DOI 10.4171/8ECM/40

Computational/algorithmic thinking in school mathematics

Djordje M. Kadijevich

Abstract. As a result of the globalization and internationalization of the mathematics curriculum, there is, for example, a rapidly developing interest in including computational/algorithmic thinking (CT/AT) in mathematics education. After briefly presenting an emerging educational context regarding the application of CT, this contribution first examines critical issues of CT/AT concerning the notion of CT/AT, the state of CT/AT-oriented educational research, and the integration of CT/AT in the school mathematics curriculum. Then, it presents how CT might be cultivated through data practice and, to this end, data modeling using interactive displays is applied. The contribution ends with a summary of the issues examined and implications for research and practice. This contribution is an extended version of a keynote talk delivered at the symposium "Mathematics in Education."

1. Introduction

Today, technology is increasingly used in all areas of work and life. To practice problem solving with technology successfully, apart from applying disciplinary reasoning (i.e., reasoning applied in the particular discipline such as mathematics), students need to apply computational thinking (CT), which, in short, denotes reasoning processes used in solving problems when solutions are represented in forms that can efficiently be performed by computers [63]. CT clearly involves some degree of algorithmic thinking (AT) that is applied in the work with algorithms. This is because algorithms are used to describe, in a precise manner, which steps (e.g., calculations, visualizations, logical inferences) need to be taken and in what order, to solve the problem under consideration (e.g., [9]).

Recent educational research evidences a growing number of researchers and educators who call for cultivating CT, not only in teaching computer science (informatics) but also in teaching other subjects, such as mathematics and statistics. To illustrate this state, the role of CT in three international projects is summarized below.

2020 Mathematics Subject Classification. Primary 97C70; Secondary 97B99, 97C30, 97K40, 97P99.

Keywords. Algorithmic thinking, computational thinking, data modeling, interactive displays, school mathematics.

- Students' knowledge and skills regarding computer and information literacy have been evaluated worldwide using International Computer and Information Literacy Study carried out by the International Association for the Evaluation of Educational Achievement (https://www.iea.nl/). In a 2018 study cycle, students' CT was assessed for the first time, using tasks that required them to analyze problems, divide these into subproblems, and then find steps that lead to their solutions [18].
- The Organization for Economic Co-operation and Development (https://www.oecd.org/) has evaluated educational achievements in reading, mathematics, and science worldwide using the well-known PISA study (Program for International Student Assessment). In its current 2021 cycle [48], students' CT is assessed for the first time by including it in the steps of mathematical modeling that have already been used in this study (e.g., in the step of employing, one may apply technology to find exact/approximate solutions).
- The growing need for experts in the field of data science, i.e., for the so-called data scientists (e.g., [34]), demands the development of skills required by such a profession, including CT. Within an international project named International Data Science in Schools Project (http://www.idssp.org/), the content of the high school subject on data science has been developed, aiming at the integration of computational and statistical thinking. The development of various resources to support teachers in the realization of this subject is planned [23].

In the remaining text of this contribution, we first consider critical CT/AT issues concerning their definition, state of research, and curricular integration. This consideration is primarily based on a recently published encyclopedia entry [56]. Then we present a way to cultivate CT through data practice to support the position that other learning practices (not only programming, as is often assumed) could be used to develop CT/AT. This presentation is mainly based on two recently published contributions: a chapter in an edited book [31] and an entry in an encyclopedia [33]. The contribution concludes with a summary of the issues examined and implications for research and practice.

2. Critical CT/AT issues

This section comprises three subsections. The first clarifies the notion of CT/AT, the second summarizes the current state of CT/AT-oriented educational research, whereas the third examines the integration of CT/AT in the school mathematics curriculum.

2.1. Definition

As mentioned in Section 1, algorithms are used to describe, in a precise manner, which steps (e.g., calculations, visualizations, logical inferences) need to be taken and

in what order to solve the problem under consideration. It is usually assumed that the notion of AT is used to describe reasoning processes applied in work with algorithms, which require their user/developer to comprehend, test, evaluate, correct, improve, or design them. CT clearly involves work with algorithms because to apply computers in problem solving, problem solutions have to be represented in an algorithmic fashion ("do this, do that, in the following order") using forms that are recognized by the computer programs applied.

CT is widely spread and increasingly used in educational research. As CT is based on AT, it may be expected that most researchers agree on what the notion of CT stands for. This is not true, however, because a widely accepted definition of CT is lacking [45].

Various CT definitions have been proposed in the literature. To define CT, researchers have referred to its main facets, practices, concepts, components, and dimensions, and examined them within the specific educational context. This context ranged from specific subject area(s), such as programming or STEM (Science, Technology, Engineering, and Mathematics) education, to a general educational setting such as K-12 subjects [56]. In a high school STEM context, for example, CT may be applied in (and thus cultivated by) various learning practices, including *data practices* (e.g., preparing data and visualizing them), *modeling and simulation practices* (e.g., building and using computational models), and *computational problem-solving practices* (e.g., programming, troubleshooting) [61].

A recent review showed that to clarify the notion of CT, researchers have used and combined entities of different sorts. Most of these entities refer to thinking processes (e.g., abstraction), problem solving methods (e.g., simulation), standard implementation practice (e.g., debugging), and general skills (e.g., technology solution design) [43]. To make progress in CT/AT-related research, researchers may focus on similarities in proposed CT definitions rather than on their differences. CT entities are still common in many of these definitions, such as decomposition (i.e., breaking a problem down into subproblems), abstraction (i.e., making general statements concerning particular examples), and algorithms [55]. These three entities are highly relevant to mathematics learning through programming because during this learning, CT makes use of decomposition, abstraction, pattern recognition, and algorithmic thinking [22].

