What global socio-ecological precarity might mean for the research and teaching of algorithmic thinking: A role for reflective knowing

Alf Coles[1] and Christof Weber[2]

[1]School of Education, University of Bristol, United Kingdom, [2]University of Teacher Education Lucerne, Switzerland

In this article we share some ideas about how socio-ecological thinking intersects with algorithmic thinking, in mathematics education. Socio-ecological approaches to mathematics education have, at their centre, a radical idea that there is no separation between nature and culture, or body and mind, or information and matter. Such approaches provoke questions about how our research concerns might alter if we take seriously the precarious nature of the sustainability of life on the planet. We illustrate such a re-working of research concerns by considering algorithmic thinking. Algorithmic thinking, as we understand it, involves not only performing and designing, but also analysing and comparing algorithms. One way its research and teaching might alter, in light of questions of sustainability, is to include reflective knowing of algorithms.

Keywords: socio-ecology; sustainability; critical mathematics education; algorithms; algorithmic thinking

Introduction

This article shares some emergent thinking as we grapple with what climate chaos, political instability and war might mean for the future of research, teaching and learning of mathematics in schools and universities. In what follows, we first set out what we mean by socio-ecological thinking, and then do the same for algorithmic thinking. Finally, we consider how the teaching of and research into algorithmic thinking might alter in light of socio-ecological precarity. We conclude by inviting other scholars to join the conversation in re-evaluating teaching and research concerns in different areas of mathematics education.

A socio-ecological approach to mathematics education

The term “socio-ecology” is an old one, being used by Bronfenbrenner (1977) and others. It has been re-kindled recently (e.g., see Coles, 2022) and been used to point to: (a) how mathematics education research has tended to take the living world as a fixed background for concerns; (b) how social and ecological concerns cannot be separated. Thinking with the socio-ecological, invites attention to research areas that are perhaps rarely considered, such as, how might mathematics teaching change if the air in a classroom is highly polluted? Or, what priorities for mathematics education exist when teaching in a refugee camp?

More philosophically, a socio-ecological perspective might alert us to the absence of considerations about the ecology of the planet, in many constructivist and socio-cultural studies in mathematics education. Indeed, it is only relatively recently that work within critical mathematics education has embraced, e.g., environmental justice (Skovsmose, 2023) as a core aim. One aim of eliding the social and ecological in the phrase "socio-ecological” is to suggest that nature and culture are not separate

realms (Kirby, 2011), or that mind and body, or information and matter, cannot be separated. We are interested in what widening our scope, to consider broader socio-ecological issues, does to considerations of the future of mathematics education. We illustrate the potential for such thinking with the field of algorithmic thinking.

Algorithms and algorithmic thinking

This section aims to clarify the terms "algorithm" and "algorithmic thinking"—a relatively new area of research, relating to (but more general than) combinatorial thinking. Algorithms are a special way of solving problems (e.g., see Maurer 1998). Consider, for example, the standard algorithm for calculating the multiplication of two natural or decimal numbers ("long multiplication"), of the algorithm for checking the primality of a natural number ("trial division") or for generating all possible combinations of different properties. Looking across these cases, we propose an algorithm:

(1) consists of a predefined and fixed sequence of finitely many instructions;
(2) solves a class of (in)finitely many individual problems ("instances");
(3) takes a finite number of steps to solve the problem; and does so systematically.

The first property (1) ensures that the sequence of instructions is already determined before they are executed (and it is not, for example, changed depending on the instance), the second property (2) ensures that the algorithm does not solve a single instance but the generalised problem (not "12 ÷ 3= ?", but "a ÷ b = ?"), and the third property (3) states that the algorithm returns the correct solution after a finite number of steps (and thus in finite time). The fact that it must proceed systematically to do so is rarely explicitly stated (e.g., see Maurer 1998). This could be due to the fact that, in order to find the correct solution in a finite number of steps, an algorithm necessarily has to proceed in a systematic way. For example, if we were to generate all possible combinations and proceed unsystematically, we would not know at any time whether we have already found all combinations. As a result, the search would require an unlimited number of steps and would therefore take an unlimited amount of time.

Moreover, some authors emphasise the relationship between the number of instructions (of the algorithm) and the number of steps (in its execution) (Modeste, 2012). Thus, in addition to properties (1) to (3), every algorithm:

(4) works out the solution in a number of steps that depends on the "size" of the elements of the actual instance.

