Problem-based Learning Design

Rutgers Graduate School of Education

15:295:620

Jamie Liao

Overview

The goals of the problem-based learning (PBL) exercise is to introduce first-year, female university students to a subset of the key concepts within computational thinking (CT). Wing (2006) first coined the modern term and later defined it as “the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be efficiently carried out by an information-processing agent” (Wing, 2010, p. 1).

This paper follows the CT model proposed by Shute et al. (2017) and focuses specifically on the first two facets: decomposition (breaking down a complex problem into smaller, manageable parts) and abstraction (extracting key information from a problem). The PBL exercise explores the “magic” behind pattern recognition through a relevant exercise to determine how a program takes in a picture of a Rutgers bus and then displays the bus route information. The exercise is situated in an unplugged and non-programming environment.

The PBL exercise is taught in a broader course at Rutgers University called Big Ideas in Computer Science. The students doing the PBL exercise are 40 first-year women intending to major in computer science (CS) and belong to Rutgers University’s Douglass Women in Science and Engineering Program (WiSE) program, whose goal is to cultivate the advancement of women in STEM. The course’s overall goal is to increase women's retention in CS by introducing them to various topics within the field. This PBL exercise supports the goal by providing an early introduction to exciting problems they may encounter until their third- or fourth-year CS electives.

Literature review

Computational thinking research has gained a lot of popularity in the past decade. However, with the rise in interest, research on CT has branched off into many varying definitions (Shute et al., 2017; Wang et al., 2020; Chen et al., 2023). Definitions vary from relating CT directly to computer science (Wing, 2006). Other definitions directly correlate CT with problem-solving with computers, such as thinking with computers as a tool and explicitly linked with programming skills (Shute et al., 2017; Wang et al., 2020; Chen et al., 2023). In addition to just relations to computer science, CT was also connected to math and science education (Wang et al., 2020).

Similar to branching definitions, CT has branching frameworks and components in different contexts. At the farthest end, CT components were defined in terms of programming concepts, including “sequences, loops, parallelism, events, conditionals, operators, and data” (Wang et al., 2020, p. 1950). Wing (2010) detailed CT as a variety of thought processes, such as logical, system, algorithmic, and parallel thinking. It also includes larger concepts of abstraction and recursion. Shute et al. (2017) developed a CT framework that includes six facets: decomposition, abstraction, algorithms, debugging, iteration, and generalization.

According to Wing (2010, p. 1), abstraction is the “most important and high-level thought process” in CT. It supports the capacity to define patterns, create models, and extract the vital components of a problem while hiding the irrelevant details. Abstraction can be recursively layered to build larger and larger systems (Wing, 2010). For example, modern programming is the result of layers and layers of abstraction to turn human-readable code into the binary 1’s and 0’s that computers understand. Shute et al.'s (2017) model of CT further defines three subcategories: (a) data collection and analysis (collecting most relevant data), (b) pattern recognition (identifying patterns/rules underlying data structure), and (c) modeling (build frameworks to represent how a system operates).

In concert with abstraction, decomposition defines the process of breaking down a large, complex problem into smaller, manageable modular pieces (Wing, 2010). It supports the user in modifying a specific independent or dependent component of a complex problem without needing to fully understand the full details (Wing, 2006; Shute et al., 2017).

Exploring beyond the definition of CT, this paper’s PBL exercise revolves around both an unplugged and non-programming-based exercise for higher education students. In current literature, there seems to be sparse research pertaining to the three pillars of this PBL exercise: (a) PBL, (b) higher education, and (c) non-programming-based in the context of unplugged CT activities. In this context, unplugged refers to learning activities that involve offline tools such as “board games, toys, cards, puzzles, and papers” (Chen et al., 2023, p. 3). Current research on unplugged activities revolves around teaching not only younger children but also utilizing programming concepts in some capacity (Wang et al., 2022; Chen et al., 2023). At the time of writing, Moreno-Palma (2024) is the only empirical paper that fits into the context of this PBL exercise. Therefore, it’s likely that the area of research related to this PBL exercise is still in its infancy stage. The rest of the literature review will focus on the current holes in literature surrounding the three pillars.

