affordances of computational thinking and mathematics. In Online Proceedings of the *Computational Thinking in Mathematics Education Symposium* UOIT, Oshawa (Canada) *October 2017.* Freiman with Broley, Buteau and Vasilyeva.

REPORT FROM WORKING GROUP FOCUSSED ON RESEARCH-BASED UNDERSTANDINGS OF THE INTERPLAY BETWEEN THE AFFORDANCES OF COMPUTATIONAL THINKING AND MATHEMATICS.

Viktor Freiman, *University of Moncton*, with help of Laura Broley, *Concordia University*, Chantal Buteau, *Brock University* and Natalia Vasilyeva, *Concordia University*

May 2018.

Participants in working group (in alphabetical order):

Laura Broley (Concordia University)
Chantal Buteau (Brock University)
Lyn English (Queensland U. of Technology)
Viktor Freiman (University of Moncton)
Anjali Khirwadkar (Brock University)
Amy Lin (Brock University)
Marja Miller (University of Western Ontario)
Joyce Mgombelo (Brock University)
Eric Muller (Brock University)
Natalia Vasilyeva (Concordia University)
Michelle Wilkerson (U. of California Berkeley)

Introduction

The goal of the working group (WG) was to discuss research-based understandings (issues, frameworks, methods, findings) of the interplay between the affordances of computational thinking (CT) and mathematics. Our discussions were centred on the following questions:

- What is learning mathematics versus development of CT?
- What are the common features and differences between the two?
- How could one describe levels of CT development?
- Is CT learnable?
- What are the affordances of CT, as exemplified in specific examples?
- What are the concepts involved?
- How can one assess a student's CT development?
- What are the so-called 21st century competencies and what are their roles in the development of CT?

What follows is a summary of our discussion, which took place over two days of our working group, as well as some a posteriori reflection. We structure the report by focussing on two topics, namely 1) Conceptualization of CT, and 2) Assessment of CT development. We provide a framework by inserting two task examples that were used to exemplify and prompt ideas and emerging questions. We end the report with potential avenues for future research.

1. Conceptualization of CT

Computational thinking (CT) is a relatively new construct, which is becoming the center of educational debates and an increasing number of research publications. If one searches on Google Scholar, more than 20000 entries for the period since 2017 are found. Although not all of the entries are relevant, they do reflect a growing number of studies of CT in a variety of contexts and educational settings. This said, the nature of CT and its place in today's educational systems is not yet clearly understood. To gain a deeper understanding of the complex relationship between mathematics and CT, the working group members were eager to discuss the conceptualization of CT. In other words, we asked ourselves: What is actually meant by "computational thinking"? Moreover, if we take Wing's (2006) position paper as a starting point, we wondered, how could the still relatively unclear concept of CT reach such a significant level of popularity within such a short period of time? After providing a short summary of historical milestones, we briefly discuss some different frameworks that have been developed to make sense of CT.

1.1 Historical Viewpoints

The main arguments given by Wing (2006) in her paper, namely that CT is a 'universally applicable attitude' and, like reading, writing, and arithmetic, is a 'fundamental skill for everyone', have been widely cited (see, Barr, 2011; Caspersen & Nowak, 2013; Lye & Koh, 2014; Orr, 2009). But historically, a claim of the importance of computer programming *for everyone* was already expressed several decades prior. For instance, at the 1981 Third World Conference on Computer Education in Lausanne, during the Keynote Address, Ershov, a Russian computer scientist, called for considering computer programming as 'Second literacy' which, being combined with the traditional (or First) literacy contributes to forming a 'new harmony of human mind' (Ershov, 1981, page 1).

At about the same time, Papert's seminal work 'Mindstorms' (1980) reflected on the first experiments with a computer programming environment called "LOGO". Papert considered the experiments to be still 'too primitive, too limited by technology', but nonetheless serving as the model of 'an object-to-think-with', to use to engage in 'mathematically rich activities which could, in principle, be truly engaging for novice and expert, young and old' (page 182). Already, Papert envisioned the integration of 'computational thinking into everyday life' through 'the most engaging and shareable kinds of activities' not yet accessible at the time of the fist LOGO experiments (Papert, 1980, page 182).

Later, in 1996, Papert again used the term 'computational thinking' when analyzing an approach used by Wilensky and Resnick to build a geometric model of a Rugby game to investigate the question: Where should the kick be taken from to maximize the chance of a score? At the time, Wilensky and Resnick were using the StarLogo environment, which was an extension of LOGO. They wrote a computer program that modelled several rugby players standing, each of them kicking thousands of balls in random directions. The experiment helps to determine which player would be most likely to score the most goals, whereby representing the best kicking point (Wilensky, 1996). This example illustrates how

geometric thinking can use computational thinking 'to forge ideas that are at least as "explicative" as the Euclid-like constructions (and hopefully more so) but more accessible and more powerful' (Papert, 1996).

This short excurse in the history points out some important ideas brought forth by visionaries who, in recognizing the value of CT, proposed novel pedagogical paradigms more than two decades before Wing (2006), whether it involves CT as a new form of literacy (through programming) (Ershov, 1981), or CT as providing learners with the power of computational modelling (Papert, 1981, 1996; Wilensky, 1996). It also highlights the foundations of some essential elements of today's theoretical debates about the nature of CT and how it could afford novel ways of teaching and learning, both in general, and in relation to mathematics in particular. Returning to Wing's (2006) quest for CT as a fundamental skill for everyone, it is interesting to note that she too emphasized how computational methods and models could empower us to 'solve problems and design systems that no one of us would be capable of tackling alone' (p. 33). In fact, the author suggests that the question of what *is* computable is still open for debate, thus reflecting the limitations of our partial knowledge. To gain more insight into the knowledge that does exist surrounding CT, we decided to start reviewing existing frameworks.

1.2 Some Existing Frameworks Concerning CT

Our inquiry was guiding by these questions: How can one study CT? What perspective should one adopt? What is the relationship between CT and mathematics, and particularly mathematical thinking? In the literature, we can find different approaches to conceptualize and to study CT. We focused on two frameworks in particular: i) one developed by Brennan and Resnick (2012) and another linked to Lyn's plenary talk at our Symposium, which focused on ii) the connections between CT and the STEM disciplines.

