Chapter 8

Digital Technologies and Numeracy—Synergy or Discord?

Kevin Larkin and Jodie Miller

Abstract Engaging young learners in STEM practices such as robotics and coding gives students the opportunity to use new and emerging technologies to solve problems while extending their own knowledge and understanding of mathematics. In Australia, a digital technologies curriculum was introduced in 2014 to assist with making connections between Technology and areas such as mathematics. Drawing on examples from Australia, British Columbia, the United Kingdom and New Zealand, this chapter examines how the introduction of a new curriculum intersects with existing curricula. As an example of an authentic activity that successfully combines elements of both curricula to support STEM learning, findings of a research project that has been conducted with Year 2 students ($n = 153$) from two Australian primary schools are presented. It appears as young students engage in robotics and coding (Technology) to learn mathematics concepts, they demonstrate learning that moves beyond their curriculum year level, creating a possible conflict between the digital technologies and mathematics curricula with their tightly prescribed sequence of content.

8.1 Introduction

The past decade has been characterised at all educational levels, from Early Childhood to University, by an international urgency to improve Science, Technology, Engineering and Mathematics (STEM) education in preparation for an increasingly scientific and technological society (Office of Chief Scientist, 2014). This push to improve STEM is often based on claims regarding the rapid decline in school students' engagement in STEM disciplines, e.g. Advanced Mathematics, Chemistry or Physics (Australian Academy of Science [AAS], 2016). Disengagement in STEM

K. Larkin (✉)
Griffith University, Gold Coast, Australia
e-mail: k.larkin@griffith.edu.au

J. Miller
University of Queensland, Brisbane, Australia
e-mail: Jodie.Miller@uq.edu.au

© Springer Nature Singapore Pte Ltd. 2020
A. MacDonald et al. (eds.), *STEM Education Across the Learning Continuum*,
https://doi.org/10.1007/978-981-15-2821-7_8

appears to begin at an early age (Larkin & Jorgensen, 2016; Moomaw & Davis, 2010), with many students from the upper primary years onwards performing poorly in, and disengaging from, mathematics (AAS, 2016), identified as the core subject in STEM (Finkel, 2017). To address a component of the broader STEM issue, a new Digital Technologies (DT) curriculum was introduced in Australia in 2014, to encourage, amongst other outcomes, the development of computational thinking in children across the formal years of schooling (F-10). Whilst, the convergence of policy and curriculum directions is heartening; it is also highly problematic as there is a limited evidence base to inform the implementation of STEM more generally in classrooms (English, 2016) and even less in relation to the implementation of DT as a discrete curriculum. Of particular concern is how the introduction of a new curriculum intersects with existing curricula. Does the new DT curriculum encourage opportunities for synergy with existing curricula, particularly mathematics, or is it yet another example of how new curricula are developed in isolation and when implemented, bring discord? Prior to examining this issue specifically as it relates to the Australian context, and before proposing an authentic classroom experience that combines elements of both curricula, it is opportune to consider whether, and then if necessary how, the issue of integrating digital technologies into existing curricula is dealt with internationally.

8.2 The International Context

In terms of an international perspective regarding DT curricula, we follow the methodology of Watson (2017) who used the curriculum of three English-speaking countries—Australia, New Zealand and the USA to investigate the teaching of Statistics. Thus, whilst acknowledging that curriculum documents in other countries may differ, for the purposes of this chapter we have focused on the curriculums, with explicit mention of digital technologies, in educational jurisdictions in four countries: Australia, British Columbia (Canada), the United Kingdom and New Zealand.

Least developed in terms of jurisdictions with DT explicit in the curriculum is British Columbia (Canada) where DT are included within a broader Applied Design, Skills and Technologies (ADST) curriculum envisaged as “an experiential, hands-on program of learning through design” (British Columbia Government, 2016, p. 1). The ADST is organised around three curricula competencies—Applied Design, Applied Skills and Applied Technologies—with technologies defined as “tools that extend human capabilities” (p. 1). While the ADST provides a breakdown of activities to be completed at different learning stages (K-3; 4-5; 6-7) these activities only minimally include DT and only then as a subset of the total range of technologies (both digital and nondigital) that children will encounter. Similarly to other educational jurisdictions such as California (USA), “the ADST Curriculum does not specify any content learning standards for Kindergarten through Grade 5 as “the intent is for teachers to use the Curricular Competencies from ADST K-5 with grade-level content from other areas of learning” (British Columbia Government, 2016, p. 4).

The United Kingdom is somewhat more explicit in recognising the role of DT in helping students understand their world (Department for Education, 2013). The National Curriculum identifies four aims for computing, two that mirror goals of the Australian DT curriculum (algorithmic thinking and writing computer programs) and two which relate more to the Australian ICT general capability (evaluating ICT and being ethically responsible users). A limited content base is provided for teachers at various Key Stages, e.g. at Key Stage One, broadly equivalent to F-2 in Australia, six dot points on content to be taught are prescribed (Department for Education, 2013), including again what would be considered as both DT curriculum and ICT general capability content in an Australian context. However, these dot points contain minimal information are therefore unlikely to assist teachers when planning to integrate DT and numeracy content.

