DOI: 10.47743/eufire-2024-1-14

QUALITATIVE ANALYSIS OF THE RELATIONSHIP BETWEEN MATHEMATICAL THINKING AND COMPUTATIONAL THINKING FROM THE PERSPECTIVE OF SOFTWARE ENGINEERING LECTURERS AND STUDENTS

ELENA KRAMER
Alexandru Ioan Cuza University of Iasi
Iasi, Romania
elenak@braude.ac.il

Abstract
In education, computational thinking—which is characterized as a style of thinking applicable to various professions needing problem-solving abilities—has grown in popularity. Future specialists must be prepared for the sophisticated thinking skills required to solve social and business challenges, necessitating a combination of mathematical thinking and computational thinking. Since the field of computer science was derived from mathematics, the connection between these two fields is obvious. Moreover, there is a correlation between ability in specific mathematical and computational fields; the question is which precise fields are correlated?
To derive a more precise hypothesis about the relationship among abilities in specific mathematical and computational fields, we adopted a novel approach – examining the relationship among local metalanguages of various fields in mathematics and computer science. The hypothesis of this research posits that if any mathematical and computational fields have similar metalanguages, then a correlation exists between ability in these fields. This information can aid in formulating content that is less comprehensible to students into a more accessible format.
In the first research stage, data mining techniques were employed. As a result, we identified clusters of similar fields in mathematics and computer science, based on similarity between metalanguages. Upon formulating the hypothesis, we verify it using both quantitative and qualitative research involving students' participation. This paper presents the results of qualitative content analysis of interviews with software engineering students and lecturers.
Keywords: *mathematical thinking; computational thinking; undergraduate students; education skills; metalanguage.*
JEL Classification: C6, A22.

1. INTRODUCTION

Computers and programming have revolutionized the world and have promoted technology literacy as a crucial skill to achieve academic and career success in the digital 21st century (Shute, Sun and Asbell-Clarke, 2017).

Consequently, computational thinking (CT), which can be defined as a way of thinking that can be applied to various fields requiring problem-solving skills, has become popular in education.

There is a need to prepare students, future specialists, for a complex thinking competence necessary for solving business and societal problems, for which a combination of mathematical thinking (MT) and computational thinking (CT) is required.

A key challenge is determining how to test thinking. We propose considering the linguistic dimension, acknowledging that thought occurs within a language framework (De Saussure, 1916; Heidegger, 1927). According to Noam Chomsky, "the father of modern linguistics," there is a strong relationship between language and thinking (Chomsky, 2006).

Given that mathematics and computer science are governed by explicit languages, and individuals employ metalanguage in their thought processes, we can explore the interconnections between various metalanguages to assess the nature of thinking. According to Alfred Tarski, metalanguage is the language in which linguistic forms, the meaning of expressions and sentences, the use of language, as well as the admissibility of formations, and the truth of statements are discussed (Tarski, 1944; Gruber, 2016). In other words, a metalanguage is a language used to describe another language. It consists of terms, specific syntax construction of sentences, a specific order of words in any sentence.

To receive a more explicit hypothesis regarding the relationship among abilities in specific mathematical and computational fields, a new approach was applied – comparing the metalanguages of different fields in Mathematics and Computer Science separately. In this Data mining stage (Cheng, 2017; Hand, 2007), we compared many text files from different fields in mathematics and computer science.

The following fields in mathematics were investigated: linear algebra, abstract algebra, combinatorics and probability, mathematical analysis, set theory, and logic. In computer science, the fields that were investigated include functional programming, imperative programming, object-oriented programming, data structures and algorithms, automata and formal languages, and operating systems. These are the basic fields that are included in most learning programs for students of computer science engineering.

The steps applied were:

- Using the FastText model for text classification and converting to vectors.
- Calculating the distance between vectors (Cohen *et al.,* 2009; Van Dongen and Enright, 2012).
- Applying clustering algorithms (Broder *et al.,*1997).

As a result, groups of fields in mathematics and computer science with closed metalanguages were received.