1.2.1 The Three-Dimensional CT Framework of Brennan and Resnick (2012)

Based on their work with Scratch programming blocks, Brennan and Resnick (2012) define three dimensions of computational thinking: *computational concepts*, *computational practices* and *computational perspectives*.

As key **computational concepts**, the authors give the following list: sequences, loops, parallelism, events, conditionals, operators, and data. *Sequences* mean expressing a particular activity or task in terms of a 'series of individual steps or instructions that can be executed by the computer' (Brennan and Resnick, 2012, p. 3). In some ways, the structure of a sequence is similar to that of a recipe which prescribes steps to be produced. *Loops* provide a 'mechanism for running the same sequence multiple times'. *Events* are defined as 'one thing causing another thing to happen', thus allowing for interactions of the user with the program. P*arallelism* describes 'sequences of instructions happening at the same time'; this is a feature that can be found in many modern computer languages since it provides the possibility of several processes running at the same time. *Operators* offer possibilities for 'mathematical, logical, and string expressions, enabling the programmer to perform numeric and string manipulations'. This includes a wide range of mathematical operations (including addition, subtraction, multiplication, division), functions (like sine and exponents) and operations with strings (including concatenation and

length of strings). Finally, working with *data* 'involves storing, retrieving, and updating values' (Brennan and Resnick, 2012, p. 6).

When defining components of **computational practices**, Brennan and Resnick (2012) used data from interviews with children about the strategies they adopted while developing interactive media. From the variety of such strategies, four main categories were identified: *being incremental and iterative*, *testing and debugging*, *reusing and remixing*, and *abstracting and modularizing*.

First, when considering the process of designing interactive media, Brennan and Resnick (2012) noticed its adaptive nature: there was always the possibility that the initial plan would 'change in response to approaching a solution in small steps' (p. 7). Indeed, the process is *incremental and iterative*, rather than being linear, leading from conception to solution in a straightforward manner. Second, developing strategies for *testing and debugging* is also crucial for the designers of interactive media, as it enables them to solve problems they might encounter 'through trial and error, transfer from other activities, or support from knowledgeable others' (p. 7). Third, a design of a complex interactive media is rarely possible without building on others' work by *reusing and remixing.* This is 'amplified by network technologies that provide access to a wide range of other people's work, like the one generated by *the Scratch online community'* (p. 8). Finally, *abstraction and modularization* is the practice of putting together collections of smaller parts, which helps designers build something large that can be employed at multiple levels, 'from the initial work of conceptualizing the problem to translating the concept into individual sprites and stacks of code' (p. 9).

The third dimension of CT, one of gaining **computational perspectives,** describes the different roles people can play when working with interactive media, roles that go beyond 'pointing, clicking, browsing, and chatting'. Although this role of being consumers is important for learners, it is not sufficient in terms of the development of CT. According to Brennan and Resnick's (2012) framework, working with design tasks encourages more active roles, which emphasize *expression, connection* and *questioning*.

The computation itself becomes a medium for '*self-expression'* of someone's ideas, while giving them a power to create. At the same time, a design activity seems to reinforce social aspects of CT, leading people to become more *connected* by enriching their opportunity of interaction with others (face-to-face or through online community). Finally, computational perspective of *questioning* comes from indicators of young people feeling 'empowered to ask questions about and with technology' hence developing their abilities to 'negotiate the realities of the technological world', rather than feeling some kind of disconnect from the surrounding increasingly technological environment (Brennan and Resnick, 2012, p. 11).

1.2.2 CT Through the lens of a more Integrated STEM Education (English, 2017)

While Brennan and Resnick base their definition of CT on computer programming (Scratch) tasks (or, more precisely, the process of designing interactive media), English (2017) looks at CT through the lens of STEM education, another growing field in teaching and learning theories and practices. With the aim of challenging an isolated technology curriculum component (the "T" in STEM), English grounds her

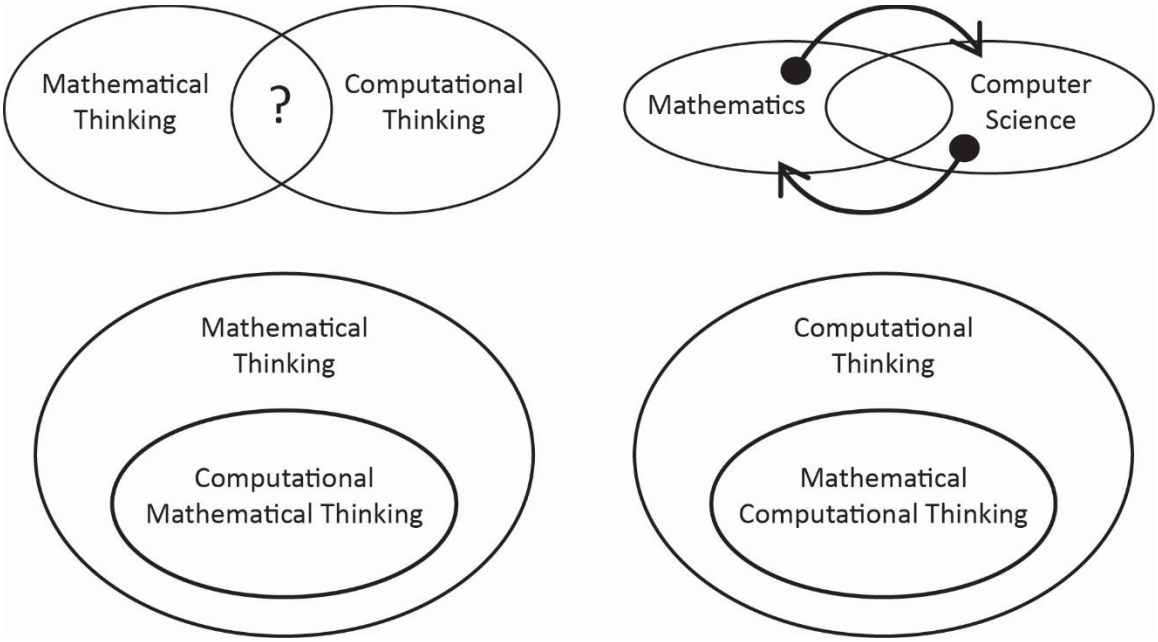
arguments on recent studies by Gadanidis et al. (2016), Schneider et al. (2014), and Weintrop et al. (2016), all of whom discuss CT skills in a broader and more integrated manner.