With the recent release of the New Zealand DT curriculum (Ministry of Education, 2017), this jurisdiction most closely resembles the Australian context. The main modification to the previously existing Technology Learning Area is the addition of computational thinking for digital technologies. According to the new curriculum, “computational thinking enables students to express problems and formulate solutions in a way that a computer can be used to solve them. They also develop knowledge and skills in using different digital technologies to create digital content for the web, interactive digital platforms, and print” (Ministry of Education, 2017, p. 11). As was the case with British Columbia, no specific content is provided for teachers; however, very concise progress outcomes for key stages are mandated and within these outcomes, specific content can be found. For instance, at Progress Outcome Two (roughly Year 3 in Australia), students are required to write computer programs (using algorithmic thinking) such that computers can clearly follow them and then use these algorithms in an age-appropriate programming environment—e.g. *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group, & Playful Invention Company. [*ScratchJr*]), undated). Exemplars of student outcomes at various stages are also provided—(e.g. in Year 3 students programme a bee to visit various flowers to gather pollen using *ScratchJr*). So whilst no explicit content is provided, teachers can gain a sense of what they are to teach and can therefore, at least in principle if not in practice, begin to think about how this might be incorporated in existing numeracy activities. The brief account of what is occurring around the world indicates that, to varying degrees of specificity, DT is on the educational agenda in the three English-speaking jurisdictions discussed.

8.3 The Australian Context

Having briefly scanned the international educational context we now turn to the Australian context. In so doing, as was the case with Watson and Neal (2012) who examined only one curriculum sub-strand in one country, we make use of a number of specific instances of mismatches and missed opportunities in the two Australian Curriculums to reinforce the more generalised observation that national curriculum

documents from many countries have failed to capitalise on opportunities for synergy between digital technologies and numeracy education.

In 2014, The Australian Curriculum and Reporting Authority (ACARA) released the inaugural Technologies curriculum: subdivided into Design Technologies and Digital Technologies (previous to this Australia had a Technology Curriculum that incorporated both but did not specify digital content). Given that this article concerns connections among coding, robotics, numeracy and mathematics, we limit further discussion to the DT curriculum as it is presented in the Foundation to Year 6 (approximately 5–11 year olds) levels. The DT curriculum in primary school is divided into three bands (F-2; 3–4 and 5–6) and these bands are further subdivided into content descriptors under the two subheadings of *Knowledge and Understanding* and *Processes and Production Skills*. Readers familiar with the Australian mathematics curriculum will see some connections between the development of data concepts in both the DT and mathematics curricula. We will see later that at times this can be problematic. Within each subheading there are content descriptors and accompanying elaborations that provide further information related to the specific content descriptor. For example, in Year 4, under the *Knowledge and Understanding* subheading the students—*Examine how whole numbers are used to represent all data in digital systems* with one of the elaborations for this content descriptor being *explaining that binary represents the numbers using 1s and 0s and these represent the on and off electrical states, respectively, in hardware and robotics*.

8.4 Numeracy and the Digital Technologies Curriculum

An examination of the DT curriculum raises a number of issues in relation to the overarching goal of all curriculum areas to develop numeracy and numerate citizens (Australian Curriculum, Assessment and Reporting Authority [ACARA], 2016a). These issues fall broadly under the categories of: (1) the mismatch between numeracy understandings developed in the DT curriculum and how numeracy is developed in the mathematics curriculum; (2) superficial or inauthentic links to numeracy and (3) missed opportunities to engage with authentic numeracy opportunities.

8.4.1 Mismatches in Mathematical Understanding Promoted by the Two Curricula

Many of the mismatches between the two curricula occur in the F-2 section of the DT curriculum. As the mathematics curriculum is designed at a year level designation—i.e. Foundation is separate from Year 1, which is separate from Year 2; and the DT curriculum is organised across the three years, it is not readily apparent when in the

DT curriculum the various activities are to be completed. This adds to the potential confusion for teachers when implementing both curricula.

Within both curricula, content descriptors are coded. By way of example, *ACT-DIK002* is a shortcut for “Australian Curriculum Technology Digital Knowledge and Understanding Content Descriptor Two”. This particular content descriptor indicates that students in F-2 are to—*Recognise and explore patterns in data and represent data as pictures, symbols and diagrams*. It is not made clear what representing data using “diagrams” involves; however, this competency is likely to be beyond most early years students as the corresponding mathematics content descriptor only requires them to initially investigate simple yes/no questions (Year 1) and then represent simple categorical data using picture graphs (Year 1) or tally marks (Year 2). Within the overarching content descriptor, one of the elaborations requires students to discuss that the symbol 12 *may* represent different data to the symbol 21. Given that students at this age are beginning to develop conceptual understanding of place value, this elaboration is likely to cause confusion if the two representations are used interchangeably. A second DT elaboration at this level requires students to change pixel density to represent changes in data. Again, given that understanding pixels requires knowledge of scale and proportion, and may even require knowledge of decimals, this will very likely be beyond anything other than a superficial understanding by young students. Furthermore, the students are also required to link the resolution of the photograph to its file size, a task requiring knowledge of scale, which is a concept only developed in the final years of the primary school mathematics curriculum.

