Challenges encountered in mathematical problem-solving through computational thinking and programming activities

Said Hadjerrouit, Nils Kristian Hansen

University of Agder, Institute of mathematical Sciences, Kristiansand, Norway

said.Hadjerrouit@uia.no, nils.k.hansen@uia.no

This paper aims at exploring the challenges students face when engaging in mathematical problem-solving through computational thinking (CT) and programming by a combination of theoretically derived insights and task-based activities. The main method used is a semi-structured interview of one undergraduate student who was presented with a mathematical task to solve while responding to questions in a dialog with the teacher on the mathematical problem solving process through CT and the programming language MATLAB. Conclusions are drawn from the results to promote CT and programming in mathematics education.

Keywords: Algorithm, computational thinking (CT), MATLAB, mathematical problem-solving, usability

INTRODUCTION

Mathematics students are expected to have basic CT skills in parallel to emerging programming languages (Wing, 2014). Moreover, CT as a competency for future work in society should be acquired by all university mathematics students. It is argued that CT can improve mathematical problem-solving by benefitting from the power of computational processes and programming languages (Shute, Sun, & Asbell-Clarke, 2017). This study explores the challenges students face when engaging in mathematical problem-solving through CT and MATLAB.

THEORETICAL BACKGOUND

CT, or similar designations such as algorithmic thinking, is becoming an important learning goal at all levels of mathematics education. According to Misfeldt and Ejsing-Duun (2015), CT is described as the ability to work with algorithms understood as systematic descriptions of problem-solving and construction strategies. Similarly, Wing (2014) describes CT as "the thought processes involved in formulating a problem and expressing its solution(s) in such a way that a computer—human or machine—can effectively carry out". More precisely, algorithmic thinking can be defined as the process of solving a problem step-by-step in an effective, non-ambiguous and organized way that can be translated into instructions to solve problems of the same type by an individual or a computer (Filho & Mercat, 2018). The main commonality between CT and mathematical thinking is problem-solving processes (Wing, 2008). CT is also quite similar with engineering thinking in terms of design and evaluation of processes (Pérez-Marína et al, 2018). Moreover, CT and algorithmic constructs such as variables and flow statements (if-then-or-else, for, repeat, etc.) are closely connected to arithmetical and mathematical thinking (Lie, Hauge, & Meaney). This close connection might provide opportunities for mathematical problem-solving by means of CT and programming. Thus, students' mathematical background and problem-solving skills are critical for building efficient algorithms for problem-solving rather than trial-and-error and getting the program to run (Topallia & Cagiltay, 2018). Moreover, CT requires students to be engaged in a continuously changing, problem-solving process in interaction with the computer.

Drawing on these research studies, the paper proposes a three-step approach to solve mathematical problems by means of CT and programming languages. *Firstly*, students should have a good mathematical background to benefit from CT and programming languages. More specifically, they should be able to benefit from their knowledge to make sense of a mathematical task and have a good understanding of it before formulating an algorithm and programming the solution. *Secondly*, CT, in turn, should enable students to analyse and decompose the mathematical task and design an algorithm and how to perform it step-by-step before programming it. Engaging students in mathematical problem-solving through CT may enable a better understanding of mathematics beyond textbook mathematics and paper-pencil techniques. *Thirdly*, students should be able to translate the mathematical problem and the associated algorithmic solution to the constructs of the programming language. This presupposes that the language is usable. Performing programming activities in mathematics education may provide opportunities to gain knowledge that is otherwise difficult to acquire without experimenting with the program and thinking algorithmically. However, this might be difficult to achieve unless the mathematical tasks are well-designed, and the programming language is usable.

When referring to the term "usability", the research literature focuses on educational software such as GeoGebra, CAS, SimReal, etc. (Artigue et al., 2009; Bokhove & Drijvers, 2010; Hadjerrouit, 2019). However, programming languages are different from educational software and how they are used to implement mathematical problems. Hence, evaluating the usability of programming languages might not be as straight forward as it may seem. Still, three usability criteria can be applied to programming languages with slight modifications. Firstly, the extent to which the language is easy to use and allows a quick familiarization with it. The second criterion aims at whether the constructs of the programming language (variables, if, for, while, etc.) are difficult to grasp. The third criterion is the feedback provided by the language in terms of error messages, and whether these are useful to foster a successful implementation of the mathematical problem through correcting and improving the program.

Finally, engaging students in mathematical problem-solving through CT and programming languages should be placed in a pedagogical context to enable a good degree of autonomy so that the students can work on their own and have a sense of control over their mathematical learning. Clearly, students should be able to acquire knowledge without being completely dependent on the teacher. Moreover, CT and programming languages should be a motivational factor for learning mathematics and should support students' engagement in problem-solving by means of motivating tasks that are tied to the students' mathematical activities. Another pedagogical issue concerns students' interactions with the language and the feedback it provides to foster computational thinking when solving mathematical problems.

