A Scoping Review of Studies on Computational Thinking in K – 12 Mathematics Classrooms.

Abstract:

Since the 1960s, a few, yet very influential, educational researchers have investigated how computer programming can be used to foster mathematics learning. However, since the term 'computational thinking' was popularised by Jeannette Wing in 2006, the number of studies in this area has grown substantially. In this paper, we present a systematic analysis of literature linking mathematics education to computational thinking in an attempt to quantify the breadth and depth of existing work in the area. Our analysis indicates that many studies: (1) originate from computer science academics rather than education experts, (2) involve mathematics but mainly concentrate on teaching programming skills, (3) present small-scale research designs on self-reported attitudes or beliefs, and (4) rarely deal with concepts in mathematical domain areas such as probability, statistics, measurement and functions. Thus, we conclude that there are opportunities for rigorous research designs reporting on observable learning outcomes, explicitly targeting mathematics, conducted by multidisciplinary teams, and focusing on less-explored domain areas. We believe that these opportunities should be investigated to in order to provide a broader evidence-base for developing meaningful digital learning experiences in mathematics for school-aged children.

Keywords: computational thinking, programming, mathematics, digital technologies

Introduction

Computational Thinking

Seymour Papert introduced the term *computational thinking* in the book *Mindstorms: Children, computers, and powerful ideas* (Papert 1980). In *Mindstorms*, Papert theorised that students learn most effectively when they are engaged in the construction of something that is meaningful to them and that can be shared. Papert, as a mathematician, was particularly interested in the teaching of mathematics. He envisioned a learning environment, which he referred to as "Mathland", that students could use to explore abstract mathematical concepts in a concrete way. This idea led him to develop the programming language Logo, which he described as "an instrument designed to help change the way you talk about and think about mathematics and writing and the relationship between them." (Papert 1990, pg. 7). Papert envisioned that students programming in Logo would be able to develop their understanding of learning and thinking through the process of testing and debugging their ideas in code. This vision resonated with some teachers and researchers, who saw

Logo and Papert’s philosophy of education as an “alternative to the prevailing technocentric and behaviourist notions of computer-aided instruction” (Agalianos et al. 2006, pg. 241), that were common in the 1980s.

During the 1980s some teachers were inspired by Papert’s *Mindstorms* to make changes to their teaching approaches, and soon it became part of national educational reforms in both the USA and the UK (Agalianos et al. 2006). Initially, some teachers were inspired to make changes to their teaching approaches at a grass roots level, but in the years following it became part of national educational reforms in both countries. Microcomputers were also introduced into many classrooms in the UK during the 1980s, which Agalianos et al. (2006) argued was largely motivated by industrialists, rather than in the spirit of a true educational reform. Thus the main motivations for this reform in the UK appeared to be supporting the British computer industry and increasing the UK’s international competitiveness. The view that computers could be used by students to explore ideas as Papert envisioned was largely absent. Furthermore, the versions of Logo that were compatible with the most common microcomputer used in classrooms, the BBC Micro, were difficult to install and had limited functionality. Ultimately, Logo was “stripped of its radical potential” (Agalianos et al. 2006, pg. 241), and for many teachers Logo became synonymous with ‘turtle graphics’. Consequently, in both the UK and the USA, teachers mainly used Logo as an activity that was carried out in the classroom, rather than as an environment within which students could develop their thinking.

Almost thirty years after *Mindstorms*, Jeanette M. Wing wrote an article on computational thinking, which she defined as “thinking like a computer scientist” (Wing 2006, pg. 35). Wing argued that computational thinking was a skill that could benefit everyone, not just computer scientists and that it should be taught alongside “reading, writing and arithmetic” (Wing 2006, pg. 33). In the decade following Wing’s article, the teaching of computer science (CS) became more widespread and it began to be introduced into compulsory K – 12 curricula (Wing 2016). Countries that have recently introduced CS curricula in K – 12, with a focus on computational thinking, include Australia (Falkner et al. 2014), England (Brown et al. 2014), the United States (Fisher 2016) and New Zealand (Bell et al. 2012a).

