these challenges being that traditional computational activities were deemed boring by students and often lead to a lack of persistence in completing activities and courses. The second challenge pinpointed was how to isolate and disentangle the complex syntactical thinking inherently embedded within code writing. Thus, this challenge probed the question of how best to separate the complex, language specific, syntax associated with a computational language from the conceptual problem being asked of students in traditional activities. PPs were suggested as a learning tool that acknowledges those challenges and allows for students to begin learning computing without the stressors of learning the syntax simultaneous with, the conceptual problem itself. These computational activities achieved this, treating the computational activity more as an arrangement puzzle in which completed codes were created and then separated line by line either in pieces of paper or online. These allow students to "recreate" someone else's code line by line and consider how the conceptual problem's logical solution pathway could be achieved by the arrangement of the small pieces of code when put back together. The syntax thus is correct in each line but requires students to understand the order it takes to complete a working code. To add to the creation of a puzzle activity approach, Parson and Haden also considered numerous features to consider when designing Parsons Problem such as Distractors, or unnecessary lines included in the code, or incorrect syntax, in order to order to solve the first challenge and add some complexity to make it more align with a puzzle game [7]. Education researchers and practitioners have found many opportunities to integrate and research these Parsons Problems to better support learning in the classroom in a wide range of constructs.

PPs are constructed at the intersection of numerous constructs such as self-regulated learning, metacognition, and abstract problem solving [8,9]. Parsons Problems serve for a potential solution to the complex challenges that exist within introductory computing courses such as having student simultaneous learning syntactical thinking while also engaging with conceptually understanding computational practices. Cognitive Load Theory considers an evolutionary framework to better understand these situations in which students process and store information while maintaining cognitive load for the working memory [10]. Sweller et al. has defined intrinsic load as the inherent interactivity of the elements of information while learning. Now it is understood that improved learning occurs when creating lower intrinsic loads such as long-term member connections [10]. Another form of cognitive load is extrinsic load, primarily created from external stimuli such as pedagogical techniques, which impact student learning. Studies have highlighted that the introductory programming course are environments that call upon a heightened level of intrinsic and extrinsic cognitive load that is cause for concern [11,12]. The level of cognitive load is impacted by three primary components: the difficulty of the material, the way the instruction is given, and strategies used to construct understanding [8]. PPs have positioned themselves with the intentional characteristics to serve as an activity to reduce cognitive load and support such unique challenges that novice programming students face.

Ericson et al. (2022) found 141 papers to synthesize identifying variable spaces interested in improving computing educational approaches. These shared spaces of interest in innovative scholarships shared increased interest within academic communities regarding Parsons Problems as an educational support mechanism for introductory computing classes [8, pg. 11]. This positive trend suggests the interest and need to further investigate these computational programs is continuing to grow as more and more dissemination of these studies provide strategic decision making in first-year engineering programming contexts. The dominant coding language being investigated within these studies was Python, Java, and C/C++/C# suggesting an under exploration of critical languages such as MATLAB and/or R. [8, pg. 15]. The study also had another critical finding the themes being discussed within scholarly articles. Learning programming dominated the research question themes, appearing 134 times of the 141 articles, which makes sense when considering the nature of the activity; however, student perception and student engagement were only found 20.5% (29/141) and 9.9% (14/141), respectively. This finding is interesting as one critical root problem PPs aim to support is to ensure student engagement does not become categorized as “bored” or “boring.” Thus, suggesting more work needing to be done to better understand the student experiences when engaging with PPs and how to maintain student engagement to support practitioner integration of them into the classroom smoothly for educators and students.

This gap in the literature of considering student experiences when engaging with PPs is critical as it highlights the features of PPs to be considered when developing and implementing them into the classroom. As such, this project seeks to understand two primary objectives of understanding the types of features in PPs that impact student experiences and secondly to understand the salience and/or prevalence of these student experiences to prioritize the optimization of the activities by each of feature. Thus, the project works to answer the following research questions: *(1) What are the features of Parsons Problems that impact student’s experiences when engaging with the activity? and (2) How can researchers and educators prioritize these features to better support student learning when operationalizing Parsons Problems?*