Regarding CT's main entities (cornerstones), instead of decomposition, abstraction, pattern recognition, and algorithmic thinking, researchers may consider decomposition, abstraction, algorithmization, and automation. There are two reasons for this conceptual shift. Firstly, pattern recognition may be viewed as an instance of abstraction and generalization [53]. Secondly, CT relies on automation of calculations, i.e., using computers that apply certain computational models; a human may formulate a problem solution, but this solution is primarily carried out by a computer not by a human [39]. Although the term CT was coined more than forty years ago [49], it can

```
Known facts about triangle
side(a).
side(b).
angle(alpha).
angle(beta).
opposite(alpha, a).
opposite(beta, b).
greater_side(a, b).

New fact added
greater_angle(alpha, beta).

Discovered rule afterwards
greater_angle(X, Y) :- angle(X), angle(Y), opposite(X, X1), opposite(Y, Y1),
    side(X1), side(Y1), greater_side(X1, Y1).
```

Figure 1. Rule discovery.

be said that CT has been used for centuries to design computational procedures and computing machines to automate them; to formalize a computing procedure, mathematicians have usually described its steps using an algorithm [12], which may also deal with (frequently overlooked) model of computation (to be) applied [11].

If we accept the conceptual shift mentioned above, AT main entities (cornerstones) might then be defined as decomposition, abstraction, and algorithmization, and it is precisely the application of automation that separates AT from CT. This means that AT is not equal to CT but is rather included in it [28, 32]. Interestingly, mathematics educators/researchers may prefer to use AT even when technology is applied, whereas computer science educators/researchers may prefer to use CT even when technology is not used (see [6, 42] for this preference), which may be the result of distinguishing (securing) the position and role of AT/CT in their discipline.

Although the relevance of automation to CT cannot be questioned, its importance to the development of mathematical thinking might be lower than that of other CT cornerstones (decomposition, abstraction, algorithmization), which were also critical learning activities in Pólya's [50] approach to problem solving [13]. Such a state that puts automation in the CT background was found in a recent study involving twenty-five mathematics and computer science experts. They considered CT aspects in mathematics courses and reached a much lower consensus for applying automation than that for using decomposition, abstraction, and algorithmic thinking [36].

It might be that the role of automation is devaluated in general for a few reasons but this position is questionable. By applying abstraction (e.g., through selecting variables), we provide building blocks for automation to be carried out. However, computer programs may not only provide means to support abstraction (e.g., through the work with classes in object-oriented programming), but also they may do abstraction themselves as well. Think about a computer program that enables rule discovery (e.g., [10]). Figure 1 presents facts that may be needed to support the discovery of the well-known rule that says that a greater angle of a triangle is opposite a greater side.

A similar bi-directional relationship holds true for decomposition. By applying decomposition (through identifying substantial or relational sub-problems [52]), we also provide building blocks for automation to be carried out. Regarding the contribution of automation to decomposition, consider computer programs (the so-called expert systems) that instruct their users which sub-problems to solve first and how to use their solutions in order to solve the initial problems (e.g., [24, 25]). A video available at https://www.mi.sanu.ac.rs/~djkadij/FRA20.avi presents the work with such a system concerning problems on multiple proportion (e.g., if three workers repaired 6 windows working 8 hours, how many workers are needed to repair 9 windows working 6 hours?)

2.2. State of research

The notion of CT originated from learning mathematics with technology, i.e., the work with Turtle Geometry through LOGO programming more than forty years ago [49]. Since 2000, due to the elaboration of CT done by a computer scientist Wing [63, 64], this notion has been mostly used by computer science experts, who link it with computer science topics, mostly programming [21]. As a result, during the previous decade, CT has become a critical curricular component in computer science (informatics) education in a number of countries worldwide (e.g., [59]).

Due to limited research on CT in mathematics learning, CT has not had a similar status in mathematics education. Studies on linking CT and learning mathematics in an explicit way are rather rare, and in doing so they mostly refer to areas that are traditionally connected to programming, including numbers and operations, algebra, and geometry. There are, of course, other areas suitable for this linking, such as functions, probability, and statistics. Functions might be explored through modeling, probability through simulations, whereas statistics could better be understood through data analysis [21]. In solving problems, these areas are often combined. Data analysis may, for example, reveal the most probable distribution of the values of a particular variable, and this distribution might be used to build a mathematical model with simulation.

To pursue these explorations successfully, appropriate learning paths need to be followed. Such paths have been proposed outside the mathematics education community, such as an understand-debug-extend path [8], or a use-modify-create path [40], which, when combined, may result in the following path: use problem solutions (to understand or evaluate them) – modify problem solutions (to debug or extend them) – create problem solutions, i.e., develop problem solutions from scratch. Mathematics educators have proposed CT pedagogy for the work with various conceptual or digital objects in the classroom [38]. The proposed pedagogy assumes that this work makes use of four overlapping activities: *unplugging* (not using computers), *tinkering* (dividing existing objects into their components and changing or modifying these components), *making* (constructing new objects), and *remixing* (producing

new objects through the appropriation of existing objects or their components). As an example of unplugging, consider sorting mathematical expressions. Tinkering is applied when the content of a spreadsheet is modified, whereas remixing is practiced when a dashboard (a set of interactive reports) is created through combining and modifying existing interactive reports. Although these four activities (unplugging, tinkering, making, remixing) are present in the combined learning path mentioned above, this pedagogy can be applied without using computers, which opens the question "What is actually being developed when computers are not used: CT or (nevertheless) AT only?"

AT is a central activity in mathematics. Although AT is widely practiced in mathematics classes (though mostly implicitly), research on AT in mathematics learning is also limited. There are, however, several studies whose valuable findings may contribute to fostering both AT and mathematics learning. It was found, for example, that procedural knowledge rich in connections could be developed through designing and implementing procedures and algorithms [42]. AT may also be used to develop conceptual knowledge representing a deeper conceptual understanding when a special case of an algorithm in general, or a formula in particular, is considered in detail to ask advanced questions about its result [1]. In other words, AT may contribute to developing and relating procedural and conceptual mathematical knowledge. When AT is supported by technology (i.e., when CT is practiced in our terms), it is important to understand in what ways mathematics learning could be mediated by technology [14], especially in developing and relating these two types of mathematical knowledge (e.g., [2, 30]). To develop AT gradually, the following learning path (derived from the combined path mentioned above) could be applied: consider formulas, procedures, and algorithms given (to understand or evaluate them) – modify formulas, procedures, and algorithms given (to debug or extend them) – create formulas, procedures, and algorithms, i.e., develop them from scratch. Furthermore, as in case with CT, the activities comprising this path are not realized separately but, as a rule, overlap each other.