First, the specifics of PBL come in many different forms, including the most popular contexts of “game design, robotics, and computational modeling” (Wang et al., 2020, p. 1958). An example is creating a program using Scratch, which is a click-and-drag block-based programming environment, to create a programmed solution to a problem.

Other unplugged examples include robotic kits, paper tasks, and board/card games (Moreno-Palma et al., 2024). Although these examples are programming-based, the students were given an open-ended problem to solve.

Second, Chen et al. (2023) and Huang and Looi's (2021) literature reviews specifically looked at K-12 age ranges. The focus on younger children may result from the current demand to expand computer science knowledge to younger grades. In the United States, the Next Generation Science Standards framework supported the initiative, which lists "using mathematics and computational thinking" as one of the eight core elements (National Research Council, 2012, p. 49). Moreno-Palma et al. (2024) performed an unplugged card-based PBL activity for university students. While Agbo et al. (2024) also explored university students, their study explored CT from the context of programming.

Third, most empirical studies on unplugged approaches are within the context of teaching technical programming (Huang & Looi, 2021; Chen et al., 2023). Huang and Looi (2021) reviewed 40 empirical papers and noted only one paper to separate CT from programming activities. Chen et al. (2023) also conducted a systematic literature review on unplugged CT research entirely from the programming perspective.

While papers exist in some combination of PBL, higher education, unplugged activities, and non-programming-based CT, a literature gap in the research area overlaps between all three. The most significant reason may be the lack of a cohesive definition of CT. Although Wing (2006, p. 35) explicitly mentions that "computer science is not computer programming", there are still some definitions that correlate CT with programming concepts (Wang et al., 2020). Unplugged CT activities were still focused on teaching programming concepts (Huang & Looi, 2021; Chen et al., 2023). Another reason may be that the birth of CT was rooted in computer science (Wing, 2006). Programming skills are one of the many benefits of CT (Shute et al., 2017), so assessing programming skills may be one of the more obvious assessment methods to evaluate a student's acquisition of CT skills.

Learning Outcomes

The core learning goal is for the students to better understand or practice the key CT concepts of decomposition and abstraction, including the three subcategories (data collection and analysis, pattern recognition, and modeling) mentioned by Shute et al. (2017). The students will learn each of these within the context of what happens behind the scenes when a computer program performs pattern recognition, a popular subfield of AI: (a) decomposition (how to break down the overall problem question into smaller, manageable parts), (b) abstraction (how to abstract away details of the project, such as image parsing and database usage), (c) data collection and analysis (determine what components of an image is necessary for this project and what data to store in the database/decision tree), (d) pattern recognition (identifying unique features about letters to create a decision tree to identify letters computationally), and (e) modeling (create the decision tree to "model" an AI's decision making).

The problem space contains topics on decision-making, anatomy of a picture, databases, classifying new information, and feature analysis. The related conceptual spaces include architecting a technical system, basic programming concepts such as loops, understanding what counts as good/bad input data, and basic human cognition principles such as visual search and decision-making.

Beyond the problem space learnings, the students will also broaden their understanding of the field of CS in a breadth of areas through the PBL exercise. First, the students will refine their understanding of CS itself. They'll get hands-on experience learning that CS is about solving problems and how to construct a system built with foundational CS concepts creatively. They'll learn CS is not just programming, a common misconception (Wing, 2006). Second, CS is an interdisciplinary field. This lesson will connect CS concepts to those in cognitive science, such as visual search and decision-making. Third, CS functions as a means to solve a problem in another field, an area of life, or just for fun. Similar to Pinkard et al.'s (2017) Digital Youth Diva program, the PBL exercise creates a solution to an immediately relevant problem: acclimating to the Rutgers bus system. Lastly, they'll learn the broad concept of a "creating a technical personal project," which many novice CS students struggle to grasp; it's too nuanced and personalized per student/project to give meaningful, broad advice. It's similar to telling a high schooler to "just choose a major"; this is vital because personal projects are almost necessary for CS students to find a job as a software engineer, which is the most common career path for CS majors.

Problem and Key Activities

The overarching question/problem is: In an attempt to learn the RU bus system, we want to create a personal project that takes in a picture of a bus and displays the bus route. How might a computer program accomplish this?