Methods

The overall approach to the study was to better understand student experiences when operationalizing these computational activities in a first-year engineering course by leveraging experiences of students when engaging with PPs. The course included an end-of-course survey used to understand student perceptions and experiences for many features and activities of the course including the PPs. Using prior responses from the Autumn 2022 and Autumn 2021 semesters, a qualitative codebook was constructed deductively from student responses, supporting the identification of different types of student experiences previously communicated in prior semesters. A new survey was developed and integrated into the end-of-course survey to include more specific questions for students on their experiences with PPs, specifically targeting barriers and issues students experienced. This updated course survey was then disseminated to the students in the most recent offering of the course (AU23) to elicit more responses from students in the context of these computing activities. The student responses were analyzed inductively with the initial codebook but without rigidness as new codes were hypothesized to occur within this new data sets suggesting that a non-rigid approach would allow unique deductive codes the ability to emerge from the new student responses on experiences. The finalized codebook was constructed and provided an understanding of the types of experiences that exist for students, pinpointing the features serving as a mechanism for these experiences, and ultimately, how prevalent they are.

Course Context

The Fundamentals of Engineering Courses at The Ohio State University are offered to first-year engineering students in the University Honors program. The Fundamentals of Engineering is composed of two courses in subsequent order in which the first course is the focus of this study. The five-credit hour course meets four times a week through three lecture days and one lab-based classroom day. A primary goal of this introductory engineering course is to provide students the opportunity to engage with problem solving, collaboration and computation through computational software's and languages such as Microsoft Excel, MATLAB, and C/C++. The course is not primarily a programming course but includes learning outcomes for the course on students learning how to engage with these computational toolsets. In this course specifically, students engage and learn MATLAB for four weeks of the semester and C/C++ for six weeks of the semester, while the rest of the time is spent on other non-computational foundational skillsets within the curriculum. The teaching team for the program is composed of a lead instructor or faculty member, two Undergraduate Teaching Assistants (UTAs), and 36 students per section.

The course design leverages a flipped classroom model in which pre-class assignments are completed prior to class to allow students an introductory understanding prior to lectures and labs. Students are then asked in class to implement the content through a variety of computational activities. One such activity is the integration of PPs referred to in the classroom as "Weekly Activities". The activities are designed to support collaboration in the classroom by centering the Weekly Activities (WA) as a group activity. In doing so, students are asked to work with one another to solve the puzzles of code to reconstruct the working code in the correct order and this is done both using the MATLAB and C/C++ to support both sets of computational languages. The PPs also appear in the assessments for the classroom with two midterm examination existing, one for MATLAB and one for C/C++, that asks students to solve a PP independently as one component of the assessment. Other assessments exist for these

computational outcomes such as weekly homework, weekly quizzes, and a final Software Design Project – but these examples like the direct integration of Parsons Problem and will not be further focused within this study. As a result of students interacting with these PPs in the WAs and midterms provide numerous interactions between the educational tool and students allowing for undergraduate students taking the course to provide insight into their experiences when engaging with PPs within this first-year engineering course.

Data Collection

Every year for the past several offerings of the course provide students with an end-of-course evaluation survey in which students are asked about their experiences with the WAs/PPs. Initially, the study was designed with the intent to use this dataset with prior collected data to be the focus of analysis within the study. The data was analyzed for Autumn 2022 and 2021 from the course evaluation survey that included numerous topics such as demographic data collection, prior computing experience level, student opinions on features of the course, and the methods of technology used throughout the course. The survey used Likert Style questions and asked students, through five options, to respond to prompts such as the following:

- The Weekly Activities benefited my learning.
- The Weekly Activities were a good use of class time.
- The Weekly Activities were a good difficulty level
- The students at my table usually worked together to complete the Weekly Activities

The Likert scale question provided five options including: strongly disagree to strongly agree with neither agree nor disagree being in the center. The findings of the analysis from these course evaluation surveys from prior semesters were deemed inadequate and misaligned from the data needed to properly answer the research questions at hand. Thus, the team developed and disseminated a second survey to gain access to the student experience data described in the next section.