Although research on CT/AT in mathematics learning is limited at present, it seems to be a growing research area as evidenced, for example, with the inclusion of CT in PISA 2021 [48]. Also, in 2021, research and practice regarding CT/AT were explicitly represented (probably for the first time) at an international congress on mathematical education. In particular, at the 14th International Congress on Mathematical Education" (ICME-14, https://www.icme14.org), there was a topic study group titled "Teaching and learning of programming and algorithms" (TSG-14) and a discussion group titled "Computational and algorithmic thinking, programming and coding in the school mathematics curriculum: Sharing ideas and implications for practice" (DG-1), whose participants emphasized the importance of fostering CT/AT in mathematics education. This might be done through problem solving using

a CT/AT lens described in this contribution. Such an approach would result in more focused (and explicit!) instruction on AT and its core components (decomposition, abstraction, algorithmization), possibly supported by particular computer programs. As assumed by a model of mathematical thinking based on the triad abstraction-modeling-problem solving [15], these components denote critical activities applied in mathematics learning.

2.3. Curricular CT/AT integration

This subsection comprises three parts. The first explains the rationale for this curricular CT/AT integration, the second summarizes different models of integrations applied worldwide, while the third examines various educational implications of this integration.

2.3.1. Rationale for integration. More and more workplaces require specialized knowledge based on the use of modern information-communication technology (ICT); tens of millions of specialists with this knowledge are needed today worldwide [4]. Among them data scientists are particularly important, whose competencies (e.g., [3]) are essentially supported by CT/AT. An increasing demand to (better) prepare students for a range of ICT-based jobs (with many future ones unknown at present) clearly provides a good rationale for the inclusion of CT/AT in school mathematics. There is another good rationale for this inclusion. Due to an increasing reliance on computations in scientific inquiry (e.g., [17]), students should learn how to solve problem with technology for the development of their mathematical thinking. To this end, they should act as information-processing agents (e.g., [64]). Although these two rationales clearly represent different perspectives (a societal one *vs* a professional one [7]), they are not separated, obviously influencing each other. Note that an extensive rationale for including CT in school mathematics (at least for senior high school students) was elaborated in a discussion paper developed by four mathematical and computer science academies in France by using the following arguments: (1) CT is becoming increasingly embedded in university courses in mathematics; (2) certain areas such as graphs, combinatorics, and logic could be used to establish creative interfaces between mathematics and computer science; (3) CT can strengthen students' mathematical development [19]. This paper also contains a number of examples that can be used to foster creative interfaces between mathematics and informatics (computer science).

2.3.2. Models of integration. Various models of the integration of CT in the school mathematics curriculum have been applied worldwide. Let us provide some examples. To integrate CT/AT across different school subjects, a cross-curriculum model may be applied like in Finland. If this integration is realized within the curriculum of an information technology (IT) subject, an IT model may be in use like in Australia

and England. CT/AT may be integrated in mathematics and other school subjects in several grades gradually, meaning that a gradualist model is being applied, like in Japan where the focus is on programming thinking not on CT/AT. Finally, the integration may be realized within a new school subject, like in France (subject *Algorithmique et Programmation* in the middle grades taught by mathematics and IT teachers) and Australia (subject *Algorithmics* in the senior high school). Clearly, although CT/AT integration has often been realized within one or several existing subjects, it could be done within new subjects as well. Although all these models remain unexamined and are by and large untested, certain *pros* and *cons* can be identified. For example, the cross-curriculum model might be implemented in a shallow way; in the IT model, teachers may focus more on using technology than on mathematical connections; the gradualist model allows time for teacher preparation, but creating interfaces between school subjects with entrenched boundaries would be challenging; a separate subject, especially if taught by mathematics and IT teachers, can provide opportunities for exploring interfaces between mathematics and computer science, but may, at a higher educational level, require rich prior experience with CT/AT [56]. For a thorough evaluation, the curricular integration of CT may be examined in terms of critical curricular components (e.g., goals, content, materials, forms of teaching, student activities, assessment [47]). Note that a detailed integration of CT in the school mathematics curriculum is planned in Australia. The new Australian F-10 curriculum for mathematics (from Foundation to Year 10) calls for the application of CT in problem solving, and gives examples and instructions of doing that from Year 4 to Year 10 [5]. In this document, the phrase *computational thinking* occurs almost forty times (e.g., Year 10: "apply computational thinking to model and solve algebraic problems graphically or numerically"). In July 2021, the status of this document was "waiting for approval."

2.3.3. Educational implications. Due to technological advances, computational mathematics has been increasingly used in research mathematics; there are great number of respectable research publications with the words *computational* and *mathematics* in their titles, whose authors, stated briefly, primarily examine various algorithms carried out by computers. Such a reliance on computations has changed the practice of scientific inquiry in which "together with theory and experimentation, a third pillar of scientific inquiry of complex systems has emerged in the form of a combination of modeling, simulation, optimization, and visualization" [17, p. 2]. Hence, the development of CT/AT in mathematical classes should cultivate such an inquiry by applying different kinds of practice, such as those already mentioned *data practices* (e.g., preparing data and visualizing them), *modeling and simulation practices* (e.g., building and using computational models), and *computational problem-solving practices* (e.g., programming, troubleshooting) [61]. To this end, instruction may relate

(integrate) content, technology, and pedagogy through, for example, identifying relevant CT/AT practice(s) for each curriculum strand content descriptor and computer tool available (e.g., [51]).

Although classroom practice should be different from disciplinary practice (increasingly using computation to support experimentation, approximation, conjecture testing, visualization, and other aspects of mathematicians' work), the latter should inform the former and help design it [41]. Hence, students may in general use CT/AT to define (construct) objects, identify their possible properties (of algebraic, geometric, or other nature), and verify these properties. Furthermore, like mathematicians who apply computation to find approximate solutions to intractable problems, students may use CT/AT to approximate solutions of mathematical models that cannot (easily) be solved in the context of school mathematics (e.g., [37]). Regarding the use of algorithms in particular, it may, for example, support students to (1) unpack concepts and procedures, (2) identify the mathematical structure of a given problem and generalize its solution, (3) familiarize themselves with modeling, optimization, operations research, and experimental mathematics, and (4) generate examples of problems for which the given algorithm works or does not work [56].