Programming and computational thinking are generally considered to be separate skills, but programming requires the use of computational thinking and is often used to teach it (Lye and Koh 2014). Programming is the act of writing code that instructs a computer to perform some actions, whereas computational thinking is a “problem solving methodology” (Barr and Stephenson 2011, pg. 115). The teaching of computational thinking does not necessarily require students create programs, however. For example, CS Unplugged is a set of resources that have been developed to teach this skill without the use of a computer (Bell et al. 2012b).

Introducing programming and computational thinking into compulsory K – 12 classes has presented some challenges for educators, as the learning of these skills has not usually been part of teachers’ formal education. This is particularly true for primary school teachers, as they are unlikely to have completed a technology major and are usually generalist teachers (Vivian et al. 2014). One of the approaches suggested for helping prepare teachers for teaching these concepts is to integrate computational thinking into the subjects they are currently teaching, for example, mathematics (Barr and Stephenson 2011).

Existing Reviews of Computational Thinking Literature

Reviews of the computational thinking literature have been conducted in the past but none of these reviews have been focused on finding links between computational thinking and the learning of mathematics in K – 12 Education. Lye and Koh (2014) searched 2 educational research databases for studies that were related to computational thinking in education and, after discarding non-empirical and irrelevant studies, selected 27 for review. Lye and Koh (2014) were interested in how computational thinking (through the use of programming) had been incorporated into K – 12 curricula, the reported performance outcomes of the studies' participants and the types of interventions that had been used in the reviewed studies. However, this review did not provide any information on how computational thinking had been linked to mathematics learning outcomes. Of the 27 studies selected for review, nine of these had study participants who were K – 12 students and only two of these nine studies involved integration of computational thinking and mathematics. Furthermore, the performance outcomes that Lye and Koh (2014) reported were related to the computational thinking dimensions defined by Brennan and Resnick (2012) - computational concepts, computational practices and computational perspectives - and not subject content knowledge.

In another pertinent study, Kalelioglu et al. (2016) searched six databases, four of which were multidisciplinary research databases and two of which were CS research databases, to find all computational thinking studies published and indexed in these databases between 2006 and 2014 and developed a framework from a review of the studies found in the search. In this systematic review, Kalelioglu et al. (2016, pg. 586) classified the selected 125 papers according to their: "purpose", "targeted population", "emphasised theoretical/conceptual backgrounds", "suggestions definitions" (of computational thinking), "chosen framework/scope", and the type of paper and "employed research design". Of the selected 125 papers, 47 had K – 12 students as the targeted population and a small amount (2%) of all the papers reviewed included a definition of computational thinking that included mathematical reasoning as an aspect of this term. The links between computational thinking and mathematical reasoning were not considered in this systematic review. Kalelioglu et al. (2016) concluded that the computational thinking literature is "at an early stage of maturity" and that the most of the studies they reviewed did not have "research designs" (p. 591), perhaps trying to note a perceived lack of well-designed methodologies. Similarly, Falkner et al. (2014) conducted a semi-systematic review of CS education literature, concluding that rigorous research was lacking in their results and found that most studies took place outside of classrooms. Falkner et al. (2014) searched two databases, ACM Digital Library and Google Scholar, for studies published between 2003 and 2013 on Computer Science Education in K - 12 and found a total of 71 studies. The classifications of the studies included: the type of research methods used, the context of the study and the presence of concepts from Australia's Digital Technologies curriculum (the equivalent of England's Computing Curriculum). Four of the 71 studies reviewed were classified as being conducted in the context of learning mathematics but it was not clear what the research methods or assessment approaches were in these studies. Falkner et al. (2014, pg. 9) also discussed the potential for research into CS education that involved ideas that have been previously studied in mathematics education, such as "gender-based stereotypes and achievement". These reviews all found a dearth of empirical research providing evidence of the transfer effects of programming to the learning of 21st century skills (Scherer 2016).

None of the existing reviews of the computational thinking literature have focused on how computational thinking and the learning of mathematics have been linked.

Limitations of this study

As a team of mathematics educators and computer scientists, we are interested in the links between computational thinking and the learning of mathematics in K – 12 Education, and in this study, we review existing literature to discover how these links have occurred. However, we acknowledge that using the term "computational thinking", misses important research that was conducted prior to the introduction and popularisation of this term. Essential foundational research exploring the use of programming for the teaching of mathematics from the 1980s, expanding on Papert's work with Logo, but conducted prior to Wing (2006), will not be present in this review.