A second survey, hereafter referred to as the Parsons Problem Student Experience Survey (PPSES), was created to support this pivot in approach due to limitations identified in the initial phase of data collection and analysis. The PPSES looks to identify three pillars: (1) the demographics and identities students align with to establish groups of students with shared racial, ethnic, and gender identities, (2) challenges experienced by students in qualitative form, and (3) asking students to suggest improvements to improve these barriers from their perspective. The first pillar investigates the demographic landscape of the student within the first-year engineering course was critical to the study as assuming all barriers are the same for all students neglects the stressors and challenges that students face when not aligned with the traditional engineering culture that centers heterosexual, cis-gendered, white males' narratives in engineering [13]. These questions leveraged inclusive practices grounded in the recommended format from Hughes et al. (2022) that investigated inclusive survey techniques that were employed for two primary purposes: (1) To showcase a level of intentionality in considering constructs such as gender and race and (2) to provide a set of options for identities that showcase the variable representation possible within these diverse classrooms at a large R1-midwest university [14,15]. The second pillar allows students to describe their experiences from sources of frustrations. The third domain allows for the researchers to elicit another dimension of feedback, more so from a growth mindset in which students are asked to describe and elaborate on how they would

mitigate these difficulties. This allows students to metacognitively center their experiences and consider problem solving these educational activities from an engineering problem solving lens. The PPSES was integrated into the course evaluation survey for the most recent offering of the course, Autumn 2023. Students responded with a 70% rate of the 340 students providing 238 responses providing insight into student experiences with the WAs/PPs in the course.

Data Analysis

The qualitative codebook was constructed for student experiences when engaging with Parsons Problems in which the codes emerged inductively through the analysis of the student response data to the PPSES. The qualitative codebook was inductively constructed by two researchers independently and then calibrated across researchers through Interrater Reliability in reconciling definitions and codes until 100% agreement was achieved. To identify the prevalence of the codes within the dataset, the qualitative codebook was then imposed on the dataset for a second, deductive analysis of the 218 responses. The two researchers again coded the prevalence of each code and their connotation of the code into praise, critique, and neutral. This second iteration of coding was also calibrated using the same reconciliation process and goal of Interrater Reliability as the first iteration. The results of both iterations of analysis are better detailed below in the results section.

Results

Two primary artifacts were created as a result of this study: a qualitative codebook for student experiences when engaging with Parsons Problem as an educational activity and a report of the prevalence of each element impacting student experiences with the activity within this context. Table 1 showcases the first outcome of the study, the qualitative codebook, in which each code includes an operational definition reconciled by the researchers and an example of a student response that would elicit the qualitative code. The table describe eleven unique elements or features being described during the interaction of Parsons Problems and students in the this first-year engineering course that could be transferable to other contexts, such as classrooms that integrate groupwork into Parsons Problems. The eleven features of interest for this transferability consideration includes the following: Accessibility, Assessment, Classroom Dynamics, Difficulty, Distractors, Format, Group Dynamics, Length, Preparation, Time, and Utility Value were all identified as unique elements impacted student experiences. The definitions for each can be described in Table 1.

Table 1: Finalized Codebook for Parsons Problems Features Impacting Student Experiences

Feature	Definitions	Example(s)
Accessibility	Student Experiences regarding access to tools & resources to support the demonstration of a student's knowledge acquisition	*Click and drag, full team seeing activity, etc.*
Assessment	Student Experiences regarding the alignment of the assessment in the course and the Parsons Problem Activities	*"I thought they were good representations of the test problems"*
Classroom Dynamics	Student Experiences regarding the design of the course or features of the course itself that impacted how they engaged with Parsons Problems	*"8:00 am class does not help with focus"*

Difficulty	Student Experiences on the level of difficulty in engaging with Parsons Problems	*"The Parsons Problems were very challenging"*
Distractors	Student Experiences regarding distractors or unnecessary lines of code intentional put in the Parsons Problems	*"The distractors lines were important because they helped me learn what mistake can be hard to find"*
Format	Student Experiences regarding the modality at which the Parsons Problems are engaged with by students	*"They were effective on paper and allowed for teamwork to happen"*
Group Dynamics	Student Experiences regarding collaboration with other students during problem solving for PP.	*"It helped me, and my classmates work together to learn code syntax and order"*
Length	Student Experiences regarding the number of lines within a Parsons Problem	*"Long and Confusing", "too many lines"*
Preparation	Student Experiences regarding a lack of needed skills and/or knowledge to adequately complete the Parsons Problem (i.e. lacking individual understanding needed to solve the problem).	*"Useful when you understand what's being done...kind of pointless if you do not understand the individual pieces as you couldn't understand the overarching problem"*
Time	Student Experiences regarding the amount of time in which a student engages with the activity.	*"They took more time than they were worth in terms of learning"*
Utility value	Student Experiences regarding how much worth or value the Parsons Problems had for their learning	*"Parsons Problems weren't used to code – they did help with problem solving and puzzle logic"*