This fourth property (4) states that an algorithm for solving an instance usually executes more or fewer steps than it contains instructions, achieved by structures such as iterations (loops or repeating steps) and branchings (if / then / else conditions). This means that formulas (such as the quadratic formula) would *not* yet be considered algorithms (Modeste, 2012): they do not satisfy property (4), since the number of steps in computing the solution calculation does not depend on whether the coefficients of the quadratic equation to be solved are "large" or "small". Furthermore, property (4) makes it clear that comparisons such as "algorithms are like recipes" are not appropriate. Recipes do indeed consist of a fixed sequence of a finite number of instructions (which, if executed correctly, lead to a "solution", the menu). However, since they contain no iterations or branchings, they should be considered as procedures or strategies and not as algorithms in the strict sense.

Another debate is about equating algorithms with computer programmes (Modeste speaks of the "amalgamation of algorithm with program" (2012, pp. 127–129)). Algorithms can be implemented in a programming language in order to be run as a programme on a computer. However, mathematics was concerned with algo-

rithms long before there were programming languages and computers. Furthermore, not every programme is based on an algorithm in the strict sense, for some applications it is virtually essential that a programme does not terminate (e.g., controlling traffic lights). The fact that, apart from everyday language, we do not have a standardized language for representing algorithms could be (partly) responsible for equating algorithms with programmes. It is therefore quite natural to use a notation from the computer environment (programming language, flowchart, etc.).

Algorithms in schools and curricula

Algorithms are a great thing from a mathematical point of view: with their help, tasks that initially appear as challenging problems become mere routine exercises. Instead of concentrating on solving a problem and struggling with the process, we can focus on other points, such as the follow-up question, etc.

In the school context, however, this advantage can be a double-edged sword: on the one hand, algorithms make the solution process easier; on the other hand, students do not need to understand why an algorithm solves the problem, how it works, if they simply follow its rules in order to arrive at the solution. In the 1980s, empirical studies began to discredit the learning (and teaching) of algorithms in primary schools. The term "algorithmic" was increasingly equated with "rote learning" and used in a pejorative way (e.g., algorithmic as the opposite of conceptual). This was probably also the reason why some countries removed long division and other standard algorithms from the curriculum (Fan & Bokhove, 2014). But is this not throwing the baby out with the bathwater? Is the problem not less with the algorithms themselves than with the way we teach them? What might it mean to have "deep knowledge of algorithms" (ibid., p. 484) rather than just rote knowledge?

Algorithmic thinking: a revaluation

In the context of teaching and learning mathematics, we often speak of "mathematical thinking", more specifically of "algebraic thinking", "functional thinking" or "combinatorial thinking". The addition of "thinking" also gives the adjective "algorithmic" a new lustre: "algorithmic thinking". Last but not least, the educational policy demand for the development of "computational thinking" in (computing) education is probably also responsible for the fact that "algorithmic thinking" now appears as an objective in education plans (e.g., OECD, 2023).

Despite this, there seems to be little consensus on what algorithmic thinking can and should mean in the classroom. Like linguistically related constructs such as algebraic or functional thinking, algorithmic thinking could be understood as a human activity that is "typical" for working with algorithms, for solving problems in an algorithmic way. This circumscription shifts the question about constitutive elements of algorithmic thinking to the question of typical practices with algorithms. We will now briefly present four relevant *algorithmic practices*:

(1) Probably the most common classroom practice with algorithms is their *performing*: algorithms are presented step by step by the teacher and imitated by the students until they can perform them at a good speed and error-free.

(2) A second form of practice is *designing* and developing algorithms. In computing education, design practices would be decomposing, abstracting, debugging etc. In the context of primary school arithmetic lessons, some educators may instead suggest that pupils develop informal or their own idiosyncratic

strategies, or that they reinvent an existing algorithm (for examples, see Freudenthal 2002, pp. 57–61).