For instance, Weintrop et al.'s (2016) model of computational practices reflects and represents how many domains in mathematics and science are becoming 'computational endeavours'; they are being enhanced by CT but are also reciprocally enhancing CT through practices involving data, modelling and simulation, computational problem solving, and systems thinking. In a similar vein, Schneider et al. (2014) identify problem solving, modelling, analysing and interpreting data, and statistics and probability as shared features of mathematical and computational thinking.

While arguing for a more equitable focus on each of the STEM disciplines, English (2017) sees value in the approach to CT adopted by Gadanidis et al. (2016). This approach focuses on investigating, depicting and learning 'from cases of "what might be" (or "what ought to be"), to disrupt common conceptions of what CT and mathematics are accessible to young children, how they might engage with it, and how CT affordances may affect mathematics teaching and learning'. In the case of mathematics, this approach can open the door to higher-level mathematics even to a very young learner. Extending these sentiments to the broader scope of STEM-education, English (2017) suggests that equal access to a high-quality STEM education that integrates CT is a 'key issue for future research, not only with respect to socioeconomic, gender, and ethnicity factors, but also in terms of capitalizing on and extending the capabilities of all learners'.

1.3 Debates About the Nature and Role of CT and its Possible Relationship with Mathematics: Denning (2017), diSessa (2018), Pólya (1945), Wing (2008).

At some moment in our discussions arose, unsurprisingly, a general debate on the relationship between computational thinking, computer science, mathematical thinking, mathematics, and so on. This naturally led to the drawing and dissection of several diagrams like the ones shown in Figure 1 below.



affordances of computational thinking and mathematics. In Online Proceedings of the *Computational Thinking in Mathematics Education Symposium*
UOIT, Oshawa (Canada) *October 2017.* Freiman with Broley, Buteau and Vasilyeva.

Figure 1. Examples of Visualizing the Relationship between CT and Mathematics

Wing (2008) proposes a very specific way of perceiving the relationship between CT and other kinds of thinking. She suggests CT to be a complex venture that shares with *mathematical thinking* an approach to dealing with solving problems, with *engineering thinking* strategies for designing and evaluating large, complex systems that operate within the constraints of the real world, and with *scientific thinking* a common understanding of computability, intelligence, the mind and human behaviour. While analyzing computing from an operational point of view, Wing (2008) relates it to answering the question: 'How would I get a computer to solve this problem?' By computer, Wing (2008) means a machine, a human, the combination of a machine and a human, or the combination (e.g. a network) of such computers. From this perspective, the author points at defining abstractions as the 'mental' tools of computing, 'the nuts and bolts' in CT. This includes working with multiple layers of abstraction and understanding the relationships among the different layers. Computing is thus defined as 'the automation of our abstractions' (p. 3718).

diSessa (2018) critiques this view as being too centred on computers and computer science (i.e., "thinking like a computer scientist"). In place of such a view, he offers a bigger picture of CT by shifting the focus to "computational literacy": a 'new, deep, and profoundly influential' kind of literacy that 'will impact all STEM disciplines at their very core'. When reviewing Wing's reference to CT as a problem-solving process, diSessa also points out many similarities between her approach and the significant work done by Pólya back in 1945 within the specific context of mathematics and mathematics education (his stages of decomposition and re-composition, drawing a picture, generalizing, and planning).

Among the different principles diSessa (2018) advocates, one refers to the use of 'epistemologically rich' computer systems with 'yet unrealized consequences for the mathematics we can experience and might teach'. Hence, a deeper understanding of 'what calculation or computation can accomplish, their limitations, their built-in assumptions about the world, signs of failure, and what might be done to contextualize algorithms better, or even change them to suit local needs' could be a promising starting point of real integration of computations in work situations (diSessa, 2018, p. 24). In comparison to Wing, diSessa seems to be promoting a perspective that places a discipline like mathematics at its center and aims to figure out how to make CT work within and for that discipline.

2. PAUSE 1: Thinking with an Example

As the working group discussion progressed, the members felt the urge to consider a specific task in hopes of grounding their ideas within a concrete situation, while avoiding over-generalized statements. Chantal proposed the following example of a CT-based mathematics task where the goal is to design, program, and use an interactive interface to investigate a self-selected or self-stated conjecture about prime numbers by building on a program that checks whether a positive integer is prime.

2.1 Example of a Project Integrating CT into an Undergraduate Mathematics Course

affordances of computational thinking and mathematics. In Online Proceedings of the *Computational Thinking in Mathematics Education Symposium* UOIT, Oshawa (Canada) *October 2017.* Freiman with Broley, Buteau and Vasilyeva.

This example is an assigned task in the first-year *Mathematics Integrated with Computers and Application (MICA) I* course at Brock University (Buteau, Muller, & Ralph, 2015). It is presented at week 3 of the course, when students have learned about designing an interface (by drag and drop), writing/reading from textboxes, variables, loops, and conditional structures. The aim of this first assignment is for students to experience the use of programming (vb.net) in a natural context (i.e., similarly to what a mathematician might do; Broley, Buteau, & Muller, 2017): selecting an open mathematics conjecture or a question of interest (to themselves), and then using programming to investigate whether it may hold true or not. In particular, students are encouraged to use the potential of the computer (e.g., "Students: checking only up to 10,000 might be rather little for the computer!"). They are also asked to summarize their investigation in a written report, including a sample of data collected and their conclusions.

The topic of prime numbers for the first-year undergraduate mathematics students and pre-service teachers provides a relatively easy mathematics topic to tackle (low floor); meanwhile, it may lead to many questions still unknown to mathematicians (high ceiling). Programming-wise, the assignment prompts the students to remix a given program and apply all of the concepts learned thus far.