In groups of 3 students, the students will explore the problem in three areas with an overarching exploration phase across two class sessions. Then, they'll compile what they learned by creating a step-by-step set of rules to follow what happens in the program they map out.

In the overarching exploration phase, the students will dissect the question (decomposition) to break it into easier, manageable parts. There are many paths forward to break up the problem, and the initial exploration phase will be supported by a scaffolded worksheet of questions to guide the student's thinking. After the students have explored their options forward, the next four areas will be key aspects they should explore in some capacity during their PBL exercise. The students can explore each area in any order as they can be individually abstracted.

First, the students will break down an image and understand how to get data/information from an image file. The students will navigate the concept of pixels, the hex code of a pixel, and understand how to parse through an image.

Second, they will develop a basic table of information (also known as a relational database) to correspond bus names to bus route data. The students will be given a scaffolded version of a database in the form of a classic Excel-like table to help them organize data. Additionally, they'll understand how to use the database of information to provide input (e.g., "A" bus) and then obtain the information they desire (e.g., the route of the "A" bus).

Third, they will use feature analysis to identify the unique features of each letter. The students will start by analyzing two bus letters, "A" vs. "H," for their feature analysis. An example is identifying the presence of horizontal, horizontal, or diagonal lines and then determining which letter is likely in the image. After creating a few patterns, they'll create a very simple model in the form of a decision tree (Song & Lu, 2015) to determine which letter is present in the input image.

To complete the PBL exercise, the students will combine areas 1-3 to build a decision tree for all bus letters to create a "trained" AI classification model. In essence, they'll combine their exploration phase with the three specific areas they've explored to create an overall system design to answer the PBL question. The activity will be considered complete when the group has created a step-by-step set of rules to describe what happens between a program accepting a bus image as input and displaying the bus route.

Materials

The materials needed will correspond to the various areas described prior in the Problem and Key Activities section: exploration phase, pixels and images, databases, and decision tree.

For the exploration phase, the students will have a worksheet with questions that guide them through dissecting the problem and creating more manageable components.

Then, the students will explore images with blocks will be made out of paper and colored with markers. During the exercise, they'll move them around to "recreate" a pixelated picture to showcase that high-definition images are boiled down to millions of squares put together in a specific order.

Then, the students will receive a worksheet with a partially filled-in table (similar to Excel) that will help them correlate "finding" data in a database. This skill is not a key skill of the activity, but it's still nice-to-know information as the concept of databases is very important in CS.

Lastly, the students will receive a worksheet that scaffolds the decision tree to determine if a letter is an "A" or "H." This will include branching of key features of a letter (e.g., Does the letter have only horizontal/vertical lines- yes or no). Then, the students will produce a decision tree on their own to map out all bus letters. If given enough time at the end of class, each group of students will create a short presentation to showcase their step-by-step solution to the problem. As there's no right or wrong answer, it will be insightful for them to learn how others approached the problem.

Scaffolds

The scaffolds available for the students will also correspond to the various areas described earlier in the Problem and Key Activities section: exploration phase, pixels and images, databases, and decision tree.

First, for the exploration phase, the students will receive a hard scaffold in the form of a worksheet that has the questions listed in Table 1. Each question is accompanied by the intended learning goal and the CT concepts explored per question. These questions are intended to guide the students through the overall PBL exercise; then the students will uncover more and more information about how to achieve each step. After the initial exploration phase, this worksheet will act as a logical guide so they'll explore each abstracted component further to deepen their understanding, such as how an image is processed or how to construct/use a database.