Finalized Codes

Upon finalizing the qualitative student experience codebook (Table 1), the student responses from the survey were analyzed for a second round of qualitative coding that produced 185 unique emergent codes. These codes were also further analyzed by their connotations, binning the codes into praise, neutral, and critique to capture a more comprehensive understanding of student experiences when engaging with PPs. The results of this embedded coding schema can be seen below in Table 2, with the number of each theme that emerged within each connotation coded for. Of the 185 total codes, 106 were identified as praises, 29 were identified as critiques of features and 10 codes were neutral general comments. The prevalence of each theme within each connotation was completed by taking the number of codes within each theme and dividing it by the total codes for that connotation. Thus, providing the prevalence of each code within each connotation to prioritize what features provide the most positive impacts and negative impacts for student experiences.

Praises were coded when students explained how a feature supported their learning or provided a more positive experience because of it. Table 2 characterizes each element/feature and their prevalence in student responses. The most prevalent features that provided this positive experience were identified as difficulty (12.26%), group dynamics (13.21%), and utility value (58.49%). Students often spoke of the Difficulty of the PPs being challenging and eliciting more engagement. This praise aligns with the original goals of Parson and Haden as the activities are meant to combat boredom [7]. Group Dynamics often were moments in which students spoke on how the opportunity to be collaborative provided the ability to persist or learn more because of the social component provided by PPs. Students described their experience in which the ability to brainstorm and work throughout the process with others provided an overall positive experience when engaging with PPs. Finally Utility Value was frequently described by students in three forms of either being helpful for learning, to improve basic understanding of syntax, and for skill development that were transferable beyond this classroom, such as being able to

understand and manipulate someone else's code to solve a problem. Though students overall had more positive experiences when engaging with PPs, the negative experiences described suggest opportunities for ongoing improvement and further potential research on the activities.

The student experiences categorized as critiques here are those in which students identified a barrier or potential opportunity for improvement within PPs. The most salient of these themes of critiques by students were utility value (13.04%), group dynamics (18.84%), and difficulty (31.88%). Students when describing utility value often saw the activities as providing improved skills in puzzle logic but not in skills that would transfer to "actual code writing" as one student described. Group Dynamics contain examples of what would be described as anti-collaborative or instances in which the activities being collaborative negatively impacted their experience such as the following student's response,

> *"It is a good way to get students of the same table talking and working together to solve a problem. However, I noticed there were always one or two vocal people that took control and did all the work for everyone else."*

In which this student showcased how in some circumstances students took over or "steamrolled" others limiting their engagement with the activity and ultimately negatively impacting their experience with the activities. Lastly, the Difficulty was often described by students as above average difficulty. These responses often were double coded with other themes such as time, length, or accessibility in which other themes ultimately impacted a perceived heightened amount of persistence to complete the activities. One such skill perceived as contributing to the increased difficulty of some students was the ability to read someone else's code. These results provide critical insights in considering PP as computational activities in a multitude of ways described below in the discussion section.

Table 2: Characterization of Parsons Problem Features Impacting Student Experiences

	Praise		Critiques		Neutral	
	n	Prevelence	n	Prevelence	n	Prevelence
Total Codes	106	-	69	-	10	-
Accessibility	4	3.77%	6	8.70%	0	0.0%
Assessment	3	2.83%	1	1.45%	0	0.0%
Classroom Dynamics	0	0.00%	1	1.45%	0	0.0%
Difficulty	13	12.26%	22	31.88%	5	50.0%
Distractors	4	3.77%	1	1.45%	1	10.0%
Format	1	0.94%	4	5.80%	0	0.0%
Group Dynamics	14	13.21%	13	18.84%	3	30.0%
Length	1	0.94%	4	5.80%	0	0.0%
Preparation	3	2.83%	5	7.25%	0	0.0%
Time	1	0.94%	2	2.90%	0	0.0%
Utility Value	62	58.49%	9	13.04%	3	30.0%