Below (Figure 2) is a screenshot of one student's interactive interface (Buteau et al., 2016). This student decided to investigate the Opperman's conjecture, which states for any integer n > 1, there is at least one prime number between n(n−1) and n^2, and at least another prime number between n^2 and n(n + 1):

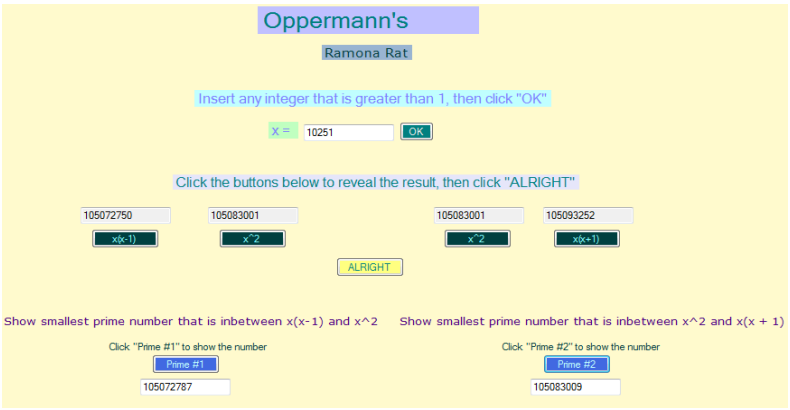


Figure 2. Screenshot of a student's project (programmed in vb.net) who aimed at investigating the Opperman's conjecture.

The student indicated in her written report:

> I wanted to select a conjecture that genuinely interested me, so I selected this one. I thought it was interesting how such a simple process could potentially work for every single integer greater than one, and I wanted to see the legitimacy of the conjecture for myself by using this opportunity of writing a program…[w]hile I was inputting various integers and confirming them, I became more and more surprised that this idea is still a conjecture and not yet a theorem. Even though there is still no proof of it up to this day, my sample data has proven to me that

> Oppermann's conjecture is a great idea, seeing as how I, personally, believe it works. (Buteau et al., 2016, pp.150-1)

2.2. Reflecting on This Example Within Our Working Group

Here are the points or questions brought by the WG participants while reflecting on the example:

- Mathematical contexts can be used to bring CT into teaching and learning thus providing fertile opportunities to conduct meaningful investigations; this can deepen students' understanding of mathematics and the role of CT and affordances of computational tools.
- There are several levels of engagement into programming identifiable with this example: modifying, analysing, creating – teacher chooses what to emphasize.
- Computational paradigm can be used while investigating different topics using CT (statistics; algebra); for example, when drawing a square using CT approach provides with different understanding and variety of affordances,
- Referring to the Instrumental Approach (Rabardel, 1995) – see also the book *Tools and Mathematics* (Monaghan, Trouche, and Borwein, 2016) – when is the tool (programming) becoming an instrument? What form would it take in such a task (or would it need more of such tasks maybe)?
- How can one assess this type of student activity, and that would be an assessment of what exactly: mathematics, CT, CT and mathematics? CT for doing mathematics? Should one use qualitative or quantitative assessment or both? Chantal pointed to a co-authored paper with Eric that examined the assessment of this kind of tasks: (Buteau & Muller, 2016).
- Are there different CT developmental levels? (see section 3.1)
- What is the role of 21st century competences (problem-solving, communication, critical thinking, collaboration, among the others) and soft-skills, and how can these competences and soft-skills be developed in tasks similar to the example?
- CT could be used as tool to help to solve problems and may facilitate certain concepts; in this example, it assisted the student by computing (large) examples of the mathematical conjecture for whatever integer value she inputted.
- Consider collaborative problem solving
- What are the student skills (mathematical; computational; and mathematical-computational skills) involved in such a task?
- What are key aspects in such tasks: producing artifacts, communicating, understanding, …?
- Is CT learnable? Is it teachable? What about self-directed learning?
- A kind of manipulatives – thinking about what students are doing in the activity, i.e., the math that comes out

3. Assessment of CT Development

On the second day, assessment of CT development was at the core of our discussions. We asked: *How can we assess CT development? What can we use?* Four kinds of methodological perspectives were

considered while discussing these two questions, each of which can be classified as: (a) developmental or task and process oriented; and (b) mathematics-specific or general. The following table summarizes the frameworks for assessment discussed by the WG participants. These frameworks are further elaborated below.

Table 1. Frameworks for Assessment of CT

	Mathematical	General
Developmental	Broley, Caron, & St-Aubin's (2017) praxeology-based 5 levels-framework Buteau & Muller (2016) – adaptation of Brennan and Resnick's framework (2012) to computational practices in mathematics	Brennan and Resnik's (2012) three approaches: portfolios, interviews, scenarios

Task & process - oriented	Buteau, C., Muller, E., Marshall, N., Sacristán, A. I., & Mgombelo, J. (2016)'s Instrumental approach (Rabardel)	Bebras competition tasks, Djambong et al, 2018 – 4 components of CT algorithmic, task decomposition, pattern recognition, abstraction (Dolgopolovas, Jevsikova, Savulioniemé & Dagiené, 2015) OCDE PIAAC (2009). Problem-solving in technology-rich environments: problem situation(task), technology, cognitive dimension – levels of complexity Zhong et al. (2016) 3-dimensional framework: directionality (forward or reverse), openness (three types of openness), and process (self-report). Wilkerson, Shareff, Laina, & Gravel (2017) epistemic games (Collins & Ferguson, 1993) learners play when constructing models using the SiMSAM tool

3.1 A six-levels model based on the praxeologies of mathematicians and their students (Broley et al., 2017).

Broley et al. (2017) studied the praxeologies of mathematicians in their research and the praxeologies they propose to their undergraduate mathematics students, in relation to computer programming. To compare the place of programming within the two collections of praxeologies, they identified the following six levels of engagement in the activity. A student or mathematician could:

- L0: Strictly observe the results of a computer program (under the direction of someone else);
- L1: Manipulate the interface of an existing program (in an extracurricular fashion);
- L2: Observe (and analyze) the code of a program;
- L3: Modify existing code to accomplish something new;
- L4: Construct the code of a program, with some elements (e.g., the algorithm) provided; or
- L5: Create a program, including algorithm development, coding, and verification/ validation.