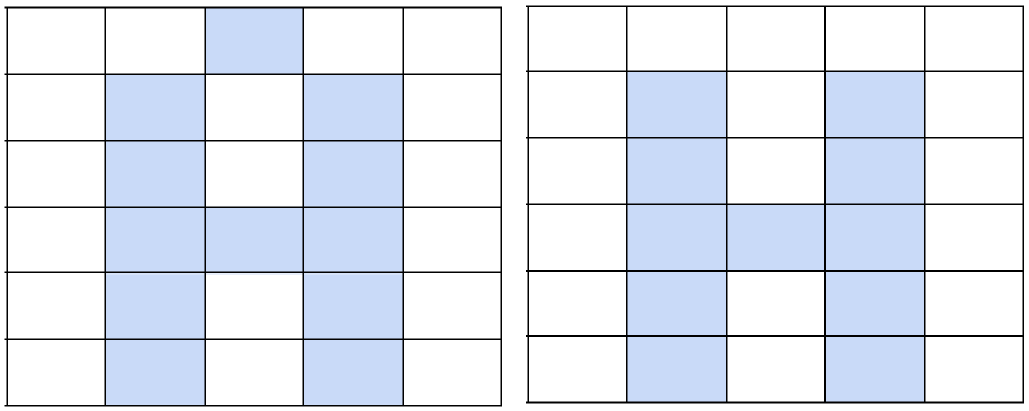
Table 1
Exploration Questions with Their Relevant Learning Goal and CT Concept

Question	**Learning Goal**	**CT Concept Explored**

What are the core problems of the problem?	Challenge the students to practice breaking down the problem into manageable parts.	Decomposition, abstraction
What do we know about the input and the type of data we can get from it?	Prompt the students to consider the image anatomy area.	Data collection and analysis
What are the various types of data or information we need? How can we organize and use it?	Prompt the students to consider the database area.	Data collection and analysis, modeling
What are some unique aspects of each letter we can use to uniquely identify them?	Prompt the students to consider feature analysis area.	Pattern recognition, data collection and analysis
What are some visual ways to represent a set of rules?	Prompt the students to consider the decision tree area.	Modeling, pattern recognition

Second, for the pixels and images area, the students will have access to paper blocks colored white or blue (Figure 1). Each group will have 30 white and 10 blue physical one-inch squares. They can move around to visually simulate image/pixel manipulation. They will move the blocks around so they create either an “A” or “H” with the “pixels.” This exercise will provide them with an abstracted version of what actually happens when a computer parses through an image to obtain the relevant information they need/want.

Figure 1
Example of White and Blue Squares in the Form of “A” and “H”



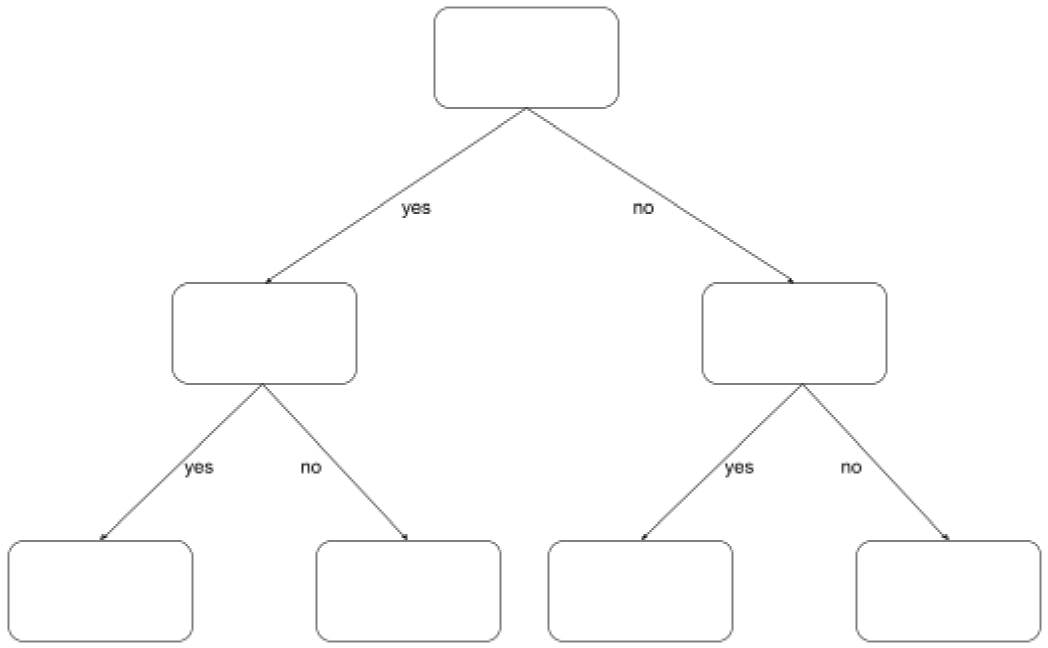
Third, for the databases area, the students will receive a worksheet scaffold to support their data organization (Table 2). The worksheet will look similar to an Excel table to organize their data. The students will observe the minimally completed example for Bus A, which will prompt them to fill in the remainder of the blanks in the table.