Some readers may insist on the position that despite the fact that various CT/AT-based practices might be applied to solve a variety of task types, CT/AT should nevertheless be promoted primarily through programming. The following examples may help these readers make this position less strict. In preparing these examples, it was supposed that we apply CT whenever we recognize aspects of computations in problem solving and deals with them in appropriate ways by using tools and techniques from computers science [57]. Regarding this computing support, the examples make us of Wolfram Alpha (https://www.wolframalpha.com/). Of course, the use of computing support in general may generate various learning challenges and appropriate didactic treatments need to be applied to alleviate them (e.g., [20, 26]).

Example 1. To determine the greatest common divisor, one can simply use a built-in command `gcd`, such as `gcd(24,16)` that yields 8. Another way to do this is to apply a four-step-approach: (1) find the set of the first number divisors, (2) find the set of the second number divisors, (3) determine the intersection of these sets, and (4) find the maximum value in the intersection set (with each step supported by an important algorithm). To combine these steps, clearly in an algorithmic fashion, use the following commands: `Max[intersect[divisors(24),divisors(16)]]`.

Example 2. To discover functional dependence that connects two arrays of natural numbers, we may apply a curve fitting approach with perfect fit. The number of diagonals in a triangle, quadrilateral, and pentagon are 0, 2, and 5, respectively. If the number pairs $(3, 0)$, $(4, 2)$, $(5, 5)$ are fitted with a quadratic model, the following dependence is found $0.5x^2 - 1.5x$, and this fit is perfect because $R^2 = 1$. When this

dependence is factorized, the result is $0.5(x-3)x$, directing students what key elements to consider: the role of x is clear, but why 0.5 and $x-3$ are included? Relevant commands are `quadratic fit{3,0},{4,2},{5,5}` and `factor(0.5x^2 - 1.5)`.

As AT is critical to the processes of conjecturing and proving, the development of algorithms may be connected with these processes. There are some areas of discrete mathematics (e.g., combinatorics, graph theory) that are particularly suitable for fostering creative interfaces between mathematics and computer science through exploring relations between algorithm, proof, logic, and programming [44]. In this exploration, different conceptions of algorithm might emerge: an algorithm is implicitly included in the proof of a theorem (if the activity is from a problem to a theorem to a proof, or, in short, problem-theorem-proof); a proof of the correctness of an algorithm is given (problem-algorithm-proof); an algorithm is given as a computer program whose validity is established in some way (problem-program-validation) [16].

Although the exploration sketched in the previous paragraph may only be suitable for senior high school students, an algorithm should be considered not only as a useful tool that can solve certain problems, but also as a separate entity that can be investigated in itself. For example, apart from applying the algorithm for determining the greatest common divisors of two natural numbers when we use this algorithm as a tool, we may examine its applicability to whole or other numbers (or its complexity in terms of the number of operations needed to complete it) when we treat the algorithm as a separate entity (e.g, [16]). Such an approach calls for considering the so-called process-object nature of algorithm, whenever this approach is appropriate and accessible to students. This dual nature also characterizes other mathematical entities, such as relations and functions (e.g., [54]).

3. Cultivating CT through data practice

3.1. Preliminaries

As mentioned in Section 2.2, CT has mostly been cultivated through programming (e.g., [21]), which is hence often assumed as a dominant learning practice that would support CT development. However, to this end, other learning practices might be applied as well (e.g., [61]). Among these are data practices (e.g., data preparation and visualization) that may activate different CT components, such as abstraction, decomposition, and pattern recognition.

The relevance of data practices to developing CT is, for example, recognized by a CT definition that refers to core CT facets, assuming that these facets might be: abstraction (data collection and analysis, pattern recognition, modeling), decomposition, algorithms (algorithm design, parallelism, efficiency, automation), iteration,

debugging, and generalization [55]. Bearing in mind this relevance, CT assessment may also include some aspects of data practice. This was, for example, done in a large worldwide assessment named ICILS 2018 (International Computer and Information Literacy Study completed in 2018), which used tasks that called for programming as well as structuring and manipulating datasets [18].

As mentioned in Section 2.3.1, tens of millions of workers with specialized ICT knowledge are needed worldwide today, among whom data scientists are particularly important, applying various (often complex) techniques from mathematics, statistics, and computer science to obtain useful information from (big) datasets. It is reasonable to expect that in their future jobs most students would have to work with data as a foundation for their claims and actions regarding various professional issues, and, to this end, they may primarily use some simple data science techniques. Among these is exploratory data analysis that is applied to summarize the main characteristics of the dataset analyzed by using data visualization methods, primarily charts, aiming at discovering what the data can tell us not at formal data modeling or hypothesis testing [58]. This expectation regarding such use of exploratory data analysis is supported by the increasing application of dashboards (e.g., [62]), which are particularly suitable tools for this kind of analysis. In a specialized computer environment, building charts and dashboards (combining various types of charts and summary measures) can be (relatively effortlessly) done visually using the drag-and-drop approach.

Dashboards are interactive displays that are composed of two or more interactive reports, mostly charts, whose content updates automatically whenever there are changes in data or variables considered [33]. Dashboards are today used in various industries and areas (for a gallery of dashboards, visit https://www.yellowfinbi.com/analytics-best-practice/dashboard-gallery). Among them is learning analytics in education (e.g., [60]), where such interactive displays summarize the values of various learning indicators. Dashboards may also be used in education to support the work with data in various school subjects and university courses. If this work is practiced within a suitable learning cycle (e.g., a mathematical modeling cycle [29]), it would not only support the understanding of this cycle and the realization of its values in capturing the main features of disciplinary thinking (i.e., thinking applied in the particular discipline), but also support the development of important (disciplinary or general) notions, such as variable and functional dependence [33]. In other words, although interactive displays are primarily a means for visualizing data, they can also be a learning tool if used within an appropriate learning cycle [31]. Note that a growing demand for the inclusion of data science in secondary education (e.g., [23]) may, at introductory levels, profit from the work with interactive displays, whose visualizations (although mostly based on simple mathematical models such as frequencies, sums, and means) can support the discovery of useful (interesting) patterns, trends, effects, and interactions in the data examined. To find an interesting inter-

action in a tourism dataset, the reader may examine the visualizations available at https://www.mi.sanu.ac.rs/~djkadij/Dashboard.htm.