In relation to the highest level of engagement, some mathematicians interviewed by Broley et al. (2017) suggested that programming was the "most mathematical" when it involved complex, deep, or original thinking, as in the search for an efficient algorithm. Echoing section 1.3, some mathematicians also compared programming to the process of solving a problem or constructing a proof.

In the example described in section 2.1, the professor chose to invite his/her students to first create their own programs (L5) and then to modify them to accomplish something new (L3). As such, we could predict that the students are developing some programming techniques that may integrate into their collection of mathematical praxeologies. In comparison, had the students only observed the results of a program developed by their professor (L0), played with such a program at the interface level (L1), and/or analyzed the code of the program (L3), we could predict that their awareness of and skill in programming (during that particular project) may have been developed to a lesser degree, if at all.

Of course, such lower levels of programming engagement may be more appropriate when the aim of the teacher is something other than encouraging the development of students' programming techniques (e.g., challenging their students' intuition, working on their exploration and interpretation of visual output, or exemplifying how (a particular piece of) code works). In fact, it is not difficult to imagine how teachers could use any of the levels to develop aspects of computational thinking in their students, should we adopt the broad definitions discussed throughout this report.

3.2 Three approaches to CT assessment (Brennan & Resnick, 2012)

First approach, project portfolio analysis:

Each member of the Scratch online community has a profile page that displays their creations, as well as other dimensions of participation, such as projects they have favorited and Scratchers they follow

According to the authors (page 15), this approach (conducting content analysis as a means of assessing computational thinking) has several limitations, among them:

- entirely product-oriented, and reveals nothing about the process of developing projects,
- no information about the particular computational thinking practices that might have been employed

Second approach: Artifact-Based Interviews:

Here are examples of questions used in the interviews with members of the Scratch community:

- *(about Scratch, in general) How did you find out about Scratch? What is Scratch? Where do you use Scratch? What do you do with it?*
- *(about the project) How did you get the idea for your project? How did you get started making your project? What happened when you got stuck?*

- *(about the online community) What do you do in the online community? How do you find interesting people and interesting projects? How do you interact with other Scratchers?*
- *(looking forward) What do you dis/like about Scratch? What would you keep, add, change? What are other(non) tech-related things you like to do?*

Some limitations: time consuming, the discussion was limited by what the Scratcher was able to remember and did not typically explore practices in real time.

Third approach: Design scenarios

Three sets of Scratch projects with increasing complexity. Within each set, there were two projects; the projects engaged the same concepts and practices while appealing to different interests. In a series of three interviews, students were presented with the design scenarios. The students were then asked to select one of the projects from each set, and (1) explain what the selected project does, (2) describe how it could be extended, (3) fix a bug, and (4) remix the project by adding a feature.

Limitations: design scenarios are time consuming; the nature of the questions and the use of externally-selected projects may not connect to personal interests and the learner's sense of intrinsic motivation.

Six suggestions for assessment:

- *Supporting further learning*
- *Incorporating artifacts*
- *Illuminating processes*
- *Checking in at multiple waypoints*
- *Valuing multiple ways of knowing*
- *Including multiple viewpoints (pp. 23-24)*

While working within mathematical (developmental) context, Buteau and Muller (2016) have extended Brennan and Resnick's (2012) computational thinking assessment framework into the domain of mathematical inquiry by suggesting a selection of both formative and summative multifaceted assessments to measure student learning through tests in mathematics and programming, prescribed and original programming-based projects and students' written reports. The authors have also adapted Brennan and Resnick (2012)'s *computational practices* to include practices such as 'modelling abstract mathematical concepts into concrete code, designing and coding a mathematical simulation, and engaging systematically in a computer-assisted mathematical inquiry' (Buteau & Muller, 2016) with reference to Weintrop's (2016) four categories for CT in mathematics and science: data practices; modelling and simulation practices; computational problem solving practices; and systems thinking practices.

3.3 Bebras CT competition tasks (Dolgopolovas et al, 2015)

Bebras tasks (Dolgopolovas, Jevsikova, Savulioniemé & Dagiené, 2015) target one or more properties of CT (cognitive skills): Abstraction (AB), Decomposition (DE), Algorithmic thinking (AL) and pattern recognition (PR). Based on the sets of the tasks used in Bebras competitions, Djambong et al. (2018) have built a set of tasks and tried them with Grade 6 and Grade 9 students enrolled in a Broad-Based Technology course. As a result of the questions containing multiple CT properties the authors were

unable to show that the test was able to measure a change in CT development. It was also found that the Bebras questions were unable to provide with exact measurement of each computational thinking cognitive skill.

3.4 Rabardel's Instrumental approach (4 integration levels – tasks – appropriation of the tool)

The instrumental approach (Rabardel 1995, 2002) describes how material or symbolic artefacts can be transformed into instruments through schemes of usage and action appropriated through instrumental genesis (Artigue, 2002). A concept of instrumental integration (Goos and Soury-Lavergne, 2010, cited in Buteau et al, 2016), in its turn, allows for defining four stages of growing technology use in the classroom (Assude, 2007): (a) instrumental initiation (stage 1)—students engage only in learning how to use the technology; (b) instrumental exploration (stage 2)—mathematics problems motivate students to further learn to use the technology; (c) instrumental reinforcement (stage 3)— students solve mathematics problems with the technology, but must extend their technology skills; and (d) instrumental symbiosis (stage 4)—students' fluency with technology scaffolds the mathematical task resulting in an improvement of both the students' technology skills and their mathematical understanding. E.g. Buteau et al. (2016) have used this approach examining a university student experience in a sequel of programming-based mathematics courses.

3.5 OECD-PIAAC Problem solving in Technology-rich environments Framework

The Organisation for Economic Co-operation and Development (OECD) has developed a framework for assessing problem solving in technology-rich environments for its PIAAC (Programme for the International Assessment of Adult Competencies) 2012 international study (for adults 16-65-year-old): four levels of competence – 0 (no experience with technology) to 3 (fluency in problem-solving of complex tasks).