In the DT curriculum, students in F-2 are required to—*Collect, explore and sort data, and use digital systems to present the data creatively*. This content descriptor is problematic, as the elaboration requires students to locate and purposefully use visual or text data found from digital sources. This is in direct contradiction to the explicit emphasis in the mathematics curriculum on the use of primary (i.e. data collected by the students from their experience) rather than relying on secondary data (i.e. data from web sites) in the early and middle years of primary school. In addition, in relation to pictorial representation, in the F-2 DT curriculum elaboration, students are to use column graphs to represent different types of items; however, this form of representation is not introduced until at least Year 3 in the mathematics curriculum.

In a second example from F-2, students in the DT Curriculum are required to—*Follow, describe and represent a sequence of steps and decisions (algorithms) needed to solve simple problems*. Most of this descriptor and accompanying elaborations match with the respective mathematics content, except for the observation that students are to plan their sequences and steps using metric units—which are not introduced until Year 3 in the mathematics curriculum. The delay in using metric units in the mathematics curriculum is deliberate and based on the research literature regarding measurement (Clements & Sarama, 2014) and is an important step in the development of spatial reasoning and measurement.

Whilst overall reasonably accurate in terms of mathematical terminology, there are a few examples in the DT curriculum where careless use of mathematics terms may cause confusion. One example, in Year 3 and 4, is the content elaboration that includes, in part—*three circles, drawn as lines*. Given that lines are one dimensional

and circles are two dimensional and constructed from curves this is likely to be problematic. A second example occurs when children are asked to examine datasets regarding modes of travel—whilst mode can refer to type; its use in mathematics is normally reserved to refer to a measure of average.

It could, of course, be argued that students are capable of successfully completing many of the mismatched activities noted above; however, given that mathematics education experts designed the mathematics curriculum, it is highly likely that the mismatches generated from the DT curriculum side of the equation might be problematic for the numeracy development of young children. Given that both the numeracy general capability and the mathematics curriculum were in place well before the release of the DT curriculum, it is disappointing that these mismatches have occurred, as they are likely to cause some confusion for early years teachers responsible for delivering both curricula.

In addition, if we are to take ACARA at its word, the various individual curriculums are designed to work synergistically, especially when it relates to the development of the general capabilities, in this instance numeracy, which spans the entire range of curricula. It is thus disappointing that there are mismatches between the mathematics and DT curriculum in terms of developing numeracy. Furthermore, we shall shortly see that the DT curriculum, on some occasions, either treats numeracy in a superficial manner, or fails to take the opportunities provided to develop numeracy when this could easily have occurred.

8.4.2 Superficiality

As hinted above, in addition to mismatches between the DT and mathematics curricula in terms of content and language, of great concern is the generally superficial way that connections are made with the numeracy sub-elements of *Interpreting statistical information* and, to a lesser degree, *Using spatial reasoning*. Here we will focus only on the superficiality of the links from the DT curriculum to support the development of *Interpreting statistical information*. This numeracy element requires that students “gain familiarity with how statistical information is represented and to use data gathered in authentic contexts to solve problems” (ACARA, 2016a).

Superficiality in relation to statistics is prevalent across all three primary bands of the DT curriculum. In Year F-2 three of the elaborations accompanying the content descriptor—*Follow, describe and represent a sequence of steps and decisions (algorithms) needed to solve simple problems*—are linked to the *Interpreting statistical information* numeracy component and thus are meant to serve as exemplars of how this component can be developed. It is not immediately apparent to the authors how, for example—*scanning personal photographs or presenting a set of instructions or events in a slide show or instructions for recording a TV show or how their lunch order is delivered*—is in anyway a statistical understanding. Whilst there are certainly valuable learnings in terms of programming, it is inaccurate to imply that these develop students’ skills in interpreting statistical information.

Both the previous content descriptors are related to process; however, the superficial links to statistics are also evident in one of the knowledge and understanding content descriptors in Years 3–4—*Identify and explore a range of digital systems with peripheral devices for different purposes and transmit different types of data*. However, there is no evidence of statistics knowledge required in any of the three accompanying elaborations for this descriptor. In addition, in the descriptor—*data can be represented in different ways* with one elaboration indicating that *Identifying statistical information* is used when *recognising...waves for sound*. This is a complex scientific understanding, of little immediate relevance mathematically to students in Years 3 or 4, and unlikely to help them to become more numerate.

8.4.3 Missed Opportunities

In direct contrast to the superficiality of some links to numeracy elements presented in the DT curriculum, the reverse is also the case with many authentic opportunities for connection with numeracy components overlooked. In Year 3–4, *students are required to transmit different types of data* and the elaboration focuses on the device type (Interactive White Board [IWB], mobile); however, no links are made to the numeracy general capability. This is an ideal opportunity for students to transmit authentic data that they have collected in meaningful statistical activities, for an authentic purpose, e.g. to communicate to parents, peers or other schools. In addition, while we agree with the content descriptor—*students should be developing an understanding that data can be represented in a variety of digital forms*, it is a clear oversight not to link this elaboration back to numeracy learning outcomes given that numbers are forms of data. Problem-solving is a core component of the content descriptor—*Define simple problems, and describe and follow a sequence of steps and decisions (algorithms) needed to solve them*—and yet, despite problem-solving being one of the four overarching proficiencies of the Australian Curriculum: Mathematics—there is no link to either the numeracy general capability or the Problem-Solving Proficiency in the first elaboration where methods for solving problems are discussed. This would have been an ideal opportunity to connect with various problem-solving strategies promoted in the mathematics curriculum and vital for numeracy.