This paper presents the results of a qualitative content analysis conducted to verify the hypothesis that fields with closed metalanguages must provide the same understanding of their content.

2. METHODS

The primary aim of the interviews was to evaluate the central hypothesis of this research which suggests that students are likely to achieve the similar levels of success across various computer science and mathematical disciplines that have closed metalanguages. Given that individuals often gravitate towards areas where they experience success, the interviews also seek to determine whether there is an interest in fields with closed metalanguages.

Another objective was to gather insights on individuals' perspectives concerning the relationship between mathematical and computational thinking, as well as the potential for enhancing these cognitive processes to achieve greater success in related fields.

2.1 Participants profile

The research participants were 10 students studying software engineering at a college of engineering. These students were chosen in their final academic year of studies, so they had already studied the courses taken as representative fields in mathematics and computer science. They also bring a varied background level in mathematics and computer science from school education.

Additionally, the research includes a contingent of 10 lecturers from the college of engineering who instruct courses in both mathematics and computer science fields. These lecturers also have IT applied experience.

Participants received comprehensive information about the research's purpose, and they were informed that their participation was voluntary and that their anonymity would be maintained. Additionally, the research was conducted with the approval of the college ethics committee.

2.2 Interview Guide

I designed the interview questions to address all the research questions and gather people's opinions regarding the relationship between the two types of thinking. Before distributing an interview guide, it underwent expert validation. The guide, initially designed as semi-structured, underwent adjustments during the expert validation phase and interviews. These modifications were prompted by the interviewees' responses, even though most of the questions retained their original structured format.

The interview method was chosen for this research because it allows for an in-depth exploration of the perspectives and experiences of participants. Specifically, interviews provide rich qualitative data that can capture the complex and subjective aspects of how individuals perceive and relate to these forms of thinking. This method is particularly effective in educational research where the goal is to understand not just the outcomes but also the processes and cognitive frameworks that underpin learning and problem-solving in fields like mathematics and computer science.

Interviews enable researchers to probe deeper into the thought processes of students and lecturers, allowing for a comprehensive understanding of how mathematical and computational thinking are interrelated and how they influence each other in educational settings. By engaging participants directly, the research can uncover insights that might not be evident through quantitative methods alone, such as standardized tests or surveys. Additionally, the use of interviews supports the exploration of hypotheses related to the use of metalanguages in these disciplines, as participants can articulate their experiences and reasoning in a way that reveals underlying cognitive processes.

References to the benefits and applications of qualitative interviews in educational research can be found in works by O'Connor and Gibson (2003), who discuss the value of qualitative data in providing context and depth to research findings, and Mayring (2004), who outlines the systematic approach to qualitative content analysis that ensures the reliability and validity of the insights gained from interviews.

The interview guide is outlined in Table 1.

Table 1. The interview questions

<table>
<tr><td rowspan="4">For lecturers only</td><td>Gender</td></tr>
<tr><td>Educational Background</td></tr>
<tr><td>Teaching experience</td></tr>
<tr><td>IT applied experience</td></tr>
<tr><td rowspan="5">For students only</td><td>Gender</td></tr>
<tr><td>Level and quality of prior mathematical knowledge at school</td></tr>
<tr><td>Level and quality of prior computer science knowledge at school</td></tr>
<tr><td>At what age have you been exposed in programming?</td></tr>
<tr><td>Assess your interest in mathematics and computer science before beginning your college studies, and re-evaluate it as you approach the end of your studies in college</td></tr>
<tr><td colspan="2">From the following list of courses, which ones did you include in your specialization? If you must divide them into exactly two categories, which courses would you place in each of the two categories? Are there any courses that seem “similar” to you? Please explicitly state why you consider them to be similar.
The list is: linear algebra, calculus 1, combinatorics and probability, logic, discrete mathematics 1, abstract algebra, introduction to system programming, data structures and algorithms, java programming, automata and computation theory, operation systems,</td></tr>
</table>