Discussion & Implications

The characterization of student experiences when engaging with PP highlights continued opportunities to employ evidence-based practices to better support student learning when

integrating these activities. One such critical finding that can be used to better inform curricular design decisions is understanding the most salient features from the analysis. Utility Value and Group Dynamics in both critiques and praises were identified as critical aspects to consider when implementing PP as they impact student experiences depending on the context in which students engage with the activities. For Group Dynamics, students often spoke to instances in which the activities being collaborative rather than independent provided opportunities for peer-to-peer learning that reinforced a student's individual understanding. It is a well-known pedagogical strategy to employ groupwork or opportunities for collaboration in computing classes and PPs seems to be a sufficient activity to achieve learning while leveraging groupwork. However, a frequent critique from students was that without guardrails or norms to establish each student's role during problem solving often to lead to anti-collaborative peer to peer interactions. One student spoke about their experience in which more experienced students in coding took the activity and completed it independently on behalf of the entire group The student stated, *"There were many instances where one person would take over and do all of the work themselves, leading it to be hard to follow sometimes."* For computing educators, this provides insight into the potential need to explicitly call out and set norms in the classroom when students are engaging with PPs to support an equitable integration of the activity for experienced and inexperienced coders. For computing education researchers, further investigation and consideration into what student roles are present during PP collaborative problem solving and how to best align students' experiences and interest with each role in a way that would better support the power dynamics present.

Utility value has been an important aspect of consideration for student experiences as students are more likely to engage with an activity and learn more if they see transferable value from the activity to their learning journey. Often utility value for students is able to construct computational code with the reduced extrinsic cognitive load of syntactical thinking that is deeply embedded into computing; however, this study highlights how other aspects such as transferable practices across computing languages and the improved skill of reading, understanding, and operationalizing someone else's computational codes have also emerged as new characteristics in which students found heightened utility value in the activity. This finding suggests interesting aspects of PP can be intentionally leveraged to act as a mechanism for deeper engagement from a wide range of students if these skills that students see as valuable were explicitly highlighted. The student experiences highlighted within utility value also have identified skillsets beyond this singular class that shows an intentional alignment with career aspirations which reinforced the findings of other studies [16-18]. Teachers attempting to integrate these activities can also shine a light on transferability of computing skills for example in the introduction to allow students to engage with the activity with intentional skills they can work to develop. The findings of this study pinpoints exact parameters to consider when integrating these activities into introductory computing courses for first year engineers and beyond as a tool that supports reduced cognitive load when conceptualizing understanding of computing topics.

The limitations of this work include the sample populations all attending and utilizing the activities within the same context thus limiting the potential variable student experiences that may not be represented within this work. Another limitation in the study can be found within the survey design. Initially, the project took a deficit framing and developed the survey instrument to contain questions related to barriers rather than student experiences. In doing this, results may be skewed more towards sharing frustrations or negatively framed experiences in replacement of

authentic positive experiences that may not have been elicited provided the question framing. Lastly, the students were asked to reflect on experiences at the end of the course, in which the experience reflected in a student's response may not be representative of their authentic as time and other experiences may have skewed memory of experiences in the moment.

Conclusion and Future Work

Investigating student experiences to better understand how to strategically improve Parsons Problems as a pedagogical intervention served as the guiding purpose for the study. Identifying eleven features of PPs impacting student's experiences provides considerations for educators in introductory programming courses to specifically consider, as well as increased understanding in how the prevalence of those experiences can help guide activity development. The study itself provided further investigation of the integration of Parsons Problems into the class. In future studies, researchers should aim to understand how the prevalence of these identified experiences shift when looking at identities of student or through an intersectional identity lens to know if disproportional critiques and praises exist for certain groups of students. In a similar way, further investigation in optimizing the learning tool could be made by examining how each of the critical features impacting student experience interplay with one another to impact student learning. The findings with group dynamics also suggest further understanding of the complex social interactions that may occur when leveraging PPs with groupwork. As education researchers continue to explore solutions to the complex challenges faced by students and instructors in First-Year Engineering Programing Courses, Parsons Problems are positioned as an advantageous tool that can continue to be improved on to further support student centered learning.