Our premises thus far have been that

1. In most international curricula, the role of DT has yet to be fully articulated and thus creating synergies between DT and mathematics is very difficult for teachers.
2. In Australia, and to a lesser extent the UK and New Zealand, the DT curriculum is more clearly articulated; however, there are numerous examples of content mismatches, lack of authenticity in connections that are made, and missed opportunities to engage students in learning incorporating both curricula.

Our conclusion is therefore that it is necessary to assist teachers in developing opportunities for authentic synergies between the two curricula, whilst still remaining

true to the intent of each of the individual curricula. In the remainder of this article, we outline how *ScratchJr*, a coding program for young children, offers some potential in bridging between the DT and the mathematics curricula to develop student's programming and algorithmic skills, and hence their overall numeracy, as required in both. Before looking at how this might occur, it is necessary to digress briefly to investigate the role of programming in schools in general, and the usefulness of *ScratchJr* in particular as a tool to support programming skills.

8.5 *ScratchJr* as a Programming Language for Young Children

ScratchJr (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) is a visual programming (henceforth coding) language developed by the Lifelong Kindergarten group at MIT Media Labs to promote creative thinking, reasoning and innovation (Resnick et al., 2009). The rich digital environment of *ScratchJr* utilises building block command structures to manipulate graphics, audio and video functions (Calder, 2012). The building block commands are forms of simplified syntax; hence students are not required to type the code themselves, rather they drag and drop the interlocking blocks of symbolic code together to create chains of code. There are ten categories of building block command structures, and each of these is represented by a specific colour. Examples of these categories include: motion blocks (blue); logic/control blocks (gold); and, data blocks (orange) (Francis, Khan, & Davis, 2016). Each coding block also incorporates text and symbolic commands to assist the user to select the appropriate code for the action that they would like to undertake. Similar to the *LOGO* turtle (Papert, 1980), the *ScratchJr* interface enables a cat to move on a two-dimensional screen. Figure 8.1 presents the *ScratchJr* interface with an example of one chain of code for drawing a hexagon.

8.6 Studies Focusing on Programming

A body of research indicates that coding provides an opportunity for developing students' cognition and mathematical knowledge (Papert, 1980). Noss and Hoyles (1996) state that "writing a computer program provides a broad canvas on which the learner can sketch half-understood ideas, and assemble on the screen a semi-concrete image of the mathematical structures he or she is building intellectually" (p. 55). Research into student learning through coding and programmable robots, including the use of *LOGO* (Clements, Battista, & Sarama, 2001) and *BeeBots* (Highfield, 2010), have indicated that programmable robots support students in problem-solving, reasoning and spatial concepts (Savard & Highfield, 2015), key components in the repertoire of a numerate person. In addition, findings from quantitative studies

Fig. 8.1 ScratchJr interface

have revealed that there is a correlation between computer coding using *Scratch* and mathematics outcomes for Year 5 students (Lewis & Shah, 2012). However, with the exception of research conducted using *LOGO*, research into coding is in its infancy, and where using *Scratch* as a programming language has been investigated at all, these investigations were in middle school contexts. Finally, Benton, Hoyles, Kalas and Noss (2017) stress that much of the past research into the impact of coding on students' numeracy development is inconclusive due to the diversity of research paradigms adopted in these studies.

Although there have been pockets of enthusiasm regarding the teaching of coding (e.g. *LOGO* and *BASIC*) in the last three decades; several factors have limited its wider adoption in school classrooms: students found mastering the programme syntax difficult; programing often had little connection to young people's interests (e.g. generating a sequence of primes); and there was limited expertise available to support teachers and students using either programming language (Resnick et al., 2009). Despite the fact that the teaching of coding is now mandated in the Australian Curriculum: Digital Technologies, the case can still be made that support for implementation remains problematic with a pessimistic view proposing that many generalist primary school teachers are underprepared to teach coding and therefore will likely have difficulty in establishing links between coding and numeracy (Benton et al., 2017). In addition to these problems is the observation that the underlying numeracy elements, potentially developed by students when they engage in coding, can remain hidden from teachers who, due to unfamiliarity with coding or coding programs, often over-focus on the use of the tool (either a visual coding program such as *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) or robots such as *BeeBots*) rather than the numeracy requirements within the tasks (Savard & Highfield, 2015).