Problem solving in technology-rich environments involves using digital technology, communication tools and networks to acquire and evaluate information, communicate with others and perform practical tasks. The first PIAAC problem solving survey focuses on the abilities to solve problems for personal, work and civic purposes by setting up appropriate goals and plans, accessing and making use of information through computers and computer networks (PIAAC, 2009, p. 9). Problem solving in technology-rich environments is formed by three core dimensions:

i. Cognitive dimensions: Goal setting and progress monitoring; Planning, self-organizing; Acquiring and evaluating information; Making use of information
ii. Technology dimensions: Hardware devices, software applications, commands and functions, representations
iii. Task dimensions: Task purposes (contexts); Intrinsic complexity; Explicitness of problem statement

affordances of computational thinking and mathematics. In Online Proceedings of the *Computational Thinking in Mathematics Education Symposium*
UOIT, Oshawa (Canada) *October 2017.* Freiman with Broley, Buteau and Vasilyeva.

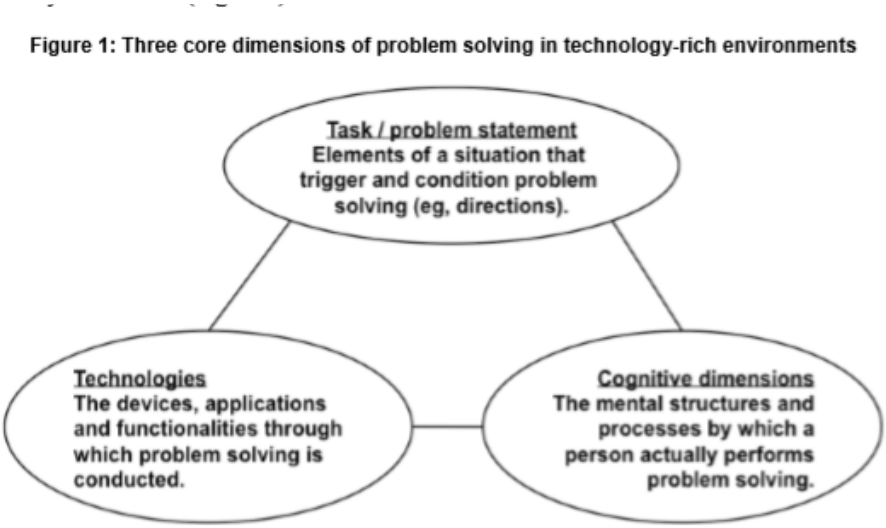


Figure 3. Three core dimensions of problem-solving in technology-rich environments (PIAAC, 2009, p.48)

3.6 Epistemic games in computational model-based inquiry activities (Wilkerson et al, 2018)

Wilkerson et al. (2018) used computational modelling activities based on VanLehn et al.'s (2013) four types of tasks in a context of science education: *model exploration; notational model construction*; *analytic model construction,* and *model-based inquiry*. In their study, the authors have engaged students in constructing models of a target system for which few details were explicitly provided to the student. Students were expected to conduct research by eventually leveraging their intuitive understandings of a target system in order to decide what should be included in a model, and what should be the model's final form (Wilkerson et al, 2018).

With reference to Collins and Ferguson's (1993) theory of epistemic forms and games, Wilkerson et al (2018) considered computational modeling and the role of representational tools and structures in supporting scientific inquiry under two assumptions: (1) a desired result of any epistemic game is the completion of a target epistemic form that satisfies the inquiry and (2) each epistemic game produces a characteristic form; the same form may be produced by more than one game (Collins and Ferguson 1993, cited by Wilkerson et al, 2018).

3.7 Three-dimensional model by Zhong et al. (2016)

Zhong et al (2016) focused on three dimensions in the design of assessment tasks: directionality (forward/reverse), openness (three types of openness), and process (self-report). Based on these dimensions, they design six types of tasks to assess CT: (a) The closed forward task, which is an unfinished task with a defined outcome and a defined process solely; (b) The semi-open forward task, which is an unfinished task with a defined outcome solely and an undefined or open process; (c) The closed reverse task, which is a troubleshooting task with a defined outcome and a defined process solely; (d) The semi-open reverse task, which is a troubleshooting task with a defined outcome solely

and an undefined or open process; (e) The open task with a creative design report, which is a creative task with an open outcome and open process; and (f) The open task without a creative design report, which is a creative task with an open outcome and open process.

4. PAUSE 2: Thinking with an Example

Once more, the working group members sought the context of a specific example within which they could more concretely position their reflections. This example, provided by Natalia, was based on a poster presented at the symposium (Vasilyeva, 2018a).

4.1. Example of a Task in an Undergraduate Mathematics Course on Mathematical Thinking

Natalia, explains: In my study I analyzed a set of MAST 217 (Introduction to Mathematical Thinking) students' solutions of a problem involving investigation. The problem required finding a formula representing the outcome of a potentially infinite haggling process and justifying it (or, from the point of view of the student – making sure the formula is correct). Since the formula was not given (it had to be found), solving the problem required engaging in a sort of small-scale "mathematical investigation".

Here is the text of the problem, as it was posted on the web page of the course:

> John is trying to sell Mark a bike for **a** dollars.
> Mark does not agree on the price and offers **b** dollars ($0< b < a$).
> John does not agree on this price but comes down to $(a + b)/2 = 1/2\ a + 1/2\ b$.
> Mark responds by offering $(b + (a + b)/2)/2 = 1/4\ a + 3/4\ b$.
> They continue haggling this way, each time taking the average of the previous two amounts.
> On what amount will they converge? Express the amount in terms of a and b.
>
> Explain your reasoning and justify your response.
> Have you tried to verify your answer? If yes, how?