Since "performing" refers to given algorithms and "developing" to algorithms that are not given—and therefore do not (yet) exist for students— the two practices can be understood as poles of a continuum. We can at least think of two other practices:

(3) A third form of algorithmic practice can be summarized under the notion of *analysing*, such as analysing the effectiveness, efficiency or limitations of a present algorithm: *for what reasons does this algorithm actually solve the problem it promises to solve? Does the algorithm solve the problem with the minimum number of computational steps, in the minimum amount of space? Which instances do not belong to the class that the algorithm solves?* While mathematics is concerned with proving the effectiveness of algorithms, computer science is interested in the question of absolute efficiency (complexity theory). At least relative efficiency can be analysed in mathematics classrooms: *How can you add a series of consecutive numbers more efficiently than by summing the numbers by magnitude? How can we check the primality of a natural number n more efficiently than by trial division up to n–1?*

(4) Another, fourth form of algorithmic practice, is *comparing* (Weber, 2019). Different algorithms that solve the same problem can be compared with each other (e.g., the traditional right-to-left multiplication with left-to-right multiplication, synthetic division with Horner's method, an idiosyncratic approach with the standard algorithm): *Which algorithm is shorter? Which one is faster? Which algorithm is clearer (for us)?* Or we compare different representations of the same algorithm (e.g., symbolic programming language versus diagrammatic flowchart): *Which representation makes which aspects of the algorithm explicit, which ones remain implicit?* In special cases where two problems are mathematically related (e.g., division of numbers and division of polynomials, long division and logarithms (Weber, 2019)), students can compare the similarities and differences of the related algorithms.

These last two algorithmic practices shift attention from "thinking *like* an algorithm" to "thinking *about* algorithms" (Maurer, 1998, p. 24): an algorithm is no longer a tool for solving problems but becomes an *object* in its own right. This corresponding attitude towards algorithms is characteristic of algorithmics, a discipline at the interface of mathematics and computer science (e.g., Maurer, 1998; Modeste, 2012). In the context of learning, the corresponding shift in perspective marks understanding (Sfard, 1991): Just as students learn to talk and think about mathematical concepts such as numbers or functions, a further *educational goal is to be able to talk about and reflect on algorithms*, not just to be able to perform algorithms without errors.

Bringing algorithmic thinking and socio-ecological concerns together

In thinking about how concerns in research and teaching of algorithmic thinking might alter, in light of questions of the socio-ecological, we turned to critical mathematics education (CME) as one possible source of insight, since CME explicitly deals with environmental justice and also deals with algorithms (Skovsmose, 1994).

Related to the use of algorithms, mathematics can be used to model our living environment ("reality"), for example to determine the average income or to analyse and predict climate change. Accordingly, mathematical modelling has a long tradition at all school levels (cf. solving word problems) and in CME. CME emphasises that in addition to this descriptive function, modelling also has a *formatting* (Skovsmose,

1994, 2023) and as such a *normative* (Pohlkamp & Heitzer, 2021) function: mathematics also shapes our living environment and our coexistence, e.g. how we use the results of mathematical modelling to make decisions and act and how we perceive a real situation (setting the tax rate, determining how we deal with the climate, etc.).

Accordingly, CME requires not only to teach (a) basic knowledge and skills and (b) operating within this knowledge and skills (e.g. modelling). Teachers should also stimulate the development of (c) *reflective knowing* as a third component (Skovsmose, 1994; Fischer, 2001, as cited in Vohns, 2017). Future citizens, in order to assume social responsibility as become well-informed laypersons, must not only be able to set up and calculate models themselves, but also have an "understanding *about* mathematics" (Skovsmose 1994, p. 47) and, in particular, be aware about the normative function of modelling and models. This means, for example, that they question modelling (and its results) in terms of its subjectivity, clarify its premises and quality criteria, etc. (Pohlkamp & Heitzer, 2021, Skovsmose, 2023). Several attempts show how this demand could be implemented in the classroom (child benefit (Skovsmose, 1994), measures of poverty (Vohns, 2017), good-monkey-bad-monkey game (Lengnink & Pohlkamp, 2022) etc.).

Reflective knowing of algorithms

Just as models and their results format our (social and other) environments, so do algorithms, e.g., when so-called "automated decision-making systems" based on algorithms calculate decisions on the basis of which real actions are then taken (Lengnink & Pohlkamp, 2022). Critical mathematics education therefore needs to reflect not only on models and their implications, but also on algorithms, whether they are modelling a real-world problem or solving a mathematical one. Instead of just asking questions like "*Did we follow the algorithm's instructions correctly? Did we use the right algorithm?*" (which would reflect practice 1, see above), teaching must also encourage critical questions such as "*Do we have a choice between different algorithms? Would a different algorithm produce the same, a different solution? Is there another algorithm that solves the problem in a shorter, faster, more understandable way? What are the limits of the algorithm, what conditions must the instances satisfy?*" (for similar questions see Skovsmose, 2023, p. 52).