8.7 Linking Programming and Numeracy in the Australian Curriculum

As outlined previously, coding is now an explicit content requirement, embedded in various degrees of complexity across the entire scope of the F-10 DT curriculum. Students are required to use “computational thinking and information systems, to define, design and implement digital solutions” (ACARA, 2016a) and by the end of Year 2, for example, students will “develop their design skills by conceptualising algorithms as a sequence of steps for carrying out instructions, such as identifying steps in a process or controlling robotic devices” (ACARA, 2016a). In determining the ways in which the DT curriculum relates to the development of the numeracy general capability, an important consideration is its alignment with the preexisting mathematics curriculum. Given that the remainder of this chapter concerns the use of *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) in a teaching experiment with Year 2 students, we will limit our focus to the relevant Year 2 mathematics and DT content descriptors, as displayed in Table 8.1.

The limited scope of studies focusing on the classroom and curriculum implementation of coding (Lye & Koh, 2014) supports our earlier argument of the need to examine the relationship between DT and mathematics curricula in supporting the development of the general capability of numeracy. As teachers use curricula to manage their planning, teaching and evaluating of student learning; making the mathematics in coding explicit for teachers is essential as this will encourage them to develop programs of work that encourage synergy between the DT and mathematics curricula rather than the discord suggested by our initial review. We now

Table 8.1 Mathematics and digital technologies content descriptors and elaborations

Subject	Content descriptors	Elaborations
Mathematics year 2: geometry	Describe and draw two-dimensional shapes, with and without digital technologies	Identify key features of squares, rectangles, triangles, kites, rhombuses and circles, such as straight lines or curved lines, and counting the edges and corners
Digital technologies foundation—year 2: process & skill production	Follow, describe and represent a sequence of steps and decisions (algorithms) needed to solve simple problems	Experimenting with very simple, step-by-step procedures to explore programmable devices, for example providing instructions to physical or virtual objects or robotic devices to move in an intended manner, such as following a path around the classroom

outline a component of a broader research project involving the teaching of coding and robotics to demonstrate how teachers can authentically integrate elements of the two curricula to achieve the curriculum intent of both without compromising the integrity of either.

8.8 The Teaching Experiment

A six-week coding and robotics teaching experiment was conducted with Year 2 students to explore how students developed mathematical reasoning (Steffe & Thompson, 2000) and numeracy as they participated in coding and robotics lessons. One of the researchers (author 2), in consultation with the class teacher, assumed the role of teacher in these experiments at both school sites. Of course, any teaching experiment is a point in time experience for the students and we, as previous primary school teachers ourselves, acknowledge that the following account does not critically analyse what impact the design of the lesson, or the teacher's way of teaching, had when discussing the students' test results. However, we are confident that classroom teachers could very effectively teach the lessons presented here.

The teaching experiment comprised of: pre-testing; 6×45 min lessons of either coding or robotics lessons (one lesson per week for six weeks with two groups of 10 students at each school site); and post-testing. Six classes of Year 2 students (age 7–8 years old) from two schools participated in the study. In total, there were 153 Year 2 students: 74 students from School A, and 79 students from School B. Both schools were matched for socio-demographic characteristics and are positioned just above the median for socio-educational advantage (School A = 1,056; School B = 1,037; ICSEA median value 1,000—*My School* website data, ACARA, 2019). The complete teaching experiment consisted of six lessons, three with a coding focus using *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) and three with a robotics focus using *LEGO Mindstorm* robots. Each lesson focused on teaching a mathematical concept using coding or robotics (e.g. drawing a shape using *ScratchJr*, or programming a robot to follow a particular path). Two video cameras were used to collect data during each lesson with one camera focussed on the researcher and one on a group of students. These video recordings were used for in depth analysis by the authors.

In order to identify their prior patterning and coding knowledge, each participant completed a pen and paper pre-test focussing on patterning (10 items) and *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) coding contexts (10 items). Patterning test items, validated in earlier research (Warren & Cooper, 2008; Miller, 2015), were modified to assess coding skills. Data from the pre-tests were analysed to determine a smaller, experimental group of students ($n = 40$) to participate in the coding and robotics lessons. Students were selected on their prior knowledge of patterning and coding (low-mid-high test scores). There were four subgroups of students identified: low patterning/low coding; low patterning/mid coding; mid patterning/low coding; and mid patterning/mid

Task	Mathematical concepts	Scratch coding concepts
Scratch Cat's favourite shape is a square. Can you help Scratch Cat draw a square?	Discussing properties of a square Testing and measuring lengths. How far is one step in Scratch? Testing and measuring Angles/turns Identifying any repeating patterns	Exploring the coding blocks and symbols used in scratch Measuring in pixels Building a code Running a code Editing a code if there are errors Using the repeat/loop function

Fig. 8.2 Drawing a square using *ScratchJr* with links to mathematics and coding concepts (Miller & Larkin, 2017)

coding. Although a possible outcome, no students were classified as high in either pattering or coding. Students who were not selected for the study ($n = 113$) stayed with their classroom teacher and participated in normal class lessons as planned by their teacher for that time. These teachers ($n = 6$) did not teach robotics or coding in their classrooms during the experiment. At the conclusion of the six weeks, post-testing (patterning and coding items) was conducted with all students ($n = 153$).

