# Developing computational thinking: an unplugged problem-solving approach

Tran Kiem Minh and Nguyen Ngoc Thanh

Hue University of Education, Department of Mathematics, Hue, Vietnam; tkminh@hueuni.edu.vn

*The field of computer science is advancing rapidly and is crucial to technological progress. To keep up with this progress, many countries have reformed their general education curricula to include algorithmic elements and computational thinking, particularly in mathematics. This study aims to elucidate the characteristics of computational thinking and assess the potential of an unplugged problem-solving approach in developing students' computational thinking skills within the context of mathematics education. Our preliminary findings indicate that the unplugged problem-solving approach effectively promotes students' computational thinking abilities. Based on our research results, we recommend a concerted effort to integrate algorithms and computational thinking into various school subjects, particularly mathematics.*

*Keywords: Computational thinking, unplugged problem-solving approach, algorithm, mathematics.*

## Introduction

In recent years, digital technology and computational thinking have emerged as critical components of scientific literacy that must be promoted to keep pace with the rapid development of society. Algorithmic and computational thinking are considered core digital literacy elements (Stephens & Kadijevich, 2021). The 2021 PISA Framework (OECD, 2018) states that 21st-century mathematical literacy should include mathematical reasoning and certain aspects of computational thinking.

Many countries have introduced computer science into their secondary education curricula, especially in mathematics, due to the close connection between computer science and mathematics (Bocconi et al., 2016; Lagrange & Laval, 2023). This integration aims to provide an alternative context for teaching mathematical concepts and developing computational thinking skills, such as problem-solving, pattern recognition, and generalization. In Vietnam, computer science concepts are taught to secondary school students as part of the informatics curriculum, but not as part of the mathematics curriculum. This creates a situation where certain computer science concepts, such as algorithms, are taught in a formal manner that is similar to how programming languages are taught, without any connection to real-world problem-solving contexts that are typically encountered when learning mathematics. This can result in a lack of engagement and relevance for students who are trying to learn computer science, and may make it harder for them to see how computer science can be applied in practical ways.

Mathematics and computer science share a common foundation in epistemology and history, and critical computer science concepts, such as graphs, combinatorics, and logic, are derived from mathematical objects. As a result, mathematics education can help individuals understand this emerging field of science. In turn, the tools and concepts of computer science can promote the

teaching of school mathematics in a more modern way (Stephens, 2018). The recent development of data science has presented a challenge for mathematics educators to recognize and incorporate algorithms and computational thinking into school mathematics curricula (Stephens, 2018). For example, Couderette (2016) argued that teaching algorithmic thinking is quite difficult due to the involvement of mathematics and computer science. According to Clark (2016), mathematics teachers should not consider algorithmic and computational thinking as content that needs to be taught in the same way as other mathematics content but should regard them as a problem-solving method or a specific skill that can be transferred to other subjects.

This study aims to examine the characteristics of the concept of computational thinking in literature. Then the paper argues that an unplugged problem-solving approach is suitable for connecting computer science with mathematics learning and developing students' computational thinking. Finally, the study reports some initial experimental findings related to the development of computational thinking among high school students in Vietnam through an unplugged problem-solving approach.

## Theoretical framework

### Computational thinking

Papert is widely regarded as the originator of the term "computational thinking," which he used in his book *"Mindstorms: Children, computers, and powerful ideas"* (Papert, 1980). However, Papert did not offer a clear definition of the concept. Later, Wing (Wing, 2006, 2011, 2014) provided explicit definitions of computational thinking. In her definition, she defined computational thinking as the cognitive processes involved in formulating and solving problems in a manner that can be effectively carried out by a computer or other agents. This definition highlights two critical aspects of computational thinking that are relevant to mathematics education: (1) computational thinking is a thinking process that is technology-independent, and (2) it involves problem-solving skills that include designing solutions that can be executed by a computer or a human.