3.2. Data modeling using dashboards

The key activities in data modeling using dashboards (key stages) may be Asking questions, Preparing data, Visualizing data, Answering questions, Validating modeling, and Recommending changes. Apart from Visualizing data, the use of dashboards would support other data modeling steps, especially Answering questions and Validating modeling. In Answering questions, students have to match patterns, trends, effects, and interactions found with the questions posed, whereas in Validating modeling they might improve the modeling applied by using other variables or charts, or even other data or another dashboard. When datasets to model are not given to students, the use of dashboards may also support (though not primarily) the stage of Preparing data, because they may signal some oddities in data (e.g., outliers, missing or inappropriate data) that should be addressed before the stage of Visualizing data is applied. In most cases, datasets to model should be given to data modelers, especially novices, because removing these oddities is a very challenging task, even for data scientists who usually spend most of their time preparing data, i.e., collecting, cleaning, and organizing data [29].

Data modeling using dashboards clearly calls for abstraction (e.g., in using variables), decomposition (e.g., in deciding what charts to include in a dashboard, or what variables to use in a chart and in what role), and pattern recognition (e.g., in recognizing an effect or a trend in data). Apart from decomposition, this modeling would promote other computational strategies, such as top-down and bottom-up approaches [31], recalling that these approaches are relevant to mathematical problem solving proposed Pólya's [50]. A top-down approach is applied when the modeler goes from a dashboard as a whole to its individual reports as parts, whereas a bottom-up approach is used when he/she starts from some individual reports and combine them to create a dashboard; instead of a single approach, their combination is often applied. In addition, building a dashboard may make use of another computational strategy called rapid prototyping, which denotes an iterative process through which the modeler incrementally presents what the dashboard under development will look like in order to get feedback and validation from peers and future users [31]. This strategy is, in general, relevant to mathematical modeling whenever models of increased complexity are developed in an incremental fashion. To consider a way to promote these computational strategies, the reader may consider the development of a dashboard whose content is presented in Figure 2, but it should be kept in mind that only a basic understanding of these strategies may be promoted because the applied dashboard development (as is the case most often) calls for simple system engineering

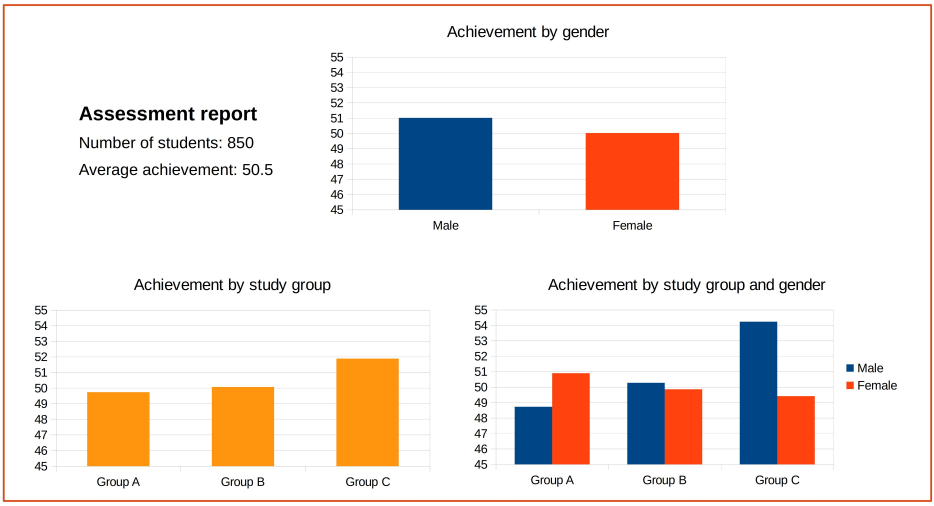

Figure 2. Assessment dashboard.

[31]. Note that although the three computational strategies, especially rapid prototyping, have been under-represented in CT-related research, there is a CT facet named iteration [55] under which these strategies might be discussed.

As the previous consideration shows, the presented work with data offers a number of learning opportunities: cultivating a modeling (or a data inquiry) cycle; supporting the development of important disciplinary notions (e.g., variable and functional dependence); promoting a basic understanding of CT strategies, such as decomposition as well as rapid prototyping, and top-down and bottom-up approaches. To be practiced skillfully, this work requires the modeler to demonstrate a range of skills, such as choosing relations to examine, identifying dependent and independent variables (Asking questions), selecting charts and measure to use (Visualizing data), recognizing regularities in charts produced, and connecting regularities to questions asked (Answering questions) [31]. A number of challenges would be faced in the development and use of these skills. Among them are the following: using appropriate sets of variables to answer questions; selecting appropriate charts and measures; considering context properly to interpret findings. There are several possible reasons for these challenges, such as complexity of this data practice when considered as a design task; limited experience in using various charts and measures; and complex interactions of knowledge from different domains [27, 29]. To alleviate these and other challenges, hints and supports (the so-called scaffolds) need to be provided to modelers, which would hopefully enable them to complete successfully, on their own, data modeling using dashboards. These scaffolds may connect key stages using their

underlying skills (e.g., variables selection with charts production; charts production with regularities recognition) and link contextual/conceptual and technology-related issues (e.g., between questions to ask and chart types to use, visualizations produced and questions to answer, or modeling features to validate and technology components used) [29, 31].

Data modeling using dashboards should be practiced within a rich computational environment that supports various CT assets, such as ZOHO Analytics (https://www.zoho.com/analytics/). Bearing in mind the learning paths examined in Section 2.2, to support (and empower) this practice, the following learning path may be applied: examine dashboards to understand or evaluate data modeling (DM) completed – modify dashboards to debug or extend DM done – create dashboards to perform DM by yourself. To assess the outcome of data modeling using dashboards, the instructor may examine students' portfolios about dashboards evaluated, improved, or fully developed (done individually or through a cooperative work), focusing on success of pursuing each key DM stage and connecting these stages in terms of major skills underlying them and their links [31].