References

1. National Academies of Sciences, Engineering, and Medicine, Assessing and Responding to the Growth of Computer Science Undergraduate Enrollments [Washington, DC: The National Academies Press, 2018 (https://doi.org/10.17226/24926)]
2. Butler, M., & Morgan, M. (2007). Learning challenges faced by novice programming students studying high level and low feedback concepts. Proceedings ascilite Singapore, 1(99-107).
3. Bowman, N. A., Jarratt, L., Culver, K. C., & Segre, A. M. (2019, July). How prior programming experience affects students' pair programming experiences and outcomes. In Proceedings of the 2019 ACM Conference on innovation and technology in computer science education (pp. 170-175).
4. Medeiros, R. P., Ramalho, G. L., & Falcão, T. P. (2018). A systematic literature review on teaching and learning introductory programming in higher education. IEEE Transactions on Education, 62(2), 77-90.
5. Bosse, Y., Redmiles, D., & Gerosa, M. A. (2019, July). Pedagogical content for professors of introductory programming courses. In Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education (pp. 429-435).
6. Robins, A. V. (2019). 12 novice programmers and introductory programming. The Cambridge handbook of computing education research, 327.
7. Parsons, D., & Haden, P. (2006, January). Parson's programming puzzles: a fun and effective learning tool for first programming courses. In Proceedings of the 8th Australasian Conference on Computing Education-Volume 52 (pp. 157-163).

8. Ericson, B. J., Denny, P., Prather, J., Duran, R., Hellas, A., Leinonen, J., ... & Rodger, S. H. (2022). Parsons problems and beyond: Systematic literature review and empirical study designs. Proceedings of the 2022 Working Group Reports on Innovation and Technology in Computer Science Education, 191-234.
9. Prather, J., Homer, J., Denny, P., Becker, B. A., Marsden, J., & Powell, G. (2022, August). Scaffolding Task Planning Using Abstract Parsons Problems. In IFIP World Conference on Computers in Education (pp. 591-602). Cham: Springer Nature Switzerland.
10. Sweller, J. (2011). Cognitive load theory. In The psychology of learning and motivation: Cognition in education, Vol. 55 (pp. 37–76). Elsevier Academic Press. https://doi.org/10.1016/B978-0-12-387691-1.00002-8
11. Abdul-Rahman, S.-S., & du Boulay, B. (2014). Learning programming via worked-examples: Relation of learning styles to cognitive load. Computers in Human Behavior, 30, 286–298. https://doi.org/https://doi.org/10.1016/j.chb.2013.09.007
12. Morrison, B. B., Dorn, B., & Guzdial, M. (2014). Measuring Cognitive Load in Introductory CS: Adaptation of an Instrument. Proceedings of the Tenth Annual Conference on International Computing Education Research, 131–138. https://doi.org/10.1145/2632320.2632348
13. Carberry, A. R., & Baker, D. R. (2018). The impact of culture on engineering and engineering education. Cognition, metacognition, and culture in STEM education: Learning, teaching, and assessment, 217-239.
14. Hughes, J. L., Camden, A. A., & Yangchen, T. (2016). Rethinking and updating demographic questions: Guidance to improve descriptions of research samples. Psi Chi Journal of Psychological Research, 21(3), 138-151.
15. Hughes, J. L., Camden, A. A., Yangchen, T., Smith, G. P., Rodríguez, M. M. D., Rouse, S. V., ... & Lopez, S. (2022). Guidance for researchers when using inclusive demographic questions for surveys: Improved and updated questions. Psi Chi Journal of Psychological Research, 27(4), 232-255.
16. Z. Kopanidis, F., & J. Shaw, M. (2014). Courses and careers: measuring how students' personalvalues matter. Education+ Training, 56(5), 397-413.
17. Turoski, S. A. (2020). Advancing student motivation and course interest through a utility value intervention in an engineering design context (Doctoral dissertation, Montana State University-Bozeman, Norm Asbjornson College of Engineering).
18. Bennett, D., Knight, E., Bawa, S., & Dockery, A. M. (2021). Understanding the career decision making of university students enrolled in STEM disciplines. Australian Journal of Career Development, 30(2), 95-105.