Computational Thinking as General Education: Evolution and Transformation

Kwong-Cheong Wong
Hong Kong Community College
The Hong Kong Polytechnic University
cckcwong@hkcc-polyu.edu.hk

Abstract

This paper consists of four parts.

The first part traces the *evolution* of the idea of computational thinking. It starts from the computer revolution, reviewing how computer science emerges and becomes the fourth great scientific domain, and how computation is being increasingly applied to studying virtually all areas of human endeavor and becomes a scientific Weltanschauung. It is then followed by reviewing how Jeannette Wing revives the phrase "computational thinking" and re-interprets it as thinking like a computer scientist that is useful in other contexts and as the skill set of the 21st Century for everyone, like reading, writing, and arithmetic.

The second part discusses the *transformation* that teaching computational thinking would have on the traditional approach to teaching information and communication technology (ICT), the latter being aimed primarily at computer fluency. In a nutshell, to teach computational thinking we need to shift to solving problems beyond ICT and to promoting various types of thinking (e.g., algorithmic, logical, abstractive, creative, etc.) based on the concepts and principles fundamental to computer science.

The third part is a critical discussion of computational thinking and its place in our general education curriculum. It first acknowledges that computational thinking is a new and fundamental way of thinking and problem solving in our technological society with ubiquitous computing. It then argues that developing computational thinking has at least two inter-related advantages: it educates creative problem solvers and it enables the *transformation* of a society of mere consumers of technology into one of potential developers of this technology, and so computational thinking deserves a prominent place in our general education curriculum.

The fourth part concludes that we have reason to be optimistic about the development of computational thinking education in Hong Kong.

1. Introduction

Computational thinking has become the subject of worldwide attention in recent years, since Jeannette Wing (2006) revives this phrase and re-interprets it as thinking like a computer scientist which is useful in other contexts. As such, computational thinking is a new and fundamental way of thinking and problem solving in our technological society with ubiquitous computing. It has been argued that computational thinking is the skill set of the 21st century for everyone, like reading, writing, and arithmetic (e.g., Wing 2006, Mohaghegh et al. 2016). Consequently, in many countries, efforts have been spent on incorporating computational thinking into the curriculum of all levels. The goal of computational thinking education is to educate creative problem solvers in this digital world and to enable the transformation of a society of mere consumers of technology into one of potential developers of this technology. The aim of this paper is to argue that computational thinking deserves a prominent place in our general education curriculum.

This paper is structured as follows: Section 2 traces the evolution of computational thinking. Section 3 describes the transformation that computational thinking would bring to the teaching of information and communication technology (ICT). Section 4 critically discusses computational thinking and its place in our general education curriculum. Section 5 concludes.

2. Evolution of Computational Thinking

> "The computer revolution is a revolution in the way we think and in the way we express what we think."
>
> --- Hal Abelson (1996) *Structure and Interpretation of Computer Programs*, Preface to the First Edition

Although the term "computational thinking" has become popular only in recent years, its evolution can be traced back, if not earlier, to the computer revolution which started in the 1940s. The two main forces behind this computer revolution that have been driving the evolution of computational thinking are the advancement of digital technologies and the rise of computer science, both of which center around the great invention of mankind: the computer. Since 1945, in which year the first modern digital computer, ENIAC, appeared, digital technologies have been advancing rapidly: mainframe computers, mini computers, personal computers, computer networks, the Internet, mobile devices, cloud computing. Computing is now ubiquitous.[1] Alongside this advancement of digital technologies, there was the rise of computer science as a recognized scientific discipline. Founded in the early 1960s, computer science is a relatively young field – the first university department of computer science was established in 1962 (Fluck et al. 2016, p. 38).[2] But over these 60 or so years, computer science has developed very rapidly. Now computing has duly been recognized as the 4th great scientific domain (Rosenbloom 2012) and computation has become a scientific Weltanschauung (Papadimitriou 2016), changing as it does enormously the landscape of scientific research (Johnson 2001, Denning 2017). What is more, computational methods are now being increasingly applied to studying virtually *all* areas of human endeavor. Consequently, with this lens of computation (Karp 2011), many fields have seen the growth of a computational counterpart, e.g., computational statistics (Gentle et al. 2012), computational biology (Navlakha & Bar-Joseph 2011, Markowetz 2017), computational linguistics (Mitkov 2005), computational psychology (Sun 2008), and computational metaphysics (Fitelson & Zalta 2007), just to name a few. Given that computer science is being increasingly recognized as important as natural and mathematical sciences, there has been a recent trend to incorporate computer science as a subject to the school curriculum in many countries, including the UK, Australia, USA, Czech Republic, Denmark, Lithuania, Poland and the Netherlands (Fluck et al. 2016).[3]