Reports on projects that used computer programming to help children explore mathematics dating back to the 1960s have not been captured in the review presented in this paper (Feurzeig et al. 1969; Papert 1972). Also missing from our study is extensive research conducted in the 1980s and 1990s in England and the United States. Most of this research was conducted by proponents of Constructionism, a learning theory developed by Papert that built on Piaget's Constructivism (Ackermann 2001). For instance, in the United Kingdom, Hoyles and Noss (1992) explored the use of Logo programming and its potential integration into K – 12 mathematics education. In the United States, Kafai and Harel (1991) investigated the learning outcomes of students who developed and designed computer games to teach their peers mathematics concepts. Wilensky and Resnick (1999) created StarLogo, a software environment for creating agent-based simulations inspired by Logo, and studied the use of this software for computational modelling in K – 12 mathematics and science education. These researchers have continued to work on research projects that do combine these, such as ScratchMaths (Benton et al. 2016, 2017), constructionist gaming (Kafai and Burke 2015) and NetLogo (Tisue and Wilensky 2004).

Outside of the work conducted by these researchers, however, it is not clear how computational thinking and the learning of mathematics are being linked in the wider literature. Some researchers, for example (Barr and Stephenson 2011; Weintrop et al. 2016), have suggested approaches for integrating computational thinking with existing K – 12 curricula, including mathematics. The scoping literature review reported in this paper sheds light on all existing work depicting this integration since 2006.

Research Questions

The aim of this study is to answer the following research questions:

1. What peer-reviewed studies have been published from 2006 to 2016 in relation to computational thinking in K-12 educational contexts?
2. Do these studies link computational thinking to the learning of mathematics, and if so, in what ways?

Methodology for Scoping Review

The approach for this scoping review was designed according to methods discussed by Arksey and O'Malley (2005) who identified four common reasons for conducting a scoping review study. The reasons for conducting this scoping review are "to examine, the extent, nature and range of research activity" and "to identify research gaps in the existing literature" (Arksey and O'Malley 2005, pg. 6-

7). In this section the process for searching the databases is explained followed bythe methods used for classifying the papers for analysis.

Search Results

Six databases (**Error! Reference source not found.**) were searched to find peer-reviewed studies relevant to computational thinking in K – 12 Education. Four of these databases were multidisciplinary databases: Springer, Proquest, ScienceDirect and EBSCO Megafile Premier. These databases included results from Education databases, for example Proquest includes results from the Education Resources Information Center (ERIC) database, as well as STEM and Computer Science databases. The other two databases searched, IEEE Xplore and ACM Digital Library, are focused on Computer Science and Software Engineering.

The search term used on each database was *"computational thinking" AND "school*" AND ("Primary" OR "Secondary" OR "High" OR "K-12")*. The results were limited to peer reviewed articles published between 2006 and 2016. 2006 was chosen as the start date, as this was when Wing (2006) published her first article on computational thinking.

Table 1: Number of results from each selected database

Database	Number of Results
IEEE Xplore	253
Springer	236
Proquest	129
ACM Digital Library	104
ScienceDirect	69
EBSCO Megafile Premier	34
Total	***825***

After exporting the results into an Endnote library, studies that were irrelevant to computational thinking in K – 12 Education were excluded and annotated. The steps are shown in Figure 1, and can be summarised as follows. Firstly, duplicate studies (n = 66), resulting from the search being conducted across different databases, were removed. We also removed studies (n = 121) that did not have full text available when the search was conducted. Next, the titles and abstracts of the remaining studies were screened for relevance, and were removed if they did not relate to computational thinking in K – 12 Education. Following the screening of titles and abstracts, full texts of all remaining studies were screened to remove irrelevant articles (n= 283). Studies that were relevant to computational thinking in K – 12 Education but that were works-in-progress or that were not written in English were also removed. After this process, 339 papers were deemed to be relevant to computational thinking in K – 12 Education. The process is summarised in Figure 1**Error! Reference source not found.**.