A two-stage research approach, using iterative refinement cycles for videotape analyses of changes in students' thinking (see Miller, 2015), was adopted to analyse the data from the teaching experiments (Lesh & Lehrer, 2000). First, the lesson video-footage was transcribed to capture students' verbal responses. These transcriptions were then analysed to consider emerging algorithmic thinking, computational thinking and numeracy development during the lesson sequence. Second, the student responses to the coding and robotics lessons were analysed to align the curriculum descriptors from both curricula. As the purpose of this article is to provide an example of how skills required in the two curricula can be developed simultaneously, we only present findings from the first lesson in the teaching experiment where students were required to draw a square using *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) (See Fig. 8.2). Further outcomes of the entire teaching experiment will be reported in full elsewhere.

8.9 Findings and Discussion

The findings are presented in two parts. Firstly, the emerging connections between mathematical thinking and coding evident in the students' response to the "Draw a square" task are discussed. Secondly, how this thinking is aligned to/misaligned with the content descriptors of each curriculum is reviewed. Each of the 40 students provided a response to the task. After analysis of the student responses, it was evident

that there were five common types of responses (see Table 8.2) to the “Draw a square” task.

When considering the student responses, in terms of the content descriptor requirements of the Year 2 mathematics and DT curricula (see Table 8.1), it is evident that some students were working at a higher level than required by either curriculum. There were three key insights from the data that demonstrated high levels of mathematical reasoning or programming: namely, working with 90° turns; orientation and perspective taking; and deducing a repeating pattern to provide a generalised code for making a square. When drawing a two-dimensional shape, using a digital tool such as *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated), an opportunity exists for students to engage with

Table 8.2 Student response and explanation, frequency of student responses, and student exemplars (Modified from Miller & Larkin, 2017)

Response and explanation	Frequency	Example
<p><i>I can't draw a square but I can draw a hexagon</i></p> <p>Student attempted to draw a square but used a number of 15° turns. Whilst not a hexagon, the majority of students said they were creating a hexagon. They clicked on the code four times to make this shape</p>	8
<p><i>I can draw stairs: Why is the cat not turning the right way?</i></p> <p>Students did not construct a square as they alternated the turns right and left</p>	4
<p><i>Is this still a square?</i></p> <p>Students correctly programmed the cat to draw a square, however, due to its nonstandard orientation, were unsure whether a square had been drawn</p>	3
<p><i>I made a square</i></p> <p>Students were able to program the <i>Scratch</i> cat to draw a square parallel to the bottom of the screen</p>	20
<p><i>I can see a pattern: move, turn, move...</i></p> <p>Students identified that they can see a repeating pattern and used the repeat coding block to program the <i>Scratch</i> cat to draw a square</p>	5

higher levels of mathematical reasoning or coding than required by their respective curricula. This may occur for three reasons: first, as a consequence of the visual programming language (icons) and representations; second, the perspective taking that performing the task requires; and, third the ability to easily create a coding chain that represents the algebraic structure of the physical shape they have drawn.

First, when using *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) to draw a square, it appears the language and representations depicted in the coding blocks require high levels of mathematical reasoning and knowledge, beyond the curricula requirements for these young students. For example, *ScratchJr* uses measures of degrees for turns, rather than mathematical language such as $\frac{1}{4}$ turn, and thus the software is, therefore, more sophisticated than programmable robotics toys, such as the commonly used (e.g. *BeeBots*) that merely use a left or right arrow to perform a 90° turn. By contrast, when using *ScratchJr*, students are required to program the *ScratchJr* cat to perform an accurate 90° turn using a variety of programmable blocks.

This is an example of where the coding structures available in the software can assist students to extend their mathematical reasoning beyond the strict requirements of the mathematics curriculum. For example, according to the Year 2 Mathematics curriculum, students are only required to identify turns using the language of “quarter and half turns” and it is not until Year 5 that students are required to understand angles in terms of degrees (Australian Curriculum, Assessment and Reporting Authority [ACARA], 2016b). Our initial supposition is that the representations and language in *ScratchJr* supports the development of higher levels of mathematical reasoning for these young students by incorporating the turn coding block. This, in turn, becomes part of the more substantive overall programming instructions for the “Draw a square” task using algebraic logic.

Second, the way in which students engage in the task of drawing a square using *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) is vastly different from drawing a square on paper using a pencil and ruler. As students code the *ScratchJr* cat to draw a square, they are required to take the perspective of the cat (i.e. the square will be drawn in the same orientation as the cat is initially facing). Students who started their cat either facing up or down, or an alternative sideways orientation other than the cat facing directly left or right of the screen (student’s perspective while looking at the screen), drew squares (if coded correctly) that looked different to the prototypical depictions of squares (parallel to bottom and sides of the screen).

This was evident in the student responses where they were unsure if they had still drawn a square for the reasons outlined above. While many students could code the *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) cat to draw a square, and thus meet DT curriculum outcomes, their limited mathematical understanding of “squareness” meant they were unable to reason if their shape was a square or not. This has implications in terms of a potential mismatch between the requirements of the DT curriculum—i.e. *program a robot to follow a path*—and spatial reasoning requirements of the mathematics

curriculum where rotations and translations, implicit in this coding, are not developed until Year 5.