Wing's definition of computational thinking has become a significant reference point for research on computational thinking in education. This definition draws on the idea of formulating problems and solutions in a way that can be carried out by an information-processing agent and the notion that solutions should take the form of computational steps and algorithms.

### Core concepts and skills related to computational thinking

Wing (2011) argues that the essential element in computational thinking is the process of abstraction. An algorithm is an abstraction of a process consisting of receiving input data, executing a sequence of steps, and producing output data that satisfies a goal. Computational thinking uses abstraction and decomposition to solve large and complex problems.

Selby and Wollard (2013) defined computational thinking as a set of core skills, including abstraction, decomposition, algorithmic thinking, evaluation, and generalization. Similarly, Bell and Lodi (2019) proposed a list of skills related to computational thinking, including abstraction, decomposition, algorithm design, generalization, logical thinking, and evaluation. The Australian Curriculum, Assessment and Reporting Authority (ACARA, 2022) also defined computational thinking as a set

of core skills, such as decomposition, pattern recognition, abstraction, modeling and simulation, algorithms, and evaluation. Explicitly, the concept of computational thinking adopted by ACARA (2022) shares the common characteristics of computational thinking presented in these definitions.

This study utilized ACARA (2022)'s computational thinking approach to develop learning situations and evaluate high school students' computational thinking abilities for two main reasons. First, ACARA's computational thinking approach is practical and emphasizes essential problem-solving skills that are not necessarily tied to technology, making them easy to integrate into various subjects and contexts in the school curriculum. Second, the study problems were sourced from the Bebras Australia Computational Thinking Challenge, which is grounded on ACARA (2022)'s definition of computational thinking as a theoretical foundation.

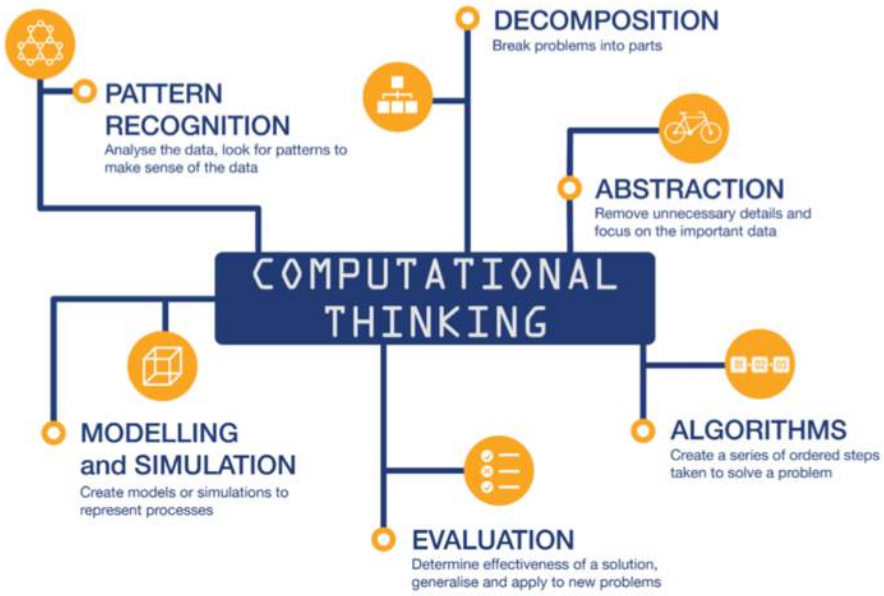



**Figure 1: Core skills related to computational thinking (ACARA, 2022)**

**An unplugged problem-solving approach to develop students' computational thinking**

The integration of computer science into school curricula is a topic of concern for computer science educators. One approach to developing students' computational thinking skills is through "unplugged" problem-solving activities, also known as CS unplugged, which teach computer science fundamentals without using computers (Curzon et al., 2014; Kotsopoulos et al., 2017). The unplugged problem-solving approach avoids the difficulties associated with computers and programming languages, which can be challenging for most students.