Although the presentation of data modeling using dashboards is linked to fostering CT in a mathematical context, this modeling may also contribute to fostering CT in other school subjects if embedded in another disciplinary context using an appropriate learning cycle (e.g., in statistics using a data inquiry cycle). Such a data practice is also in accord with a CT pedagogy regarding a range of disciplines that calls for focusing on interactive visualizations or simulations, modeling and troubleshooting of datasets, and searching for patterns in large datasets [46]. Regarding the work with data in general, this focus aligns with an already underlined today's practice of scientific inquiry, whose three pillars are theory, experimentation, and a combination of modeling, simulation, optimization, and visualization [17].

4. Closing remarks

After briefly presenting an emerging educational context regarding the application of CT, this contribution first examined critical issues of CT/AT concerning the notion of CT/AT, the state of CT/AT-oriented educational research, and the integration of CT/AT in the school mathematics curriculum. Although a widely accepted definition of CT is lacking, it was argued that CT cornerstones might be decomposition, abstraction, algorithmization, and automation, where the first three might comprise AT. The examination of the state of CT/AT-oriented educational research showed that research on CT/AT in mathematics learning is limited but growing, being concerned with exploring various areas through different activities to foster this learning, especially developing and relating procedural and conceptual mathematical knowl-

edge. Regarding the integration of CT/AT in the school mathematics curriculum, the rationale for doing that is supported by both societal and professional needs. Various models have been used (within one or several existing school subjects or within new school subjects), but they remain unexamined and are by and large untested. To develop CT/AT, instruction should, whenever appropriate and accessible to students, be based on applying a range of activities designed in accord with disciplinary practice (increasingly empowered by computations), exploring interfaces between mathematics and computer science, and considering the dual nature of algorithm (a tool to apply as well a separate entity to investigate). In doing that, suitable learning paths may be followed, such as use problem solutions (to understand or evaluate them) – modify problem solutions (to debug or extend them) – create problem solutions, i.e., develop problem solutions from scratch.

After the examination of these critical CT/AT issues concerning their notion, examination in educational research, and integration in the school mathematics curriculum, this contribution presented a way to cultivate CT through data practice. This practice, which has been increasingly advocated in educational research, is based on using sets of interactive reports called dashboards. The rationale for using such interactive displays is supported by an expectation that in their future jobs, most students would have to work with data as a foundation for their claims and actions regarding various professional issues, and to this end, they may primarily apply exploratory data analysis with dashboards, because on one hand, this analysis, as an introductory data science technique, could be accessible to most students, and, on the other, dashboards, which have been increasingly applied in various industries and areas, are particularly suitable tools for this kind of analysis. After describing the key stages in data modeling with dashboards, CT components involved in this modeling are discussed (e.g., pattern recognition), especially computational strategies (e.g., top-down approach), which, despite their educational relevance, have been under-represented in CT-related research. Next, various learning issues concerning the proposed data modeling with dashboards were discussed, including learning opportunities, underlying skills required, expected challenges in practicing this modeling and possible reasons for these challenges, scaffolds that would alleviate these challenges, as well as a learning path that may be followed in practicing this modeling. Finally, it was considered whether the advocated data practice is aligned, in a pedagogical way, with today's practice of scientific inquiry. All in all, the presentation showed that data modeling using dashboards may be a promising way to cultivate CT, provided that the discussed learning issues are adequately treated.

The content of this contribution has evidenced that more research is needed on linking CT with mathematics learning. Although it showed how CT could be developed through exploring the area of statistics using exploratory data analysis with dashboards, this is just an initial research step in this research direction. Further

research may be (more) concerned with, for example, exploring various areas (e.g., probability) through different activities (e,g., simulation), designing these activities in accord with disciplinary practice (e.g., experimenting and approximation), exploring interfaces between mathematics and computer science (e.g., computational geometry), and considering algorithm in (more) explicit and detailed way (e.g., its dual nature). The outcomes of such a directed research would considerably inform instruction and help designing it.

Bearing in mind that the models of curricular CT integration remain largely unexamined, research is also needed on this integration, and, to this end, research could apply a detailed evaluation, which may, as already suggested, examine critical curricular components, such as goals, content, materials, forms of teaching, student activities, and assessment. As materials considerably influence teaching, learning, and assessment, applying appropriate materials, developed in lines that align with the proposed goals and content, seems to be the most critical component not only for this integration, but also for teacher education and further professional development.

To summarize, as there is a long-standing reliance on algorithms in mathematics, CT/AT should be cultivated in mathematics education, especially today with a growing application of computer tools in almost every areas of our work and life. Although thinking supported by technology has been named differently in the literature – computational, algorithmic, or even programming thinking – and defined in a number of ways, the focus in mathematics education should be on cultivating the aspects of mathematical thinking using tools and techniques from computer science. If this cultivation, supported by various suitable materials describing CT/AT based activities, is realized in appropriate ways in mathematical classes, the integration of CT/AT in the school curriculum would be a success. Undoubtedly, such an integration calls for international cooperation and sharing among educators and researchers at all educational levels. In doing that, special care may be taken about the following issues: how to define thinking with technology in a precise way; how to cultivate this thinking accordingly, focusing on the development of mathematical reasoning; and how to assess its contribution to this development in an adequate way [35].

Acknowledgments. The author dedicates the contribution to his son Aleksandar.

Funding. This contribution resulted from the author's research funded by the Ministry of Education, Science and Technological Development of the Republic of Serbia (Contract No. 451-03-68/2022-14/200018).