Finally, unlike the physical drawing of a square using a pencil and ruler, some students could see on the screen the “structure” of the square, i.e. the semi-concrete mathematical structure (Noss & Hoyles, 1996) of their drawing in the *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) programming code. This led five students, unprompted by the researcher, to identify units of repeat (e.g. move 100 steps, turn 90°) and then deducing that their code (move 100 steps, turn 90°, repeat four times) would draw a square. While, students in Year 2 should be able to identify a repeating pattern, this moves well beyond the typical linear patterns presented to students (e.g. ABAB). This led to students then deducing a generalisation for the perimeter (e.g. move n length, turn 90° and repeat four times) of the square and even further to discussions about measuring the perimeter of squares using the code (e.g. if my square has a length of 10 steps, the total perimeter will be 40 steps).

We suggest that these students were demonstrating early algebraic thinking (deducing patterns) and higher levels of spatial understanding and measurement (identifying the perimeter), beyond the current required curriculum standard (e.g. Year 6—*Continue and create sequences involving whole numbers, fractions and decimals and describe the rule used to create the sequence*; and Year 5—*Calculate perimeter and area of rectangles using familiar metric units* (ACARA, 2016b). Again, the mismatch between the curricula can be seen as a double-edged sword. In coding the *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) cat some students could arguably be seen as demonstrating very advanced mathematical thinking; however, at the same time, other students who managed the coding requirements of the task were clearly incapable of understanding the mathematical requirements of the task. The numeracy development of this latter group of students may be hampered if they are required to make connections between the simplified coding environment in *ScratchJr* with the much more complicated spatial reasoning requirements of the task.

8.10 Conclusion and Implications

The implementation of the DT curriculum in the lower primary years presents a challenge for generalist, early primary school teachers; both in terms of their discrete knowledge of programming and, perhaps more importantly, in terms of integrating this content within existing curriculum demands. This article has demonstrated that the two curricula in question, i.e. the DT and mathematics curricula, have not been as well matched as is necessary to fully develop synergies between the two. Whilst it is acknowledged that the Australian DT curriculum provides a much deeper level of content support than any of the other international curricula investigated in this research, this heightened level of explicitness is problematic when the more explicit content contradicts with explicit content in existing curricula, especially the

mathematics curriculum. These contractions, missed opportunities and superficial connections result in the underdevelopment of the numeracy general capability that is meant to be an overarching concern of both.

In regard to coding more specifically, much of the past research in this area has only provided limited insights into how students use programming in the early and middle years of primary school, and even this research is inconclusive (Benton et al., 2017). This article adds to the current literature by indicating how the use of *ScratchJr* (DevTech Research Group, Lifelong Kindergarten Group & Playful Invention Company, undated) can assist teachers to support the development of programming knowledge in young students in such a way as to remain faithful to both curricula. Extant research with primary school students, when using robotics programs, have identified that young students develop numeracy skills as they use reasoning to problem solve (Savard & Highfield, 2015), in coding contexts.

Our early conjecture is that coding also provides an opportunity to identify and deduce patterns and is, therefore, a useful platform to encourage engagement with early algebraic thinking. Given the limited studies focusing on the classroom implementation of programming (Lye & Koh, 2014), and no published critical analysis of the synergy or otherwise between the DT and mathematics curricula in the Australian context, this research is an important contribution to the discourse. Although limited in scope, this research is an example of how the thoughtful combination of elements of both curricula can maximise opportunities for primary students to develop numeracy competencies requiring understanding of programming in authentic mathematics contexts.

Acknowledgements The authors acknowledge the support from the schools, teachers and students involved in the study. Aspects of the teaching experiment component of this article were presented at a MERGA conference; however, this article is a substantive development of this earlier work.

References

- Australian Academy of Science. (2016). *The mathematical sciences in Australia: A vision for 2025*. Canberra: Retrieved from <https://www.science.org.au/files/userfiles/support/reports-and-plans/2016/mathematics-decade-plan-2016-vision-for-2025.pdf>.
- Australian Curriculum, Assessment and Reporting Authority (ACARA). (2016a). *The Australian curriculum: Digital technologies*. Canberra: ACARA.
- Australian Curriculum, Assessment and Reporting Authority (ACARA). (2016b). *The Australian curriculum: Mathematics*. Canberra: ACARA.
- Australian Curriculum, Assessment and Reporting Authority (ACARA). (2019). *My school*. Retrieved from <https://www.myschool.edu.au/>.
- Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2017). Bridging primary programming and mathematics: Some findings of design research in England. *Digital Experiences in Mathematics Education*, 3(2), 115–138. <https://doi.org/10.1007/s40751-017-0028-x>.
- British Columbia Government. (2016). *Applied design, skills and technologies*. Retrieved from <https://curriculum.gov.bc.ca/curriculum/applied-design-skills-and-technologies/introduction>.