This assignment was given to students near the end of the course, in the 10th week of classes (the course lasts 13 weeks). By this time, related topics such as geometric sequences and series, the notion of the limit of a sequence, the theorem that increasing (decreasing) and bounded above (below) sequences are convergent and examples of convergent sequences related to computational algorithms were covered. Solving this non-routine problem required some mathematical investigation. It could be solved empirically by observing the numerical results and making a conjecture about the limit of the sequence. Students could try to verify the conjecture by drawing a diagram, by observing a link between the sequences involved in the amounts and geometric series or by using other means. Even though the haggling problem did not demand the formal construction of a proof, we expected students to attempt to convince themselves and others.

4.2 Natalia's Comment and Reflection on the Example

Natalia explains: In my master's thesis (Vasilyeva, 2018b), I analyzed students' solutions of the haggling problem. At least 26 students (out of 32) systematically explored examples to solve the haggling problem. I distinguished two main approaches in attacking the haggling problem: computational (7 solutions) and analytical (19 solutions).

Computational approach

By "computational approach" I mean arithmetical exploration of many concrete amounts or coefficients (usually using a computer software). Thus, students carried out numerical experiments to show that a sequence of amounts converges. For example, one of the students designed a code in Python and used several pairs of values for *a* and *b* in his calculations. Other students calculated coefficients directly. Observation of numerical results did not always lead to the correct answer. Only three out of 7 students concluded that the final amount is $\frac{1}{3}\ a\ + \frac{2}{3}\ b$ and their justification can be accepted as a *generic example proof* (Yopp at al., 2015).

Analytical approach

"Analytical approach" is assumed in my study to refer to an algebraical exploration of the sequence of coefficients, searching for patterns, and then using other heuristics, such as analogy, deduction, structure recognition, or algebraic manipulations.

A majority of students (19) tried to solve the haggling problem by using this approach: they looked for a formula to represent a sequence of prices. Most of them thought about finding the limit of this sequence. Moreover, some of them tried to show that the sequence is bounded and decreasing. However, only 8 students arrived to the correct answer and provided some deductive arguments to make their solutions count as mathematical proofs (with gaps and minor mistakes). It is worth noting that 7 students among those 8 checked their conjecture by using concrete examples. In other words, they combined analytical and computational approaches.

The findings from my study suggest that it is possible to integrate computational thinking in mathematics education. For example, a computational simulation of the haggling problem can be seen as a first step in making and testing conjectures.

5. Potential Avenues for Future Research – Proposal by Laura, Amy, and Joyce

As the meeting was drawing to an end, we decided to break into small groups to brainstorm potential avenues for future research. Each small group was asked to come up with a potential research project, including a question, a theoretical framework, and a methodology. Then we shared and discussed our ideas within the whole group. As a way to conclude our report, we next summarize the proposal from the small group involving Laura, Amy, and Joyce.

It is true that "computational thinking" is a new buzzword; but, as mentioned at the beginning of this report, the concepts related to the word have been discussed for decades. This small group wondered about the need for a comprehensive literature review that goes beyond the most commonly cited and

most recent frameworks: a sort of deep dive in the history of mathematics education literature to explore the different ways in which computational thinking has and could be conceptualized.

This led us to formulate the research question: How could we conceptualize computational thinking in mathematics education (research)? The word "could" is meant to highlight the perhaps necessary existence of a multiplicity of perspectives that may be useful to varying degrees depending on the goals of its user; for example, depending on the specific aims of a researcher. The theoretical framework to support answering such a question could involve a selection of different frameworks discovered during the literature review mentioned above. During our small-group discussion, we noticed that each of us defines "computational thinking" in a slightly different way, whether as part of a person's praxeologies (Laura), as an artefact or tool (Joyce), or as one of many modalities of learning (Amy). To answer the research question, we posed and ultimately show what different conceptualizations of computational thinking may (or may not) bring to research. Further, we envisioned a study where we analyze the same mathematical task from our three differing perspectives.

When sharing our idea with the WG, it was conjectured that different perspectives on "computational thinking" may even correspond to different kinds of tasks. Hence, the methodology we proposed could include not only an analysis of an existing mathematical task, but also proposed modifications to such a task based on each perspective. It was also noted that such an exploration of the existence and usefulness of different conceptualizations may release the tension we face when we stubbornly continue the struggle for one clear and comprehensive definition for "computational thinking", especially in relation to "mathematics" or "mathematical thinking". It reminded us that the models we use in research can simply be machines for supporting or producing certain understandings, rather than providing accurate, complete representations of a reality that exists outside of us.

References

Artigue, M. (2002). Learning mathematics in a CAS environment: the genesis of a reflection about instrumentation and the dialectics between technical and conceptual work. International Journal of Computers for Mathematical Learning, 7(3), 245–274.

Assude, T. (2007). Teachers' practices and degree of ICT integration. In D. Pitta- Pantazi, G. N. Philippou (Eds.), Proceedings of the fifth congress of the European Society for Research in Mathematics Education (pp. 1339–1348). Larnaka: Department of Education, University of Cyprus.

Barr, D., Harrison, John, & Conery, L. (2011). Computational Thinking: A Digital Age Skill for Everyone. *Learning & Leading with Technology*, 38(6), 20-23.

Brennan, K., & Resnick, M. (2012). Using artifact-based interviews to study the development of computational thinking in interactive media design. Paper presented at annual American Educational Research Association meeting, Vancouver, BC, Canada.

Broley, L., Buteau, C., & Muller, E. (2017). (Legitimate peripheral) computational thinking in mathematics. *Proceedings of the Congress of European Society for Research in Mathematics Education* (pp. 2515-23), Dublin, Ireland.

affordances of computational thinking and mathematics. In Online Proceedings of the *Computational Thinking in Mathematics Education Symposium* UOIT, Oshawa (Canada) *October 2017.* Freiman with Broley, Buteau and Vasilyeva.

Broley, L., Caron, F., & St-Aubin, Y. (2017). Levels of Programming in Mathematical Research and University Mathematics Education. *International Journal of Research in Undergraduate Mathematics Education, 4*(1), 38–55.

Buteau, C., Muller, E. (2016). Assessment in Undergraduate Programming-Based Mathematics Courses. *Digital Experiences in Mathematics Education 3*(2),97–114.