Unplugged problem-solving activities are a fundamental tool for developing students' computational thinking skills as they require limited technological resources (Kotsopoulos et al., 2017). These activities can be easily used and integrated into various school curricula, particularly mathematics. By exploring and resolving real-life situations with diverse contexts, students are exposed to fundamental computer science concepts and develop computational thinking skills. Unplugged problem-solving activities allow students to develop core computational thinking skills, such as

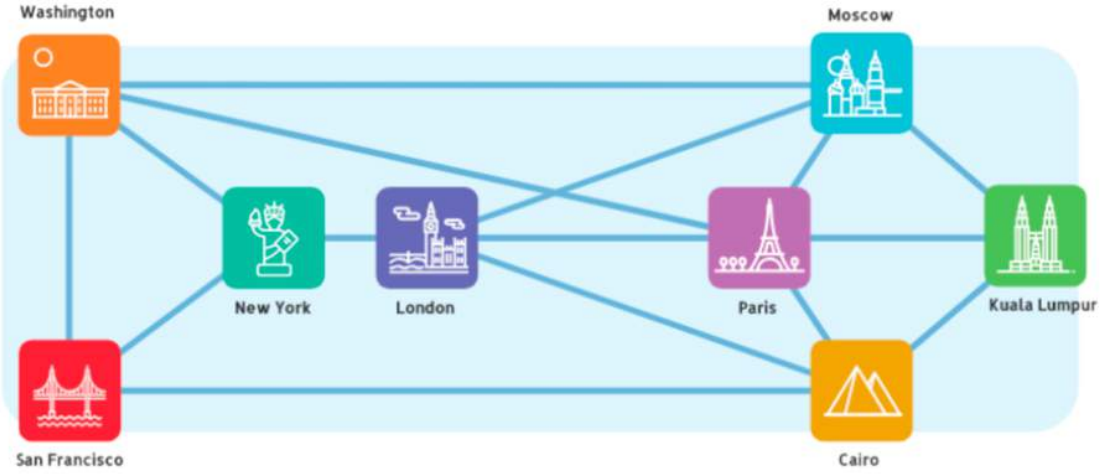
decomposing complex problems into simpler ones, identifying patterns in data, modeling situations, and proposing algorithms to solve problems.

Computer science educators advocate for exposing students to unplugged problem-solving activities early on to understand computer science's basic concepts and practices (Bell et al., 2012). Wing (2006) considers computational thinking a crucial skill in the 21st century. Many discussions have taken place on the need to develop students' computational thinking skills before learning programming (Yadav & Berthelsen, 2022).

## Methodology

This problem is one of many graph theory problems that were designed and experimented on a group of forty 15-year-old grade 10 students in Vietnam during the 2021-2022 school year. These students had no prior experience with graph theory problems. The data collection process involved three steps. First, the students were introduced to fundamental graph theory concepts and related problems. Next, the students were given a worksheet containing four graph theory problems to solve. Finally, a few students were selected for an interview. The collected data included the students' completed worksheets, audio recordings of the interviews, and the researchers' notes. The data was qualitatively analyzed to describe the characteristics of the students' computational thinking based on ACARA (2022)'s model of computational thinking.

An international airline has a lot of flight routes connecting several major cities as shown.




To reduce $CO_2$ emissions, the airline wants to cancel some of the flight routes without stopping customers from being able to fly to any city. For example, if the flight route between San Francisco and Washington, D.C. is cancelled. In those cases, customers could still fly from San Francisco to Washington, D.C. by transiting New York (unlimited number of transits).

For the flight routes shown above, what is the maximum number of routes the airline can cancel? Explain.

This problem allows students to practice various computational thinking skills, such as abstraction, modeling and simulation, pattern recognition, algorithms, and evaluation. The problem can be seen as the most basic form of the Minimum Spanning Tree (MST) problem in computer science. The MST algorithm is commonly used to address various problems related to the installation of

telecommunications networks, transportation networks, or water supply systems. The purpose of presenting this problem is to encourage grade 10 students to identify the general rule that it takes a minimum of n-1 edges to connect n vertices and create a path connecting the first and last vertices.