- Calder, N. (2012). The layering of mathematical interpretations through digital media. *Educational Studies in Mathematics*, 80(1–2), 269–285.
- Clements, D. H., Battista, M. T., & Sarama, J. (2001). Logo and geometry. *Journal for Research in Mathematics Education. Monograph*, 10, 1–177.
- Clements, D. H., & Sarama, J. (2014). *Learning and teaching early math: The learning trajectories approach*. New York: Routledge.
- Department for Education. (2013). *National curriculum in England: Computing programmes of study*. Retrieved from <https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study/national-curriculum-in-england-computing-programmes-of-study>.
- DevTech Research Group, Lifelong Kindergarten Group, & Playful Invention Company. (undated). *ScratchJr* [Coding for young children]. <https://www.scratchjr.org/>.
- English, L. D. (2016). STEM education K-12: Perspectives on integration. *International Journal of STEM Education*, 3(3). <https://doi.org/10.1186/s40594-016-0036-1>.
- Finkel, A. (2017). Plenary address. 40 years on: We are still learning. *Proceedings of the 40th Annual Conference of the Mathematics Education Research Group of Australasia*. Melbourne: MERGA.
- Francis, K., Khan, S., & Davis, B. (2016). Enactivism, spatial reasoning and coding. *Digital Experiences in Mathematics Education*, 2(1), 1–20.
- Highfield, K. (2010). Robotic toys as a catalyst for mathematical problem solving. *Australian Primary Mathematics Classroom*, 15(2), 22–27.
- Larkin, K., & Jorgensen, R. (2016). ‘I Hate Maths: Why Do We Need to Do Maths?’ Using iPad video diaries to investigate attitudes and emotions towards mathematics in year 3 and year 6 students. *International Journal of Science and Mathematics Education*, 14(5), 925–944. <https://doi.org/10.1007/s10763-015-9621-x>.
- Lesh, R., & Lehrer, R. (2000). Iterative refinement cycles for videotape analyses of conceptual change. In R. Lesh & A. Kelly (Eds.), *Research design in mathematics and science education* (pp. 665–708). Hillsdale, NJ: Erlbaum.
- Lewis, C. M., & Shah, N. (2012). Building upon and enriching grade four mathematics standards with programming curriculum. *Proceedings of the 43rd Association of Computing Machinery Technical Symposium on Computer Science Education* (pp. 57–62). ACM.
- Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51–61.
- Miller, J. (2015). Young Indigenous students’ engagement with growing pattern tasks: A semiotic perspective. In M. Marshman, V. Geiger, & A. Bennison (Eds.), *Mathematics education in the margins. Proceedings of the 38th annual conference of the Mathematics Education Research Group of Australasia*. Sunshine Coast: MERGA.
- Miller, J. & Larkin, K. (2017). Using coding to promote mathematical thinking with year 2 students: Alignment with the Australian curriculum. *40 Years on: We are still learning. Proceedings of the 40th Annual Conference of the Mathematics Education Research Group of Australasia*. Melbourne: MERGA.
- Ministry of Education. (2017). *Digital technologies—Hangarau Matihiko*. Retrieved from <https://education.govt.nz/assets/Documents/Ministry/consultations/DT-consultation/DTCP1701-Digital-Technologies-Hangarau-Matihiko-ENG.pdf>.
- Moomaw, S., & Davis, J. (2010). STEM comes to preschool. *Young Children*, 65(5), 12–14.
- Noss, R., & Hoyles, C. (1996). *Windows on mathematical meanings: Learning cultures and computers*. London: Kluwer.
- Office of Chief Scientist. (2014). *Science, technology, engineering and mathematics: Australia’s future*. Canberra. Retrieved from http://www.chiefscientist.gov.au/wp-content/uploads/STEM_AustraliasFuture_Sept2014_Web.pdf.
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books.
- Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., et al. (2009). Scratch: Programming for all. *Communications of the ACM*, 52(11), 60–67.

- Savard, A., & Highfield, K. (2015). Teachers' talk about robotics: Where is the mathematics? In M. Marshman, V. Geiger, & A. Bennison (Eds.), *Proceedings of the 38th Annual Conference of the Mathematics Education Research Group of Australasia*. Sunshine Coast: MERGA.
- Steffe, L. P., & Thompson, P. W. (2000). Teaching experiment methodology: Underlying principles and essential elements. In R. Lesh & A. E. Kelly (Eds.), *Research design in mathematics and science education* (pp. 267–307). Hillsdale, NJ: Lawrence Erlbaum Associates.
- Warren, E., & Cooper, T. (2008). Patterns that support early algebraic thinking in the elementary school. In C. Greenes & R. Rubenstein (Eds.), *Algebra and algebraic thinking in school mathematics: Seventieth Yearbook* (pp. 113–126). Reston, VA: National Council of Teachers of Mathematics.
- Watson, J. (2017). Linking science and statistics: Curriculum expectations in three countries. *International Journal of Science and Mathematics Education*, 15(6), 1057–1073. <https://doi.org/10.1007/s10763-015-9673-y>.
- Watson, J., & Neal, D. (2012). Preparing students for decision-making in the 21st century- statistics and probability in the Australian curriculum: Mathematics. In B. Atweh, M. Goos, R. Jorgensen & D. Siemon (Eds.), *Engaging the Australian national curriculum: Mathematics—Perspectives from the field*. Mathematics Education Research Group of Australasia (pp. 89–113).