Buteau, C., Muller, E., Marshall, N., Sacristán, A. I., & Mgombelo, J. (2016). Undergraduate mathematics students appropriating programming as a tool for modelling, simulation, and visualization: A case study. *Digital Experience in Mathematics Education*, *2*(2), 142-156. doi:10.1007/s40751-016-0017-5

Buteau, C., Muller, E., & Ralph, B. (2015). Integration of programming in the undergraduate mathematics program at Brock University. In *Online Proceedings of Math+Coding Symposium*, London, ON. Retrieved from http://researchideas.ca/coding/docs/ButeauMullerRalph-Coding+MathProceedings-FINAL.pdf

Caspersen, M.E., Nowak, P. (2013). Computational Thinking and Practice — A Generic Approach to Computing in Danish High Schools, Proceedings of the 15th Australasian Computing Education Conference, ACE 2013.

Collins, A., & Ferguson, W. (1993). Epistemic forms and epistemic games: Structures and strategies to guide inquiry. Educational psychologist, 28(1), 25-42. doi: 10.1207/s15326985ep2801_3

diSessa, A. A. (2018). Computational Literacy and "The Big Picture" Concerning Computers in Mathematics Education. *Mathematical Thinking and Learning*, *20*(1), 3-31.

Djambong, T., Freiman, V., Gauvin, S., Paquet, M., & Chiasson, M. (2018). Measurement of Computational Thinking in K-12 Education: The Need for Innovative Practices. In *Digital Technologies: Sustainable Innovations for Improving Teaching and Learning* (pp. 193-222). Springer, Cham.

Dolgopolovas, V., Jevsikova, T., Savulionienė, L., & Dagienė, V. (2015). On Evaluation of computational thinking of software engineering novice students. In *Proceedings of the IFIP TC3 Working Conference "A New Culture of Learning: Computing and next Generations* (pp. 90-99).

English, L. (2017). Advancing Elementary and Middle School STEM Education. *International Journal of Science and Mathematics Education,* 15(1), 5-24.

Gadanidis, G., Hughes, J.M., Minniti, L. & White, B.J.G. (2016). Computational thinking, grade 1 students and the binomial theorem. *Digital Experiences in Mathematics Education*. Advanced online publication. doi: 10.1007/s40751-016-0019-3

Goos, M., & Soury-Lavergne, S. (2010). Teachers and teaching: Theoretical perspectives and classroom implementation. In C. Hoyles & J.-B. Lagrange (Eds.), ICMI Study 17, technology revisited, ICMI study series (pp. 311–328). New York: Springer.

Ershov, A. P. (1981). Programming, the second literacy. In "Computers in Education, Proc. IFIPTC-3, 3rd World Conf. Comput. Educ. Part 1, pp. 1–7.

Lye S. Y., Koh J. H. L. (2014). Review on teaching and learning of computational thinking through programming: what is next for K-12? *Comput. Human Behav. 41*, 51–61. 10.1016/j.chb.2014.09.012

Orr, G. (2009). Computational thinking through programming and algorithmic art. SIGGRAPH2009, New Orleans, LA.

Monaghan, J., Trouche, L., & Borwein, J. M. (2016). *Tools and mathematics*. Berlin: Springer International Publishing.

Papert, S. (1980) Mindstorms: Children, Computers, and Powerful Ideas. Flammarion.

Papert, S. (1996), An Exploration in the Space of Mathematics Educations, *International Journal of Computers for Mathematical Learning*, Vol. 1, No. 1, pp. 95-123.

Pólya, G. (1945). *How to Solve It*. Princeton University Press. ISBN 0-691-08097-6

PIAAC Expert Group in Problem Solving in Technology-Rich Environments (2009). PIAAC Problem Solving in Technology-Rich Environments: A Conceptual Framework*. OECD Education Working Papers, No. 36*. OECD Publishing. http://dx.doi.org/10.1787/220262483674

Rabardel, P. (1995/2002). Les hommes et les technologies, une approche cognitives des instruments contemporains. Paris: Armand Colin

Schneider, C., Stephenson, C., Schafer, B. & Flick, L. (2014). Exploring the science Framework and NGSS: Computational thinking in the science classroom. *Science Scope*, November, 10–15.

VanLehn, K. (2013). Model construction as a learning activity: A design space and review. Interactive Learning Environments, 21(4), 371-413. doi: 10.1080/10494820.2013.803125

Vasyljeva, N. (2018a). Computational and Analytical Approaches in Solving Problems Requiring Mathematical Investigation. Poster presented at Computational Thinking in Mathematics Education Symposium, University of Ontario Institute of Technology, Oshawa (Canada), October 2017.

Vasilyeva, N. (2018b). *How do students know they are right and how does one research it?* (Master's thesis, Concordia University) Retrived from https://spectrum.library.concordia.ca/983419/7/Vasilyeva_MTM_S2018.pdf

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L. & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology, 25*, 127–147.

Wilensky, U. (1996). Modeling rugby: Kick first, generalize later?. *International Journal of Computers for Mathematical Learning*, *1*, 124-131.

Wilkerson, M. H., Sharref, B., Laina, V., & Gravel, B. (2018). Epistemic gameplay and discovery in computational model-based inquiry activities. Instructional Science, 46(1), 35-60.

Wing, J. M. (2006). Computational thinking. *Commun. ACM* 49(3), 33-35. DOI: https://doi.org/10.1145/1118178.1118215

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical transactions of the royal society of London A: mathematical, physical and engineering sciences*, *366*(1881), 3717-3725.

affordances of computational thinking and mathematics. In Online Proceedings of the *Computational Thinking in Mathematics Education Symposium* UOIT, Oshawa (Canada) *October 2017.* Freiman with Broley, Buteau and Vasilyeva.

Yopp, D., Ely, R., & Johnson-Leung, J. (2015). Generic example proving criteria for all. *For the Learning of Mathematics*, 35(3), 8-13.

Zhong, B., Wang, Q., Chen, J., & Li, Y. (2016). An exploration of three-dimensional integrated assessment for computational thinking. *Journal of Educational Computing Research, 53*(4), 562–590. http://dx.doi.org/10.1177/0735633115608444