THE STUDY

Context of the Study and Research Question

This work is a single case study conducted in the context of a first-year undergraduate course on programming with applications in mathematics. The participant was one student from one class of 8 students enrolled in the course in 2019. The student had average knowledge background in mathematics, but no experience with programming languages. The course introduced the basics of algorithmic thinking and the core elements of MATLAB, that is variables, flow statements, e.g. loops, if-then-or-else, for, repeat, etc. MATLAB suits well mathematical problem-solving because the solutions are expressed in familiar mathematical notation. The research question is: *What challenges do students face when engaging in mathematical problem-solving through CT and MATLAB?*

Mathematical Task

The mathematical task presented to the student is: The length of a curve may be approximated using Pythagoras' theorem by positioning a triangle adjacent to the curve (Fig. 1, left, below). The length of the green line between A and B may then be approximated as $\sqrt{x^2+y^2}$. The task is to write a MATLAB function approximating the curve length of $f(x) = 2^x$ between two given x-values (Fig. 1, right, below).

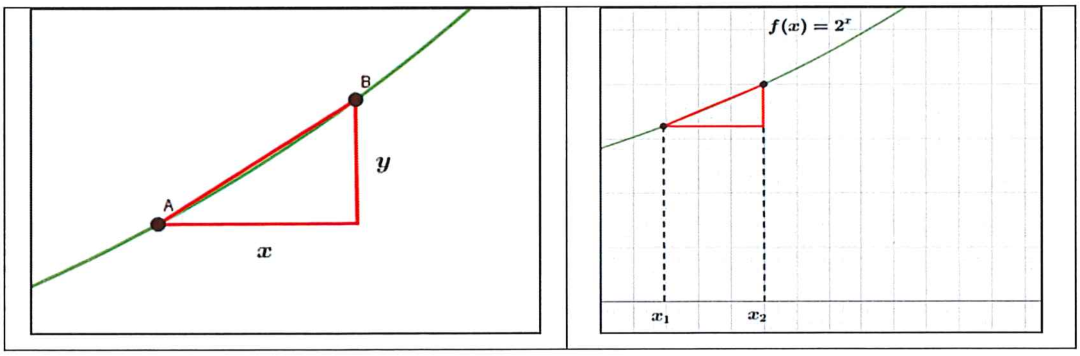

Figure 1: The mathematical task

The student was then presented with the following skeleton of a MATLAB function: function length=length estimate (x1, x2), length=? The student was asked to enter the formula, based on x1 and x2, and replace the question mark. The MATLAB function sqrt(x) can be used to calculate the square root √x.

Data Collection and Analysis Method

The main data collection method is a task-based semi-structured interview of one student who was presented with a mathematical task to solve, while responding to questions in a dialog with the teacher on the mathematical solving process by means of CT, algorithms, programming activities with MATLAB, and teacher assistance.

The order of the interview questions follows roughly the three-step approach to problem-solving presented in the theoretical framing, sometimes moving back and forth depending on the student's responses. Likewise, the data analysis method relies on the three-step approach to problem-solving, that is:

a) Understand the mathematical problem
b) Analyse and decompose the mathematical task, and then design an algorithm and how to perform it step-by-step before programming it
c) Finally, translate the algorithmic solution to the programming language code

More specifically, the student is expected to solve the task in three steps as follows:

a) Understand the task, that is using Pythagoras' theorem to calculate the hypotenuse
b) Formulate an algorithm, that is find the lengths of the triangle hypothenuse using the function $f(x) = 2^x$ and relate it to x_1 and x_2
c) Translate the algorithm into MATLAB code, corresponding to: function length = lengthEstimate (x1, x2); length = sqrt((x2-x1)^2 + (2^x2 - 2^x1)^2)

The analysis of the results seeks indications of problem-solving through CT by means of MATLAB according to this three-step approach. This is not the same as analysing and coding in the sense of grounded theory without theoretical background. Rather, the analysis tries to address the research question about the challenges encountered when solving a mathematical problem through CT and MATLAB. The interview data were transcribed and analysed according to an inductive strategy based on the interplay between the three-step approach to problem-solving and the empirical data collected by means of the semi-structured interview (Patton, 2002).

RESULTS

The results describe how the participating student engages in mathematical problem-solving through CT, MATLAB and teacher assistance. The student was given the task described above, and paper and a pen to use. It took some time before the student made sense of the mathematical task. The teacher asked the student to develop a skeleton of the solution. The student did, and started thinking about the length, but suggested an incorrect solution, based only on the values of x1 and x2. After some calculation trials, the student noticed that the attempted solution was wrong. Then, the teacher encouraged the student to think computationally.