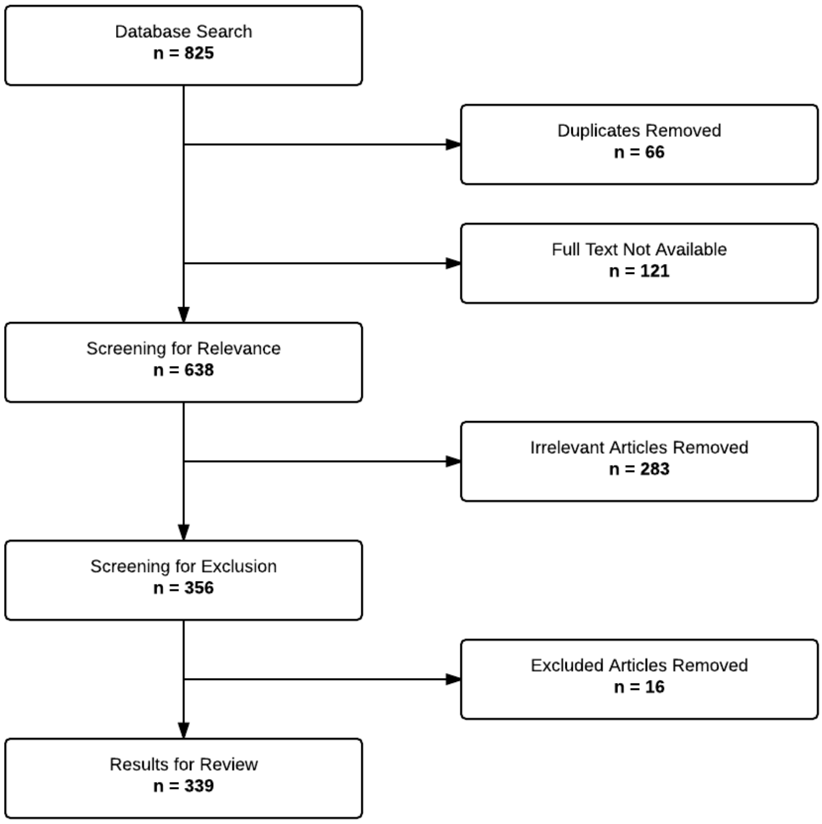

Figure 1: The review process

To verify that the study could be reproduced and to ensure validity, two researchers' reviews of the studies were compared. Initially, one of the researchers conducted the review process on the 638 results, after removing duplicate and studies for which there was no full text available. One of the authors reviewed 10% (n = 60) of the results and the classifications were compared. After a first round of coding, 78% inter-rater reliability was achieved and it became apparent that some areas of coding needed clarification. This was accomplished and a second iteration of coding was undertaken. On the second iteration a 100% inter-coder agreement was achieved, which was considered appropriate for the study.

Classification of Studies

The remaining studies (n=339) were classified according to their *Link to the Learning of Mathematics*, *Activity Approach* and *Mathematics Domain Area*. The 4-step approach for classifying the studies is explained in the following subsections.

Step 1: Link to the Learning of Mathematics

The studies were first classified according to their type of *Link to Learning of Mathematics* as: *Not at All* linked, *Incidentally* linked or *Explicitly* linked (see Figure 2). Studies that were incidentally linked to the learning of Mathematics (n=115) were those where mathematics concepts were present but where there was no evidence that the researchers had intended for students to learn these concepts. For instance, some studies involved the design and creation of games, with visual programming tools such as Scratch, that required the use of coordinate geometry (Akcaoglu 2014; Pinto and Escudeiro 2014; Alexander Repenning et al. 2010), but the researchers did not explicitly set out to teach this concept. Studies that were explicitly linked to the learning of mathematics (n=89) were those in which researchers had made a clear link between the learning of at least one mathematics concept and computational thinking. For example, Mensing et al. (2013) described several strategies for integrating computational thinking into the US mathematics curriculum. The rest of the studies (n=135) were removed for consideration in the subsequent classification process, as they were found to have no links to the learning of mathematics.