Table 2
Database Scaffold Example

Bus Name	
A	

Fourth, the students will be given the starting points of a decision tree to support a model that supports their decision-making (Figure 2). The students will use the scaffolded decision tree as a starting point to support them in building out the rest of their basic decision tree to determine if a letter is “A” or “H.” This scaffold will support their modeling skill in creating a flow chart of decisions.

Figure 2
Example Decision Tree Flow



All students will only be given the first exploration phase worksheet. The last three scaffolds will only be given when a group is stuck or unsure how to progress further. They are meant to adaptively support their learning by providing some structure to their exploration/thought processes. The reason for this adaptive scaffolding is to challenge and adapt to the technical levels of the students in this course. The students in the course vary from students who haven't programmed before to students who have AP'd out of Introduction to Computer Science at Rutgers.

Analysis of Problem Suitability

The PBL exercise is an attempt to teach students two major pillars of computational thinking (decomposition and abstraction) through a relevant problem. Table 1 summarizes the various phases the students will explore, including the respective CT pillars and the subcategories of abstraction (data collection and analysis, pattern recognition, and modeling).

Through this PBL exercise, the students will obtain a first-hand example on how to break down a problem into more manageable components. This skill is vital as they progress through more difficult classes in the computer science major. In line with Vygotsky's zone of proximal development, the adaptive scaffolding exercises will support students' varying technical expertise so they all accomplish the PBL exercise.

In addition to the direct goals of the PBL exercise, the students will learn different areas of computer science. Firstly, the exercise demonstrates that CS is not just about programming but rather problem-solving. Secondly, the students broaden their understanding of the field of CS, which is an interdisciplinary field, so the students can pair CS with anything they’re interested in. Lastly, the students also get introduced to the concept of “personal projects.” They are borderline necessary for students interested in pursuing a career in software engineering, yet a very confusing and daunting concept for novice computer scientists.

AI Use Log

Grammarly. (2024). Grammarly (Apr 20 version) [Large language model]. https://www.grammarly.com. Used for final grammar corrections before submission.

References

Agbo, F. J., Okpanachi, L. O., Ocheja, P., Oyelere, S. S., & Sani, G. (2024). How can unplugged approach facilitate novice students' understanding of computational thinking? An exploratory study from a Nigerian university. *Thinking Skills and Creativity*, 51, 101458.

Chen, P., Yang, D., Metwally, A. H. S., Lavonen, J., & Wang, X. (2023). Fostering computational thinking through unplugged activities: A systematic literature review and meta-analysis. *International Journal of STEM Education*, *10*(1), 47–25.

Huang, W., & Looi, C. K. (2021). A critical review of literature on "unplugged" pedagogies in K-12 computer science and computational thinking education. *Computer Science Education*, *31*(1), 83-111.

National Research Council. (2012). *A framework for K-12 science education : practices, crosscutting concepts, and core ideas*. National Academies Press.

Pinkard, N., Erete, S., Martin, C. K., & McKinney de Royston, M. (2017). Digital youth divas: Exploring narrative-driven curriculum to spark middle school girls' interest in computational activities. *The Journal of the Learning Sciences*, 26(3), 477–516.

Moreno-Palma, N., Hinojo-Lucena, F. J., Romero-Rodríguez, J. M., & Cáceres-Reche, M. P. (2024). Effectiveness of problem-based learning in the unplugged computational thinking of university students. *Education Sciences*, *14*(7), 693.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142–158.

Song, Y. Y., & Lu, Y. (2015). Decision tree methods: applications for classification and prediction. *Shanghai archives of psychiatry*, 27(2), 130–135.

Wang, C., Shen, J., & Chao, J. (2022). Integrating computational thinking in STEM education: A literature review. *International Journal of Science and Mathematics Education*, *20*(8), 1949–1972.

Wing, J. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

Wing, J. (2011). Research notebook: Computational thinking—What and why. *The link magazine*, *6*, 20-23.