The problem's objective is to determine the minimum number of flight routes required to connect eight cities, satisfying the travel requirements of passengers. Students are expected to solve this problem by using one of the following methods:

- Method 1: A rule is established that n cities need at least n-1 routes to connect all cities. By applying this rule, it is determined that at least 7 routes are needed for 8 cities. To confirm this rule, we will show that there cannot be less than or equal to 6 flight routes. Assuming there are 6 routes and 8 cities, then there will be two cases: (a) a city is not on any routes; (b) a city lies on precisely one route (i.e., has an edge passing through three vertices, which is not possible based on the actual context of the problem). The second case is, of course, not possible. In the first case, passengers arriving from a city that is not on any flight route will not be able to fly to any other city. Therefore, only $15 - 7 = 8$ routes can be canceled at most.
- Method 2: This approach involves selecting a city as the starting point and drawing consecutive routes that pass through each city only once until reaching the last city. It is demonstrated that at least 7 flight routes are necessary to connect 8 cities, which can be reduced to a maximum of 8 flight routes.
- Method 3: This method involves modeling the problem as a graph, with each vertex representing a city and an edge connecting two vertices when a flight route connects the two cities. The task then becomes finding the maximum number of edges that can be reduced while maintaining connectivity in the graph. Since the graph has eight vertices, it must have at least seven edges to be connected. Thus, it is possible to reduce the number of flights to a maximum of 8.

## Preliminary results

In this section, we will discuss the different aspects of computational thinking demonstrated by students while solving the above problem. Specifically, we will focus on the following categories:

*Abstraction:* Our analysis of experimental data revealed two types of evidence related to students' abstraction ability for this problem. First, students were able to identify and emphasize important information by underlining and coloring critical phrases in the problem statement and drawing observations from graphs that showed relationships between cities or flight routes. Some students used symbols instead of city names to represent vertices. For example, student HS39 counted the number of available flight routes and had a unique way of making notes (Figure 2). Second, students demonstrated their ability to illustrate a solution to the problem visually. For instance, student HS11 marked the dotted lines in the picture (Figure 3) and created a detailed visual representation of the solution, that means canceling the following flight routes: Washington – San Francisco, Washington – Paris, Washington – Moscow, London – Moscow, Paris – Moscow, London – Cairo, Cairo – Kuala Lumpur, San Francisco – Cairo.

*Modelling and simulation:* Students demonstrated their ability to model and simulate the problem by designing a diagram or graph with related elements, such as vertices and connecting lines, to represent the flight routes. Most students exhibited this skill.