The term "computational thinking" was first coined by S. Papert (1980). But it was Jeannette Wing who revived the phrase and re-interpreted it as thinking like a computer scientist which is useful in *other* contexts (Wing 2006, 2008) that eventually made this phrase extremely popular nowadays. Wing (2006) argued further that computational thinking is the skill set of the 21st Century for everyone, like reading,

[1] For a terrific history of this digital revolution, see Isaacson (2014).
[2] Some claimed that computer science was born in 1936 when Alan Turing published a remarkable paper in which he outlined the theory of computation, see, e.g., Bernhardt (2016) and Appel (2014).
[3] For a short introduction to computer science, see Dasgupta (2016).

writing, and arithmetic. But what *is* computational thinking – in more concrete terms? Scholars have put forward different definitions for computational thinking (e.g., Barr & Stephenson 2011, CSTA 2016, Curzon & McOwan 2017), with no consensus so far (see, e.g., Selby & Woollard 2014). According to Curzon & McOwan (2017), computational thinking is a loose set of problem solving skills that mainly focus on the creation of algorithms - *algorithmic thinking*. Roughly speaking, an algorithm is an explicit, step-by-step procedure for answering some question or solving some problem. At present, algorithms play an extremely important role in the modern society and are affecting people's everyday lives; for example, they are the controls driving the engines of the Internet.[4] It is important to stress, however, that algorithmic thinking, and by implication, computational thinking, does not necessarily involve computers. For instance, one can devise an algorithm to escape from a dark maze (Vöcking et al. 2010). Besides algorithmic thinking, computational thinking comprises also computational modelling, scientific thinking, heuristics, logical thinking, pattern matching, representation, abstraction, generalization, understanding people, decomposition – divide and conquer, evaluation, and creativity. For detailed explanations of these terms, see Curzon & McOwan (2017).

As a final remark, it is a common mistake to think that computational thinking is trying to get humans to think like a computer. But according to Wing (2006), computational thinking is thinking like a computer scientist and is a way *humans* solve problems.

3. Pedagogical transformation of teaching computational thinking

> "Of course, we all have computers on our desks nowadays. We all use them for email, web browsing, word processing, game playing, etc. But the computational thinking revolution goes much deeper than that; it is changing the way we *think*."
>
> --- Alan Bundy (2007), p. 67 (Emphasis original)

Computational thinking is not the same as information and communication technology (ICT). The difference between them can be likened to the difference

[4] See MacCormick (2011) for a description of 9 revolutionary algorithms that have changed our world.

between programming a computer and using a computer. In more concrete terms, traditional courses on ICT are too often focused on the *use* of technology and the standard software (e.g., spreadsheets, word processors, databases, etc.) – the so-called ICT literacy or computer literacy, whose goal it is to educate informed consumers of technology. However, most students find the subject ICT rather dull and unchallenging since in this digital world with ubiquitous computing students can easily pick up the basics of ICT skills themselves. In contrast, computational thinking, being the problem solving skills drawn on the fundamental concepts and principles of computer science, stresses the underlying principles of computation and the *creation* of technology. Consequently, courses aiming at teaching computational thinking should go beyond ICT literacy, and this naturally calls for a different pedagogy. In a nutshell, to teach computational thinking we need to shift to solving problems beyond ICT and to promoting various types of thinking (e.g., algorithmic, logical, abstractive, creative, etc.) based on the concepts and principles fundamental to computer science.