programming Languages.
If you can divide all the courses from the list into categories of “similar courses” (not necessarily only two categories), is the division different from the previous question? By what criteria did you divide? Could it be related to their metalanguages?
Suppose there were two courses (not necessarily from the given list), you are interested in while studying these courses. What do you think about the differences and similarities of these courses?
How would you explain to someone mathematical thinking, what is specific to this type of thinking? Computational thinking?
What do you think are the unique properties of mathematical thinking and computational thinking that differentiate them/that make them similar?
Would you rather have a description of the task to be performed - as a list of requirements or in pseudo code? What is the reason?
Would you rather have the proof of a theorem in mathematics - as a formal proof or a textual explanation? What is the reason?
How does the development of mathematical thinking help develop computational thinking? How does the development of computational thinking help develop mathematical thinking? What age is appropriate to begin teaching these two types of thinking?
Does our college's software engineering curriculum follow the proper sequence for the subjects of computer science and mathematics? If not, what can be improved to help students get the tools required to develop mathematical and computer science thinking?

2.3 Summary of the Content Analysis

The primary steps involved in content analysis draw inspiration from known sources, including works by Berelson (1952), O’Connor and Gibson (2003), Mayring (2004), Zhang and Wildemuth (2009) and LibGuides N.C.U. (2022).

Here are the main steps involved in conducting qualitative content analysis:

1. define the units and categories of coding.
2. develop a coding scheme.
3. code the content.
4. analyse the results.

In the upcoming section, the results of a content analysis will be found. This analysis led to the identification of various categories and themes. Additionally, the connections between these selected categories, relevant quotes, themes, and their meaning will be presented.

3. RESULTS

After conducting the content analysis, codes were generated. Afterward, seven major categories were associated with these codes:

1. Socio-demographics
2. Similarities and differences between studied courses
3. The properties and definition of computational thinking
4. The properties and definition of mathematical thinking
5. Computational thinking skills
6. Mathematical thinking skills
7. The relationship between two types of thinking

After analyzing the interview content, the following themes emerged:

- Students and lecturers categorize courses for different reasons, with students emphasizing practicality.
- Both students and lecturers divide courses into groups close to clustering division based on metalanguages similarity, but they do not think this is a reason for their division.
- The most important components of computational thinking are engineering thinking and algorithmic thinking for finding solutions. Mathematical thinking requires precision and is more about formulating problems than solving them.
- Computational thinking ability cannot exist without mathematical thinking ability, but it is possible that due to excessive interest in computer science courses, interest in mathematics decreases, leading to academic failures.
- Children should be introduced to programming from a young age. For students who have been exposed to it early on, their interest and success tend to increase throughout their studies.

The following table (Table 2) presents categories, themes, selected quotes and their relations.

Table 2. Themes and their relations to categories and selected quotes

Theme 1: Students and lecturers categorize courses for different reasons, with students emphasizing practicality.
Relation to categories:
Socio-demographics. Similarities and differences between studied courses.
Selected quotes:
Students: “I will divide the courses according to what is useful for work and less useful for work.” “Courses that are more theoretical and I didn’t get to meet them at work.” “I combine the Logic course with computer science courses because it develops the type of thinking I require for my work.” **Lecturers:** “The third group includes courses that can be taught both mathematically and in the

computer science style. For instance, a logic course." "These are fewer engineering courses, more mathematical" "All the courses in this group are actually from the field of discrete mathematics"
Theme 2: Both students and lecturers divide courses into groups close to clustering division based on metalanguages similarity, but they don't think this is a reason for division,
Relation to categories:
Similarities and differences between studied courses
Selected quotes:
"I don't think the courses I was interested in have a similar structure of their text" "Courses I put in this group differ in the structure of the proofs."
Theme 3: The most important components of computational thinking are engineering thinking and algorithmic thinking for finding solutions. Mathematical thinking requires precision and is more about formulating problems than their solving.
Relation to categories:
The properties and definition of computational thinking The properties and definition of mathematical thinking
Selected quotes:
Students: "Computational thinking is the ability to solve problems by, sometimes, using mathematical tools." "Mathematical thinking involves the ability to translate a problem from one's mind into formal, precise form." **Lecturers:** "Mathematicians formulate problems" "Computational thinking is the solution of precisely formulated problems. And this is an engineering approach." **"**Mathematical thinking is characterized by a set of well-defined rules and definitions." "Computational thinking involves analytical calculations and the development of algorithms. It is also akin to engineering thinking."
Theme 4: Computational thinking ability cannot exist without mathematical thinking ability, but it is possible that due to excessive interest in computer science courses, interest in mathematics decreases, leading to academic failures.
Relation to categories:
Computational thinking skills. Mathematical thinking skills. The relationship between two types of thinking.
Selected quotes:
Students: "I'm fine with math; I just didn't have time to invest in it during my degree." "I feel that a solid mathematical foundation significantly has helped me to succeed in