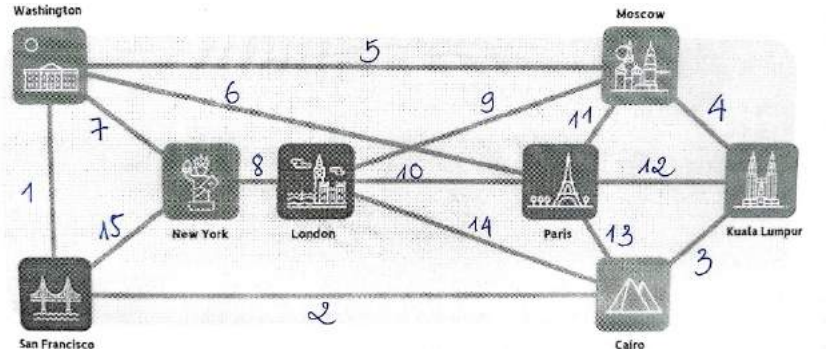



**Figure 2: Students' noticing of important information about the problem**

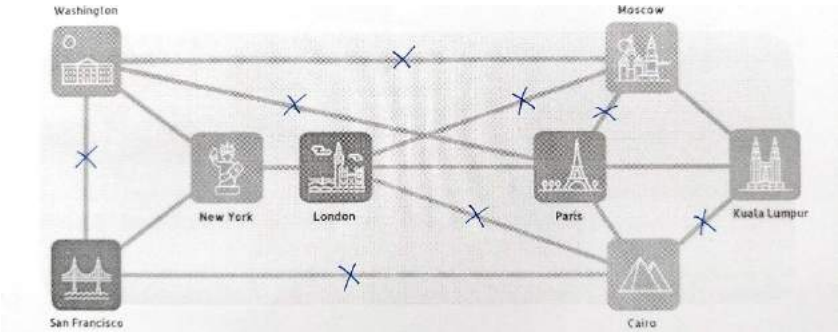



**Figure 3: Students' visual illustration of a solution to the problem**

*Pattern recognition:* Students were expected to recognize the rule that it takes at least *n-1* edges to join *n* vertices together. Students demonstrated an understanding of this rule through their arguments. For example, student HS1 said *"The flight route satisfying the problem is equivalent to the fact that the corresponding graph will be connected after reducing some edges. That graph has 8 vertices. Hence it must have at least 7 edges to be connected".* The student also indicated a specific case, as shown in Figure 4.


**Figure 4: Students' ability to recognize the rule in the problem-solving process**

*Algorithm:* Students' algorithmic skill was demonstrated by their ability to develop a series of arguments and actions that led to the problem's solution. For example, student HS2 chose a point as a landmark and drew a line to another point that did not have any line segments. This process was continued until the endpoint was reached without forming a closed loop. Students represented the flight routes as a continuous line without lifting the pen. However, some students showed poor algorithmic skills by proposing solutions that were not continuous lines or drew a closed circle (Figure 5).

*Evaluation:* The evaluation aspect was reflected in students' ability to reason about the optimality of the given solution. Most students were not able to express arguments about the optimality of the solution given to this problem. However, student HS11 explained why they could only cancel a

maximum of 8 flights and illustrated a correct solution, demonstrating the feasibility of the proposed solution (Figure 4).

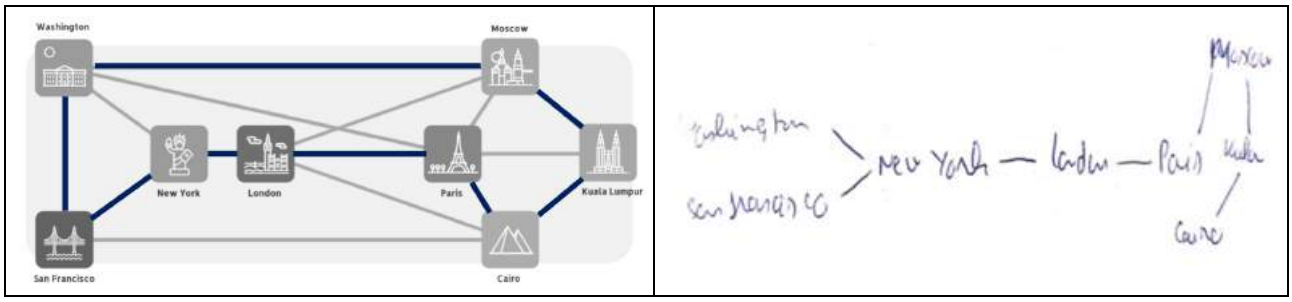



**Figure 5: The algorithmic aspect of a student's response**

Overall, these categories highlight the different ways in which students demonstrated their computational thinking skills while solving the problem at hand.

## Concluding remarks

This research focuses on the significance of computational thinking in mathematics education and explores the effectiveness of an unplugged problem-solving approach in students' computational thinking skills. Our findings from classroom observations demonstrate that this approach is feasible and can lead grade 10 students to grasp the fundamental elements of computer science and improve their computational thinking abilities.