T: But before you start using MATLAB, are you going to make an algorithm (..), for problem-solving before you start using MATLAB?

S: I just have to sit and think about it.

But still, the student continued guessing and calculating without thinking computationally. After an attempt to make sense of the task and calculate the length of the curve connecting it to the hypotenuse with a trial and error approach, the teacher provided a hint.

T: Now you say you know the hypotenuse and calculate y. But the hypotenuse is the unknown numeral here, the one you are supposed to calculate.

S: Yes.

T: So now you have turned the problem around (...). It is just that that thought was a little backwards, maybe.

S: Yes, it is quite possible.

After this short dialogue, the student started using MATLAB without developing an algorithmic solution and a clear strategy for solving the problem. The teacher then engaged in a discussion to guide the student step by step towards an algorithmic solution. Afterwads, the teacher tested the function *"on zero and one and then I got 1.4"*. Likewise, the student tested the formula and found 1.4142. The dialogue continued:

T: (...). Do you have anything to say about (...) like that afterwards?

S: No, I am, I was a little bit in doubt about how to (...) First, it was the task you asked about (...) and then it was (...) and then I thought (...) f(x) is the function in x2 would be that point minus the function of that point (...) that it would be the length. But that is where I was wrong, I felt (...) Because you meant it to be here (...) and I understand that now.

T: Yes, that is the point, (...). That is why you have to use Pythagoras to find (...) Did you think (...) There was a hint here, wasn't there? Square root?

S: Yes, yes, yes, the square root (...). I knew it was probably wrong, but I just didn't quite understand what it was.

T: Well, because there was a clue there that you couldn't use, wasn't there. Then you realize that there is something (...)

This excerpt shows there is little indication that the student was following a problem-solving strategy based on a clear understanding of the problem before formulating an algorithm and starting programming. A few minutes later, the teacher asked the student if there is a tendency to favour pen and paper to solve the task algorithmically before starting using MATLAB since developing an algorithm does not automatically require using the computer.

S: If I have it in my head, sometimes I start with MATLAB, and then I write some sort of sketch before going through it carefully. If I am not sure, I will start with paper.

T: Maybe the task was not quite clear?

S: Yes, so far, but I had probably forgotten some of the principles there.

T: Principles related to MATLAB or to the mathematical assignment?

S: To the mathematical problem.

Again, this excerpt shows that the student does not have a good understanding of the mathematical task in order to develop an algorithmic solution before programming it. Therefore, the teacher reminded the student about the importance of algorithmic thinking before translating the solution into MATLAB code.

T: Now, the point of the assignment is that you should be able to translate the mathematical solution into MATLAB code. That is really the point here.

S: Yes, I felt that when I understood the mathematical solution, I had no trouble putting it into MATLAB. It was simply that I had (...) forgotten a bit the length thing there. That f of that minus f of that is delta y, then.

Despite the challenges in understanding the task, the student admitted that the assignment was not about advanced mathematics such as calculus. Nevertheless, the student pointed out that the task is related to a logical way of thinking, which is implicitly associated with CT, but not to calculus or algebra as this excerpt shows:

S: It is not exactly (...) very advanced mathematical functions that we have been working on. It is not quite calculus. That is a lot of plus and minus and logical stuff that isn't (...)

T: Which is not directly related to mathematics?

S: Yes, yes, it is related to very basic mathematics. As many people know, and so it is related to a (...) logical way of thinking that, yes, as one might find (...) may find some of it in mathematics, but it is (...) it does not recall very much, so purely mathematical, about calculus or algebra. Although one can put some formulas into it too, then.

Then the teacher asked the student to elaborate on this issue and how to connect mathematics and the programming constructs of MATLAB.

S: Let us see. Yes, I feel it is on two levels, calculus learning and MATLAB, somehow (...) if I was just working with MATLAB (...) I do not feel like I am getting any better at calculus, because these are two different things.

T: We probably should have had two courses (...) First a basic course in programming, and then a course in (...) because it is very difficult (...) to teach high level mathematics and low-level programming.

S: Yes, I think high-level math, then you need some knowledge in pretty good programming. But our basic understanding of programming is far from it.

The excerpt shows the student considers programming very different from mathematics. There is also no indication that CT could help to bridge the gap between the two subjects. On possible explanation of the disconnection between mathematics and programming is that the student does not clearly see the connection between the mathematical task and the programming solution (length = sqrt((x2-x1)^2 + (2^x2 - 2^x1)^2)). Another explanation is the lack of CT skills which makes it difficult to connect the task with the language constructs of MATLAB. As a result, it seems that there is such a large difference in level between the two subjects that the student was challenged to connect the mathematics task to MATLAB.