References

[1] S. Abramovich, Mathematical problem posing as a link between algorithmic thinking and conceptual knowledge. *Teach. Math.* **18** (2015), no. 2, 45–60

[2] M. Artigue, The future of teaching and learning mathematics with digital technologies. In *Mathematics Education and Technology – Rethinking the Terrain. The 17th ICMI Study*, edited by C. Hoyles and J. B. Lagrange, pp. 463–476, Springer, New York, 2010

[3] Asia Pacific Economic Cooperation (APEC), *Data science and analytics skills shortage: Equipping the APEC workforce with the competencies demanded by employers*. APEC, Singapore, 2017, https://www.apec.org/Publications/2017/11/Data-Science-and-Analytics-Skills-Shortage

[4] Asia Pacific Economic Cooperation (APEC), *Project DARE (Data Analytics Raising Employment)*. APEC, Singapore, 2018, https://www.apec.org/Press/News-Releases/2018/1109_dare

[5] Australian Curriculum, Assessment and Reporting Authority (ACARA), *Australian curriculum: Mathematics – All elements F–10 consultation curriculum*. NSW, Sydney, 2021, https://www.australiancurriculum.edu.au/media/7044/mathematics_all_elements_f-10.pdf

[6] T. Bell and J. Vahrenhold, CS unplugged—How is it used, and does it work? In *Adventures Between Lower Bounds and Higher Altitudes*, edited by H. J. Böckenhauer, D. Komm, and W. Unger, pp. 497–521, Lect. Notes Comput. Sci. 11011, Springer, Cham, 2018

[7] S. Bocconi, A. Chioccariello, G. Dettori, A. Ferrari, and K. Engelhardt, *Developing computational thinking in compulsory education – Implications for policy and practice*. Joint Research Centre, European Commission, European Union, Luxemburg, 2016, https://publications.jrc.ec.europa.eu/repository/bitstream/JRC104188/jrc104188_computhinkreport.pdf

[8] K. Brennan and M. Resnick, New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association (Vancouver, Canada)*, 2012

[9] C. Clapham and J. Nicholson, *The Concise Oxford Dictionary of Mathematics*. 5th edn., Oxford University Press, Oxford, 2014 Zbl 1290.00009 MR 3186178

[10] L. de Raedt and M. Bruynooghe, Interactive concept-learning and constructive induction by analogy. *Mach. Learn.* **8** (1992), no. 2, 107–150 Zbl 0751.68051

[11] P. J. Denning, Remaining trouble spots with computational thinking. *Commun. ACM* **60** (2017), no. 6, 33–39

[12] P. J. Denning and M. Tedre, *Computational Thinking*. MIT Press, Cambridge, MA, 2019

[13] A. A. DiSessa, Computational literacy and "the big picture" concerning computers in mathematics education. *Math. Think. Learn.* **20** (2018), no. 1, 3–31

[14] P. Drijvers, Tools and taxonomies: a response to Hoyles. *Res. Math. Edu.* **20** (2018), no. 3, 229–235

[15] P. Drijvers, H. Kodde-Buitenhuis, and M. Doorman, Assessing mathematical thinking as part of curriculum reform in the Netherlands. *Educ. Stud. Math.* **102** (2019), 435–456

[16] V. Durand-Guerrier, A. Meyer, and S. Modeste, Didactical issues at the interface of mathematics and computer science. In *Proof Technology in Mathematics Research and Teaching*, edited by G. Hanna, D. Reid, and M. de Villiers, pp. 115–138, Math. Educ. Digit. Era 14, Springer, Cham, 2019

[17] European Mathematical Society (EMS), Position paper on the European Commission's contributions to European research. 2011

[18] J. Fraillon, J. Ainley, W. Schulz, D. Duckworth, and T. Friedman, *IEA International Computer and Information Literacy Study 2018 Assessment Framework*. Springer, Cham, 2019

[19] Groupe de travail des sociétés savantes de mathématiques et d'informatique, *Propositions pour le futur programme de mathématiques du lycée*. Sociétéinformatique de France, Grenoble, France, 2016

[20] D. Guin, K. Ruthven, and L. Trouche (eds.), *The Didactical Challenge of Symbolic Calculators: Turning a Computational Device into a Mathematical Instrument*. Springer, New York, 2005

[21] D. Hickmott, E. Prieto-Rodriguez, and K. Holmes, A scoping review of studies on computational thinking in K-12 mathematics classrooms. *Digit. Exp. Math. Educ.* **4** (2018), no. 1, 48–69

[22] C. Hoyles and R. Noss, Revisiting programming to enhance mathematics learning. 2015, paper presented at Math + Coding Symposium, Western University, London, Canada

[23] International Data Science in Schools Project (IDSSP) Curriculum Team, Curriculum frameworks for introductory data science. 2019, http://www.idssp.org/files/IDSSP_Frameworks_1.0.pdf

[24] D. M. Kadijevich, Can mathematics students be successful knowledge engineers? *J. Interact. Learn. Res.* **9** (1998), no. 3–4, 235–248

[25] D. M. Kadijevich, An approach to learning mathematics through knowledge engineering. *J. Comput. Assist. Learn.* **15** (1999), no. 4, 291–301

[26] D. M. Kadijevich, Neglected critical issues of effective CAS utilization. *J. Symb. Comput.* **61–62** (2014), 85–99 Zbl 1284.97033

[27] D. M. Kadijevich, Data modelling with dashbards: Opportunities and challenges. In *Promoting Understanding of Statistics About Society. Proceedings of the Roundtable Conference of the International Association of Statistics Education (IASE), Berlin, Germany, July 2016*, edited by J. Engel, ISI/IASE, The Haag, the Netherlands, 2016

[28] D. M. Kadijevich, A cycle of computational thinking. In *Proceedings of the 9th International Conference on e-Learning*, edited by B. Trebinjac and S. Jovanović, pp. 75–77, Metropolitan University, Belgrade, 2018

[29] D. M. Kadijevich, Data modelling using interactive charts. *Teach. Math.* **21** (2018), no. 2, 55–72

[30] D. M. Kadijevich, Relating procedural and conceptual knowledge. *Teach. Math.* **21** (2018), no. 1, 15–28

[31] D. M. Kadijevich, Cultivating computational thinking through data practice. In *Empowering Learners for Life in the Digital Age*, edited by D. Passey, R. Bottino, C. Lewin, and E. Sanchez, pp. 24–33, Springer, Cham, 2019

[32] D. M. Kadijevich, A cycle of computational thinking and its relevance: An empirical study. In *Proceedings of the 10th Conference on e-Learning*, edited by S. Jovanović and B. Trebinjac, pp. 136–138, Metropolitan University, Belgrade, 2019

[33] D. M. Kadijevich, Interactive displays: Use of interactive charts and dashboards in education. In *Encyclopedia of Education and Information Technologies*, edited by A. Tatnall, pp. 968–973, Springer, Cham, 2020

[34] D. M. Kadijevich and M. Stephens, Modern statistical literacy, data science, dashboards, and automated analytics and its applications. *Teach. Math.* **23** (2020), no. 1, 71–80