In today's society, where computer science has witnessed significant growth, computational thinking has emerged as an essential skill for the 21st century. Many countries have introduced reforms to integrate algorithmic elements and computational thinking into their school curricula. Graph theory is a subject in the current Vietnamese upper secondary school curriculum that can potentially promote computational thinking among students. Despite not being formally addressed in the school mathematics curriculum, the integration of algorithmic elements and computational thinking has been practiced in teaching mathematics in primary and secondary schools in Vietnam, as evident in initiatives such as the Bebras Vietnam Computational Thinking Challenge and the teaching of programming to young students. This highlights the disparity between the formal educational program and students' actual needs concerning computer science education. Our research underscores the necessity and feasibility of developing students' computational thinking skills through an unplugged problem-solving approach within the context of mathematics education in Vietnam.

## References

ACARA. (2022). Computer thinking in practice. https://www.australiancurriculum.edu.au (10 February 2023).

Bell, T., Rosamond, F., & Casey, N. (2012). Computer Science Unplugged and related projects in math and computer science popularization. In H. L. Bodlaender, R. Downey, F. V Fomin, and D. Marx (Eds.), *The Multivariate Algorithmic Revolution and Beyond: Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday*, LNCS 7370, pp. 398–456. Springer-Verlag.

Bell, T., & Lodi, M. (2019). Constructing computational thinking without using computers. *Constructivist Foundations*, 342-351.

Bocconi, S., Chioccariello, A., Dettori, G., Ferrari, A., Engelhardt, K. (2016). *Developing computational thinking in compulsory education – Implications for policy and practice*; EUR 28295 EN; doi:10.2791/792158.

Clark, D. (2016). *Computational and algorithmic thinking*, 2011-2015. Canberra, Australia: Australian Mathematics Trust.

Couderette, M. (2016). Enseignement de l'algorithmique en classe de seconde : Une introduction curriculaire probl´ematique. *Annales de Didactique et de Sciences Cognitives, 21*, 267–296.

Curzon, P., McOwan, P. W., Plant, N., & Meagher, L. R. (2014). Introducing Teachers to Computational Thinking Using Unplugged Storytelling. *Proceedings of the 9th Workshop in Primary and Secondary Computing Education (pp. 89–92)*. New York, NY, USA: ACM. https://doi.org/10.1145/2670757.2670767

Kotsopoulos, D., Floyd, L., Khan, S., Namukasa, I. K., Somanath, S., Weber, J., & Yiu, C. (2017). A pedagogical framework for computational thinking. *Digital Experiences in Mathematics Education, 3 (2)*, 154-171.

Lagrange, J-B., & Laval, D. (2023). Connecting algorithmics to mathematics learning: a design study of the intermediate value theorem and the bisection algorithm. *Educational Studies in Mathematics, 112*, 225-245.

Organisation for Economic Cooperation and Development (OECD) (2018). PISA 2021 Mathematics framework (draft). Paris: The author. https://www.oecd.org/pisa/sitedocument/PISA-2021-mathematics-framework.pdf

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books.

Selby, C. C., & Woollard, J. (2013). *Computational Thinking: The Developing Definition*. University of Southampton (E-prints).

Stephens, M (2018) Embedding algorithmic thinking more clearly in the mathematics curriculum. In Shimizu Y, Withal, R (eds.), *Proceedings of ICMI Study 24: School mathematics curriculum reforms: challenges, changes and opportunities* (pp. 483–490). University of Tsukuba, Japan.

Stephens, M., & Kadijevich, D. M. (2021). Algorithmic Thinking: Emerging Implications for School Mathematics Education. *Paper presented at the 14th International Congress on Mathematical Education, ICME 14*, Shanghai, China.

Yadav, A., & Berthelsen, U. D. (2022). *Computational Thinking in Education A Pedagogical Perspective*. New York: Routledge.

Wing, J.M., (2006). Computational Thinking. *Communications of the ACM*, *49* (3)

Wing, J. M. (2011). Research Notebook: Computational Thinking - What and Why? *The Link Newsletter, 6,* 1–32.

Wing, J. (2014). Computational thinking benefits society. *40th Anniversary Blog of Social Issues in Computing*.