We recognise that the above questions may seem somewhat removed from socio-ecological concerns. However, we assume that classroom discussions on such questions might begin to allow students to develop the knowledge and competencies they will need to better meet the challenges of our future: *the reflective knowing of algorithms*. Socio-ecological precarity must surely prompt a radical re-look at our entire curriculum and methods of schooling. At the same time, we need proposals for what we can do now, given all the current constraints of schooling and university life. Pursuing a reflective knowing of algorithms offers one possible route, which has the potential to both meet current curriculum demands and, at the same time, allow teachers and researchers to pursue agenda relating to the socio-ecological.

The prompt to consider reflective knowing of algorithms has pushed us into new research territory. As far as we are aware, this is not a question which has been considered before, nor tried out in practice. We hope our discussion has illustrated how issues and concerns in a research field might change in light of socio-ecological questions. Our aim is to open a conversation and we invite scholars in other fields to consider how their research concerns might intersect with socio-ecological issues.

References

Bronfenbrenner, U. (1977). Toward an experimental ecology of human development. *American Psychologist*, *32*(7), 513–531. https://doi.org/10.1037/0003-066X.32.7.513

Coles, A. (2022). A socio-ecological turn in mathematics education: Reflecting on curriculum innovation. *Paradigma*, 207–228. https://doi.org/10.37618/PARADIGMA.1011-2251.2022.p207-228.id1168

Fan, L., & Bokhove, C. (2014). Rethinking the role of algorithms in school mathematics: A conceptual model with focus on cognitive development. *ZDM–The International Journal on Mathematics Education*, *46*, 481–492. https://doi.org/10.1007/s11858-014-0590-2

Freudenthal, H. (2002). *Revisiting mathematics education: China lectures*. Kluwer.

Kirby, V. (2011). *Quantum anthropologies: Life at large*. Duke University Press.

Lengnink, K., & Pohlkamp, S. (2022), Mathematical algorithms in civic contexts: mathematics education and algorithmic literacy. In J. Hodgen, E. Geraniou, G. Bolondi, & F. Ferretti (Eds.), *Proceedings of the Twelfth Congress of the European Society for Research in Mathematics Education* (pp. 1951–1959). University of Bozen-Bolzano.

Maurer, S. B. (1998). What is an algorithm? What is an answer? In L. J. Morrow & M. J. Kenney (Eds.), *The teaching and learning of algorithms in school mathematics* (pp. 21–31). National Council of Teachers of Mathematics.

Modeste, S. (2012). *Enseigner l'algorithme pour quoi? Quelles nouvelles questions pour les mathématiques? Quels apports pour l'apprentissage de la preuve?* [Doctoral dissertation, Université de Grenoble]. https://theses.hal.science/tel-00783294.

OECD (2023). *The future of education and skills: OECD learning compass for mathematics*. OECD. https://www.oecd.org/education/2030/OECD-Learning-Compass-for-Mathematics-2023-13-Oct.pdf

Pohlkamp, S., & Heitzer, J. (2021). Normative modelling as a paradigm of the formatting power of mathematics: Educational value and learning environments. In D. Kollosche (Ed.), *Proceedings of the Eleventh International Mathematics Education and Society Conference* (vol. 3, pp. 799–808). https://doi.org/10.5281/zenodo.5416165

Sfard, A. (1991). On the dual nature of mathematical conceptions: Reflections on processes and objects as different sides of the same coin. *Educational Studies in Mathematics*, *22*(1), 1–36. https://doi.org/10.1007/BF00302715

Skovsmose, O. (1994). Towards a critical mathematics education. *Educational Studies in Mathematics*, *27*, 35–57. https://doi.org/10.1007/BF01284527

Skovsmose, O. (2023). *Critical mathematics education*. Springer International. https://doi.org/10.1007/978-3-031-26242-5

Vohns, A. (2017). Bildung, mathematical literacy and civic education: The (strange?) case of contemporary Austria and Germany. In A. Chronaki (Ed.), *Proceedings of the Ninth International Mathematics Education and Society Conference* (vol. 2, pp. 968–978), University of Thessaly.

Weber, C. (2019). Comparing the structure of algorithms: the case of long division and log division. In U. T. Jankvist, M. van den Heuvel-Panhuizen, & M. Veldhuis (Eds.), *Proceedings of the Eleventh Congress of the European Society for Research in Mathematics Education* (pp. 698–705). Utrecht University.