computer science courses” “Because I was deeply immersed in computer science, I ended up neglecting math.” **Lecturers:** “Based on my more than 20 years of experience as a lecturer, I’ve observed that students who excel in computer science tend to have mathematical thinking ability.” “Computer science field is derived from mathematics, and it’s inconceivable that successful computer science students lack mathematical thinking.”
Theme 5: Children should be introduced to programming from a young age. For students who have been exposed to it early on, their interest and success tend to increase throughout their studies.
Relation to categories:
Socio-demographics. Computational thinking skills.
Selected quotes:
Students: “I was introduced to programming at age 7, and my interest grew during my studies.” “I was introduced to programming during my school years. In college, I was able to tackle complex subjects that had previously sparked questions in my mind.” **Lecturers:** “Computational thinking should be cultivated from an early age, beginning in school. This approach shaped my educational journey, and by the time I pursued my degree, I had a clear understanding of my academic interests.” “My son, who is seven years old, is enrolled in enrichment classes focused on computational thinking at school. I observe that these classes are contributing positively to his development.”

4. DISCUSSION

In this section, the insights discovered from the themes, their meaning, and the conclusions will be presented. Several insights can be drawn from the content analysis presented above.

First, students and lecturers divide courses into groups for different reasons, with students emphasizing practicality. Students and lecturers received a list of courses in the field of computer science and in the field of mathematics. They were asked to divide these courses into groups. When the request was to divide into two groups, all lecturers referred to courses by content (selected quotes of lecturers in Theme 1, Table2), while students categorized courses in mathematical field as useful or non-useful (selected quotes of students in Theme 1, Table2). When the request was to be divided into more groups, both lecturers and students split them into three groups, following a similar approach but with different explanations. This observation highlights that since experienced lecturers who teach courses in both fields can naturally discern the content of various courses based on their subject matter, software engineering students highly tend to prioritize computer science-related courses, considering mathematical courses only as auxiliary tools for developing skills in computer science field.

Second, both students and lecturers categorize courses close to clustering division based on metalanguages similarity, but they do not think this is a reason for division. This is relatively logical because a human being is incapable of comparing metalanguages within their mind. Instead, technological tools, such as those demonstrated in this article or any neural network, are necessary for comparison. The aim was to explore if people can recognize similarities between the courses when it is known that they have similar metalanguages.

Third, the most important components of computational thinking are engineering thinking and algorithmic thinking for finding solutions. Mathematical thinking requires precision and is more about formulating problems than solving them.

The conclusion reached is supported by literature (Kaufmann and Stenseth, 2020; Rambally, 2016; Wing, 2006). To discuss the relationship between computational thinking and mathematical thinking, it is crucial to comprehend the distinct characteristics of each. It is noteworthy that distinguishing between computational and mathematical thinking was challenging for interview participants due to many shared properties.

Computational thinking ability cannot exist without mathematical thinking ability, but it is possible that due to excessive interest in computer science courses, interest in and dealing with mathematics decreases, leading to academic failures.

Both lecturers and students agree that a mathematical foundation is essential for success in computer science courses. However, some students excel in computer science courses while struggling with mathematics. According to student responses, they do not perceive the importance of delving deeper into mathematical courses, which may contribute to their difficulties. The issue may stem not from a lack of ability but rather from inadequate investment.