So far, approaches to teaching computational thinking can be classified into three broad categories, depending on the role programming plays in it. In the first category, programming plays a major role in introducing computational thinking, and the programming languages adopted are usually full-fledged text-based ones like C/C++, Java, and Python. Two examples of courses taking this approach are Baldwin et al. (2017) and Benakli et al. (2017). The targets of these relatively demanding courses are usually science and engineering students. In the second category of approaches to teaching computational thinking, programming (or even computers themselves) plays no role in it. Computational thinking is taught through "unplugged" activities like puzzles, games, and magic (see, e.g., Curzon & McOwan 2017, Choi et al. 2017, Meyer III et al. 2014, Bell et al. 1998, Vöcking et al. 2010). The fact that computational thinking can be taught without programming or even computers is a profound one, as it shows that although computational thinking is abstracted from computer science, it is not inherently computer science and thus can be applied to other contexts as well. The third category of approaches to teaching computational thinking lies between the first and the second categories – it involves programming but not in a substantial way. The programming languages adopted in this approach are usually visual programming languages like Scratch (Resnick et al. 2009), Alice

(Cooper et al. 2000), Kodu (Stolee & Fristoe 2011), RAPTOR (Carlisle et al. 2005). Unlike those text-based programming languages, these visual programming languages, most of which are block-based, have "low barrier" because of their minimum syntax (Kelleher & Pausch 2005). This enables students to focus on the fundamental concepts and principles of computation. Examples of courses belonging to this category include Cortina (2007), Rizvi et al. (2011), Dierbach et al. (2011), Li & Wang (2012), and Kafura et al. (2015). These courses usually involve design-based learning activities such as robotics and computer games (Jun et al. 2017). This middle-way approach is by far the most effective way to teach computational thinking to non-science students.

4. Critical Discussion

"Computational thinking education is important because it is the foundation for innovation."

--- Marjorie Yang, CTE2017 Keynote Speech[5]

A number of reasons can be used to argue for incorporating computational thinking into our curriculum. First, the world has been changing increasingly into a digital one with ubiquitous computing. Technologies are no longer tools, but becoming an inseparable part of us, forming our extended cognition. Consequently, computational thinking is simply a new and fundamental way of thinking and problem solving in this digital world that every educated individual needs to learn. Second, computational thinking educates creative problem solvers and empowers them with the ability to create computational solutions with digital technologies, thereby enabling the transformation of a society of mere consumers of technology into one of potential developers of this technology (Athreya et al. 2017, Zhou 2017). Given that the curriculum in Hong Kong emphasizes the importance of students' creativity development, and that HKSAR Government, for economic reasons, encourages innovation and technological development, it is indisputable that we should incorporate computational thinking into our curriculum. Third, teaching thinking skills has long been regarded as an important aim of general education. Consequently, in our general education curriculum we have subjects like Creative and Critical

[5] Keynote speech "Why is Computational Thinking Education important as the Foundation for Innovation?" by Marjorie Yang at the International Conference on Computational Thinking Education 2017.

Thinking, Logic and Reasoning, and Design Thinking. By the same token, computational thinking, comprising all these skills and more, should deserve a more prominent place in our general education curriculum.

In recent years, many countries, recognizing the importance of computational thinking, have started to develop their computational thinking education. For example, as mentioned above, in 2014, UK introduced its new computing curriculum in schools. Core in this computing curriculum is the importance attached to computational thinking (Yadav et al. 2017). In mainland China, computational thinking education has received considerable attention. As early as in 2010, the Coalition of 9 Universities (C9) put forward a proposal for developing computational thinking in the general computing education in universities (Chen et al. 2013). Consequently, a new curriculum has been designed and launched, and new textbooks based on computational thinking have been written. In contrast, only in the last year did Hong Kong start to develop its computational thinking education. Gratifyingly, in spite of this short period of time, we have managed to set up two milestones. One is that a HK$216 million four-year project, branded CoolThink@JC,[6] which is funded by The Hong Kong Jockey Club Charities Trust and aimed at enhancing the computational thinking ability of upper primary school children in Hong Kong, was launched in November 2016 (Ko 2017). The other is that HKSAR Government has decided to incorporate computational thinking into the school curriculum (Education Bureau HKSAR 2016). In view of these milestones, we should be optimistic about the development of computational thinking education in Hong Kong, including its development in higher education.