[35] D. M. Kadijevich, M. Stephens, and A. Rafiepour, Emergence of computational/algorithmic thinking and its impact on the mathematics curriculum. In *Mathematics curriculum reforms around the world*, edited by Y. Shimizu and R. Vithal, Springer, Cham, 2023

[36] M. Kallia, S. P. van Borkulo, P. Drijvers, E. Barendsen, and J. Tolboom, Characterising computational thinking in mathematics education: a literature-informed Delphi study. *Res. Math. Educ.* **23** (2021), no. 2, 159–187

[37] P. S. Kenderov, Powering knowledge versus pouring facts. In *Invited Lectures from the 13th International Congress on Mathematical Education*, edited by G. Kaiser, H. Forgasz, M. Graven, A. Kuzniak, E. Simmt, and B. Xu, pp. 289–306, CME-13 Monographs, Springer, Cham, 2018

[38] D. Kotsopoulos et al., A pedagogical framework for computational thinking. *Digit. Exp. Math. Educ.* **3** (2017), no. 2, 154–171

[39] I. Lee, Reclaiming the roots of CT. *CSTA Voice* **12** (2016), no. 1, 3–5

[40] I. Lee et al., Computational thinking for youth in practice. *ACM Inroads* **2** (1), no. 2011, 33–37

[41] E. Lockwood, A. F. DeJarnette, and M. Thomas, Computing as a mathematical disciplinary practice. *J. Math. Behav.* **54** (2019), Paper No. 100688

[42] E. E. Lockwood, A. DeJarnette, A. Asay, and M. Thomas, Algorithmic thinking: An initial characterization of computational thinking in mathematics. In *Proceedings of the 38th Annual Meeting of the North American Chapter of the International Group for the Psychology of Mathematics Education*, edited by M. B. Wood, E. E. Turner, M. Civil, and J. A. Eli, pp. 1588–1595, The University of Arizona, Tucson, AZ, 2016

[43] M. Lodi, Informatical thinking. *Olymp. Inform.* **14** (2020), 113–132

[44] S. Modeste, Impact of informatics on mathematics and its teaching. In *History and philosophy of computing (HaPoC 2015)*, edited by F. Gadducci and M. Tavosanis, pp. 243–255, IFIP Adv. Inf. Commun. Technol. 487, Springer, Cham, 2016

[45] C. Mouza, H. Yang, Y.-C. Pan, S. Y. Ozden, and L. Pollock, Resetting educational technology coursework for pre-service teachers: A computational thinking approach to the development of technological pedagogical content knowledge (TPACK). *Australas. J. Educ. Technol.* **33** (2017), no. 3, 61–76

[46] National Research Council, *Report of a workshop of pedagogical aspects of computational thinking*. The National Academies Press, Washington, DC, 2011

[47] M. Niss, Mathematical standards and curricula under the influence of digital affordances-different notions, meanings and roles in different parts of the world. In *Digital Curricula in School Mathematics*, edited by M. Bates and Z. Usiskin, pp. 239–250, Information Age Publishing, Charlotte, NC, 2016

[48] Organization for Economic Co-operation and Development (OECD), *PISA 2021 mathematics framework (draft)*. OECD, Paris, 2018, https://www.oecd.org/pisa/pisaproducts/pisa-2021-mathematics-framework-draft.pdf

[49] S. Papert, *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books, New York, 1980

[50] G. Pólya, *How to Solve It*. Princeton University Press, Princeton, NJ, 1945

[51] E. Prieto and K. Holmes, Working mathematically and thinking computationally: Capitalizing on commonalities for integrated teaching. 2021, paper presented at Topic Study Group 14 "Teaching and learning of programming and algorithms" at the 14th International Congress on Mathematical Education, https://www.icme14.org

[52] P. J. Rich, G. Egan, and J. Ellsworth, A framework for decomposition in computational thinking. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '19)*, pp. 416–421, ACM, New York, 2019

[53] T. Scantamburlo, *Philosophical aspects in pattern recognition research*. Ph.D. thesis, Department of Informatics, Ca' Foscari University of Venice, Venice, Italy, 2013

[54] A. Sfard, On the dual nature of mathematical conceptions: Reflections on processes and objects as different sides of the same coin. *Educ. Stud. Math.* **22** (1991), no. 1, 1–36

[55] V. J. Shute, C. Sun, and J. Asbell-Clarke, Demystifying computational thinking. *Educ. Res. Rev.* **22** (2017), 142–158

[56] M. Stephens and D. M. Kadijevich, Computational/algorithmic thinking. In *Encyclopedia of Mathematics Education*, edited by S. Lerman, pp. 117–123, Springer, Cham, 2020

[57] The Royal Society, *Shut down or restart? The way forward for computing in UK schools*. The Royal Society, London, 2012, https://royalsociety.org/~/media/education/computing-in-schools/2012-01-12-computing-in-schools.pdf

[58] J. M. Tukey, *Exploratory Data Analysis*. Addison-Wesley, Reading, PA, 1977
Zbl 0409.62003

[59] M. Webb et al., Computer science in K-12 school curricula of the 2lst century: Why, what and when? *Educ. Inf. Technol. (Dordr.)* **22** (2017), no. 2, 445–468

[60] M. E. Webb et al., Challenges for IT-enabled formative assessment of complex 21st century skills. *Technol. Knowl. Learn.* **23** (2018), no. 3, 441–456

[61] D. Weintrop et al., Defining computational thinking for mathematics and science classroom. *J. Sci. Educ. Technol.* **25** (2016), no. 1, 127–141

[62] S. Wexler, J. Shaffer, and A. Cotgreave, *The Big Book of Dashboards: Visualizing Your Data Using Real-World Business Scenarios*. 1st edn., Wiley, Hoboken, NJ, 2017

[63] J. M. Wing, Computational thinking. *Commun. ACM* **49** (2006), no. 3, 33–35

[64] J. M. Wing, Research notebook: Computational thinking—What and why? *Link Newsl.* **6** (2011), 1–32

Djordje M. Kadijevich
Institute for Educational Research, Dobrinjska 11/III, Belgrade, Serbia; djkadijevic@ipi.ac.rs