Children should be introduced to programming from a young age. For students who have been exposed to it early on, their interest and success tend to increase throughout their studies.

Students who were introduced to computer science at a young age reported that their interest in the field grew even more during their degree. While they concentrated on subjects they could not study earlier, the foundation of interest had already been established in their minds. According to their professional experience, lecturers also believe that early exposure to computer science is beneficial for students.

5. CONCLUSION

One of the interview conclusions is that students highly tend to prioritize computer science courses over mathematical courses. Additionally, the data gathered through the interviews suggest that a mathematical foundation is crucial for excelling in computer science courses. Therefore, it is important to foster an interest in mathematics among students. The key challenge lies in encouraging them to view mathematics not merely as a tool for computational thinking but as a subject with its own intrinsic value. One possibility is to rephrase mathematical course contents as a computer science course with a metalanguage accessible to students. For instance, if a student is familiar with the metalanguage of algorithms field but not combinatorics field, we can formulate the combinatorics course contents by an algorithms course metalanguage form. Today, there are AI tools that allow formulation in any style.

Our ongoing research determined if an experiment to formulating by different metalanguage will improve student succeeding in areas where they are less proficient.

References

1) Berelson, B. (1952). *Content Analysis in Communication Research*. Glencoe, III: Free Press.
2) Broder, A., Glassman, S., Manasse, M. and Zweig, G. (1997). Syntactic clustering of the web. *Computer Networks and ISDN Systems,* 29(8), pp. 1157–1166.
3) Cheng, J. (2017). Data-mining research in education. *arXiv preprint* arXiv:1703.10117.
4) Chomsky, N. (2006). *Language and Mind*. Cambridge University Press.
5) Cohen, I., Huang, Y., Chen, J. and Benesty, J. (2009). Pearson correlation coefficient. In: *Noise reduction in speech processing*, 1-4.
6) De Saussure, F (2011). *Course in general linguistics*. Columbia University Press.
7) Gruber, M. (2016). *Alfred Tarski and the "Concept of Truth in Formalized Languages": A Running Commentary with Consideration of the Polish Original and the German Translation*. (Vol. 39). Berlin: Springer.
8) Hand, D.J. (2007). Principles of data mining. *Drug safety,* 30, pp. 621-622.
9) Heidegger, M. (2010). *Being and Time*. Albany: SUNY Press.
10) Kaufmann, O.T. and Stenseth, B. (2020). Programming in mathematics education. International *Journal of Mathematical Education in Science and Technology*, 52(7), pp. 1029–1048.
11) LibGuides N.C.U. (2022). *Qualitative Content Analysis*. [online] Available at: https://library.ncu.edu/c.php?g=1007180&p=7392331 [Accessed 24.11. 2022].
12) Mayring, P. 2004. Qualitative content analysis. *A companion to qualitative research,* 1(2), pp.159-176.

13) O'Connor, H. and Gibson, N. (2003). A step-by-step guide to qualitative data analysis. Pimatisiwin: *A Journal of Indigenous and Aboriginal Community Health*, 1(1), pp.63-90.
14) Rambally, G. (2016). The synergism of mathematical thinking and computational thinking. In :D. Polly, ed., *Cases on Technology Integration in Mathematics Education*. Hershey, PA: IGI Global, pp. 416–437.
15) Shute, V.J., Sun, C. and Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, pp. 142-158.
16) Tarski, A. (1944). The semantic conception of truth: and the foundations of semantics. *Philosophy and phenomenological research,* 4(3), pp. 341-376.
17) Van Dongen, S. and Enright, A.J. (2012). Metric distances derived from cosine similarity and Pearson and Spearman correlations. *arXiv preprint* arXiv:1208.3145.
18) Wing, J. (2006). Computational thinking. *Communications of the ACM*, 49(3), pp. 33–35.
19) Zhang, Y. and Wildemuth, B.M. (2009). Qualitative analysis of content. *Applications of social research methods to questions in information and library science*, 308(319), pp. 1-12.