5. Conclusion

Different ages of humanity have required different modes of thinking. Being a set of problem solving skills abstracted from computer science and useful in other contexts, computational thinking is the mode of thinking that is required in this digital age with ubiquitous computing. In this paper, I have traced the evolution of the idea of computational thinking, and argued that teaching computational thinking requires us

[6] http://www.coolthink.hk/

to go deeper than teaching information and communication technology (ICT). I also argued that since computational thinking can educate problem solvers with digital creativity, which is needed especially for innovation and technological development, computational thinking deserves a prominent place in our general education curriculum for all students. Finally, I pointed out that though Hong Kong has been lagging behind in developing computational thinking education as compared to mainland China and some other countries, we have reason to be optimistic about the development of computational thinking education in Hong Kong, including its development in higher education.

References

Appel, A. W. (2014). *Alan Turing's Systems of Logic: The Princeton Thesis*. Princeton University Press.

Athreya, B. H., & Mouza, C. (2017). Introduction to Thinking Skills for the Digital Generation. In *Thinking Skills for the Digital Generation* (pp. 1-10). Springer International Publishing.

Baldwin, D., Barr, V., Briggs, A., Havill, J., Maxwell, B., & Walker, H. M. (2017, March). CS 1: Beyond Programming. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 677-678). ACM.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community? *Acm Inroads*, *2*(1), 48-54.

Bell, T. C., Witten, I. H., & Fellows, M. (1998). *Computer Science Unplugged: Off-line activities and games for all ages*. Computer Science Unplugged.

Benakli, N., Kostadinov, B., Satyanarayana, A., & Singh, S. (2017). Introducing computational thinking through hands-on projects using R with applications to calculus, probability and data analysis. *International Journal of Mathematical Education in Science and Technology*, *48*(3), 393-427.

Bernhardt, C. (2016). *Turing's Vision: The Birth of Computer Science*. MIT Press.

Bundy, A. (2007). Computational thinking is pervasive. *Journal of Scientific and Practical Computing*, *1*(2), 67-69.

Carlisle, M. C., Wilson, T. A., Humphries, J. W., & Hadfield, S. M. (2005). RAPTOR: a visual programming environment for teaching algorithmic problem solving. *ACM SIGCSE Bulletin*, *37*(1), 176-180.

Chen, G. et al. (2013). Reform manifesto for the teaching of computational thinking. *China University Teaching*, 2013, Issue 07, pp.7-10.　(In Chinese: 计算思维教学改革宣言)

Choi, J., Lee, Y., & Lee, E. (2017). Puzzle based algorithm learning for cultivating computational thinking. *Wireless Personal Communications*, *93*(1), 131-145.

Computer Science Teachers Association (CSTA). (2016). Operational definition of computational thinking. Available at http://www.iste.org/docs/ct-documents/computational-thinking-operational-definition-flyer.pdf .

Cooper, S., Dann, W., & Pausch, R. (2000, April). Alice: a 3-D tool for introductory programming concepts. In *Journal of Computing Sciences in Colleges* (Vol. 15, No. 5, pp. 107-116). Consortium for Computing Sciences in Colleges.

Cortina, T. J. (2007, March). An introduction to computer science for non-majors using principles of computation. In *ACM SIGCSE Bulletin* (Vol. 39, No. 1, pp. 218-222). ACM.

Curzon, P., & McOwan, P. W. (2017). *The Power of Computational Thinking: Games, Magic and Puzzles to Help You Become a Computational Thinker*. World Scientific.

Dasgupta, S. (2016). *Computer Science: A Very Short Introduction*. Oxford University Press.
Denning, P. J. (2017). Computational Thinking in Science. *American Scientist*, *105*(1), 13-17.

Dierbach, C., et al. (2011). A model for piloting pathways for computational thinking in a general education curriculum. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (pp. 257-262). ACM.

Education Bureau HKSAR (2016, December). *Report on Promotion of STEM Education - Unleashing Potential in Innovation*. Available at http://www.edb.gov.hk/attachment/en/curriculum-development/renewal/STEM%20Education%20Report_Eng.pdf.

Fitelson, B., & Zalta, E. N. (2007). Steps toward a computational metaphysics. *Journal of Philosophical Logic*, *36*(2), 227-247.