DISCUSSION AND PRIMILILARY CONCLUSIONS

The research question addressed in the paper is: *What challenges do students face when engaging in mathematical problem-solving through CT and MATLAB?* The first challenge is the lack of mathematical knowledge, which hindered the student to make sense of the task, have a good mathematical understanding of it, and then develop a problem-solving strategy that can be translated into an algorithm, and implemented using MATLAB. The second challenge is related to the implementation of the algorithmic solution in MATLAB. This was a challenging task as the student struggled to become familiar with MATLAB constructs due to lack of background knowledge and experience with programming. This shows that the minimum requirement to engage in mathematical problem-solving through computational thinking (CT) and programming is a combination of good background knowledge in mathematics and familiarity with the programming language in question in terms of usability and effective implementation of the solution. A third challenge is directed towards the integration of mathematical and programming skills to a coherent whole. To make mathematics interact better with programming, the pedagogical context around first-year undergraduate mathematics courses should be well designed to ensure a smooth integration of CT and MATLAB into the courses in terms of varied and intrinsically motivating tasks that are suited to the students' knowledge level. Moreover, as this study shows, the role of the teacher is still important to assist students in designing algorithms and implementing computational solutions. Clearly, student autonomy cannot be fully expected for novices without good knowledge background in mathematics and familiarities with programming. Hence, the acquisition of CT skills for mathematical problem-solving should consider pedagogical modalities.

In conclusion, the outcome of the study can be summarized as that both lack of mathematical skills and programming experience have led to problems with completing the task. Even though the participating student is representative for the average student enrolled in the course, the study is limited to be generalized from an empirical point of view, but it forms an hypothesis that can be explored in subsequent studies whether mathematical skills really form a prerequisite for programming mathematical problems, perhaps with students from computer science. Other relevant questions are: To what extent is CT compatible with mathematical thinking? What is the importance of CT in bridging the world of mathematics and programming? Nevertheless, two preliminary conclusions can be drawn from the study. Firstly, the relationships between mathematics, CT, and programming languages are quite complex in educational settings. Secondly, engaging in mathematical problem-solving through CT and programming seem to require both good background in mathematics and algorithmic thinking. Future work will use both quantitative and qualitative methods, and a theoretical approach that helps to analyse in more depth the interactions between mathematics, CT, and programming.

REFERENCES

Artigue, M. et al. (2009). Connecting and integrating theoretical frames: The TELMA contribution. *International Journal of Computers for Mathematical Learning 14*, pp. 217-240.

Bokhove, K., & Drijvers, P. (2010). Digital tools for algebra education: Criteria and evaluation. *International Journal of Mathematics Learning 15*, pp. 45-62.

Filho, P., & Mercat, C. (2018). Teaching computational thinking in classroom environments: A case for unplugged scenario. *Conference: Re(s)sources 2018 - Understanding Teachers' Work Through Their Interactions with Resources for Teaching*. Lyon, France.

Hadjerrouit, S., & Gautestad, H.H. (2019). Evaluating the usefulness of the visualization tool SimReal+ for learning mathematics: A case study at the undergraduate level. In: D. Sampson et al. (Eds.), *Learning Technologies for Transforming Large-Scale Teaching, Learning, and Assessment*, (pp. 71-89). Springer Nature Switzerland AG.

Misfeldt, M., & Ejsing-Duun, S. (2015). Learning mathematics through programming: An instrumental approach to potentials and pitfalls. In K. Krainer, & N. Vondrová (Eds.). *Proceedings of CERME9* (pp. 2524-2530). Prague, Czech Republic:

Lie, J., Hauge, I. O., & Meaney, T. J. (2017). Computer programming in the lower secondary classroom: mathematics learning. *Italian Journal of Educational Technology*, *25*(2), pp. 27-35.

Patton, M. Q. (2002). *Qualitative research & evaluation methods*. London: Sage Publications

Pérez-Marín, D., Hijón-Neira, R., Bacelo, A., & Pizarro, C. (2018). Can computational thinking be improved by using a methodology based on metaphors and scratch to teach computer programming to children? *Computers in Human Behavior,* pp. 1-10.

Topalli, D., & Cagiltay, N. E. (2018). Improving programming skills in engineering education through problem-based game projects with Scratch. *Computers & Education*, *120*, pp. 64-74.

Shute, V.J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, pp. 1-10.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A*, *366*(1881), pp. 3717-3725.

Wing, J. M. (2014, January 9). Computational thinking benefits society. *Social Issues in Computing,* 40th Anniversary Blog, University of Toronto. Retrieved from http://socialissues.cs.toronto.edu/index.html%3Fp=279.html