Fluck, A. E., Webb, M., Cox, M. J., Angeli, C., Malyn-Smith, J., Voogt, J., & Zagami, J. (2016). Arguing for Computer Science in the School Curriculum. *Educational Technology & Society*, *19*(3), 38-46.

Gentle, J. E., Härdle, W. K., & Mori, Y. (Eds.). (2012). *Handbook of computational statistics: concepts and methods*. Springer Science & Business Media.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher*, *42*(1), 38-43.

Guzdial, M. (2008). Education Paving the way for computational thinking. *Communications of the ACM*, *51*(8), 25-27.

Isaacson, W. (2014). *The Innovators: The Innovators: How a Group of Hackers, Geniuses, and Geeks Created the Digital Revolution*. New York: Simon & Schuster.

Johnson, G. (2001). All science is computer science. *The New York Times*, *25*, 1-5.

Jun, S., Han, S., & Kim, S. (2017). Effect of design-based learning on improving computational thinking. *Behaviour & Information Technology*, *36*(1), 43-53.

Kafura, D., Bart, A. C., & Chowdhury, B. (2015, June). Design and Preliminary Results From a Computational Thinking Course. In *Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 63-68). ACM.

Karp, R. M. (2011). Understanding science through the computational lens. *Journal of Computer Science and Technology*, 26(4), 569-577.

Kelleher, C., & Pausch, R. (2005). Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers. *ACM Computing Surveys (CSUR)*, *37*(2), 83-137.

Ko, C. (2017, February 15). HKJC hones computational thinking skills with HK$216m fund. Available at http://www.coolthink.org/en/hkjc-hones-computational-thinking-skills-with-hk216m-fund/.

Li, T., & Wang, T. (2012). A Unified approach to teach computational thinking for first year non–CS majors in an introductory course. *IERI Procedia*, *2*, 498-503.

MacCormick, J. (2011). *Nine algorithms that changed the future: The ingenious ideas that drive today's computers*. Princeton University Press.

Markowetz, F. (2017). All biology is computational biology. *PLoS biology*, *15*(3), e2002050.

Meyer III, Edwin F., et al. *Guide to teaching puzzle-based learning*. London: Springer, 2014.

Mitkov, R. (Ed.). (2005). *The Oxford handbook of computational linguistics*. Oxford University Press.

Mohaghegh, D. M., & McCauley, M. (2016). Computational thinking: the skill set of the 21st century. In *International Journal of Computer Science and Information Technologies*, Vol. *7*(3), 1524-1530.

Navlakha, S., & Bar-Joseph, Z. (2011). Algorithms in nature: the convergence of systems biology and computational thinking. *Molecular systems biology*, 7(1), 546.

Papadimitriou, C. H. (2016). Computation as a Scientific Weltanschauung (Invited Talk). In *LIPIcs-Leibniz International Proceedings in Informatics* (Vol. 53). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc..

Resnick, M., et al. (2009). Scratch: programming for all. *Communications of the ACM*, 52(11), 60-67.

Rizvi, M., Humphries, T., Major, D., Jones, M., & Lauzun, H. (2011). A CS0 course using scratch. *Journal of Computing Sciences in Colleges*, *26*(3), 19-27.

Rosenbloom, P. S. (2012). *On computing: the fourth great scientific domain*. MIT Press.

Selby, C., & Woollard, J. (2014). Refining an understanding of computational thinking. *Author's Original*, 1-23.

Stolee, K. T., & Fristoe, T. (2011, March). Expressing computer science concepts through Kodu game lab. In *Proceedings of the 42nd ACM technical symposium on Computer science education* (pp. 99-104). ACM.

Sun, R. (Ed.) (2008). *The Cambridge handbook of computational psychology*. Cambridge University Press.

Vöcking, B., Alt, H., Dietzfelbinger, M., Reischuk, R., Scheideler, C., Vollmer, H., & Wagner, D. (Eds.). (2010). *Algorithms unplugged*. Springer Science & Business Media.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

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.

Yadav, A., Good, J., Voogt, J., & Fisser, P. (2017). Computational thinking as an emerging competence domain. In *Competence-based vocational and professional education* (pp. 1051-1067). Springer International Publishing.

Zhou, C. (Ed.). (2016). *Handbook of Research on Creative Problem-Solving Skill Development in Higher Education*. IGI Global.