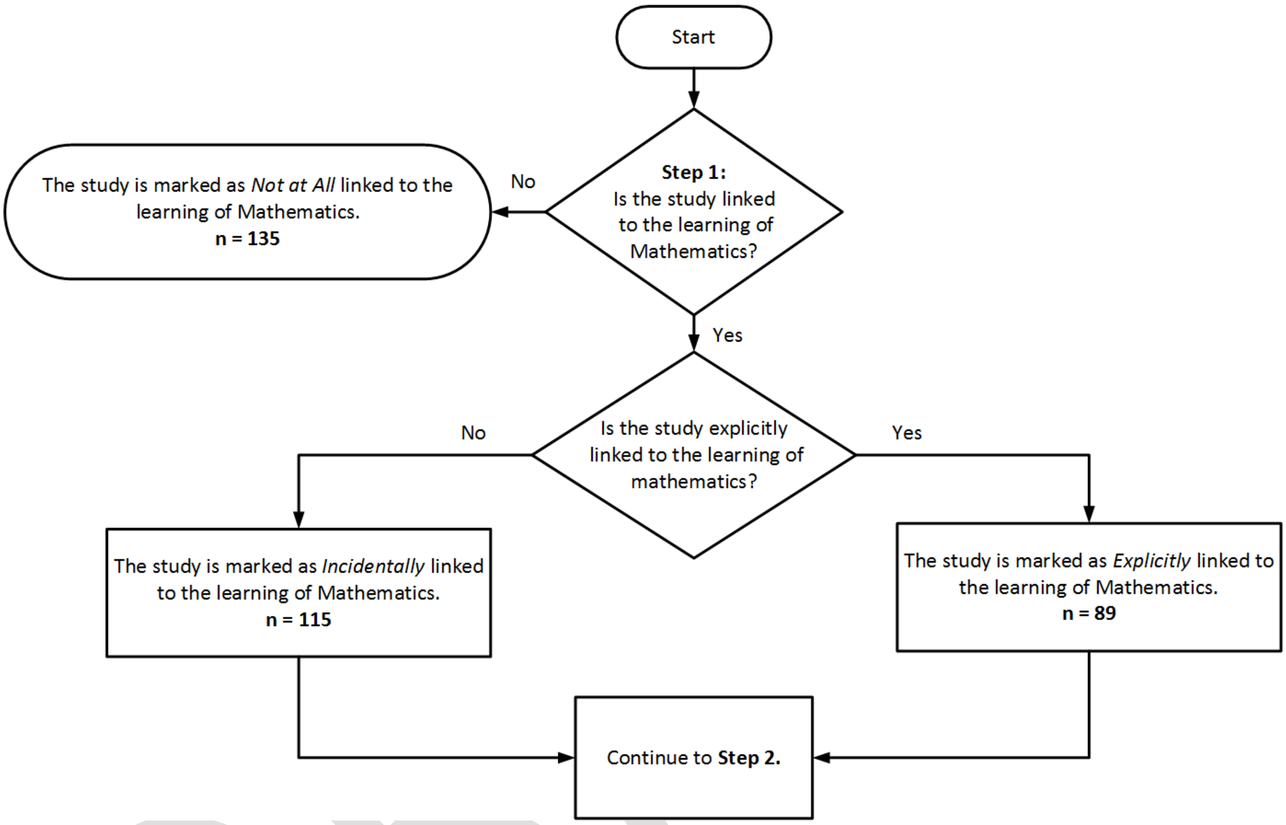

Figure 2: Step 1 in the classification process

Step 2: Classification of Mathematics Domain Area

Studies that were explicitly or *incidentally* linked to the learning of Mathematics were further classified by the *Mathematics Domain Area* present in them. Mathematics concepts were considered present in the studies if there was some evidence of the teaching of, or intention to teach, the concept through the use of computational thinking or programming. The mathematics concepts were grouped into the five following domain areas:

- **Numbers and Operations** (n=168)**:** Counting, operations, number systems and fractions
- **Algebra** (n=157)**:** Abstraction of concepts from *Numbers and Operations* and equations
- **Measurement and Functions** (n=58)**:** Ratios, proportional, linear, non-linear relationships not including trigonometric functions
- **Geometry** (n=117)**:** Shapes, Cartesian coordinates and area
- **Statistics and Probability** (n=30): Data, its measurement and representation

Step 3: Nature of the Study

This step, illustrated in Figure 3, involved checking whether the study was empirical in nature. Studies were thus marked as non-empirical or empirical, and in the latter case we noted the size of the sample and the methodology used. In the case of studies dealing explicitly with teaching mathematics, the methodology was coded as quantitative, qualitative or mixed.

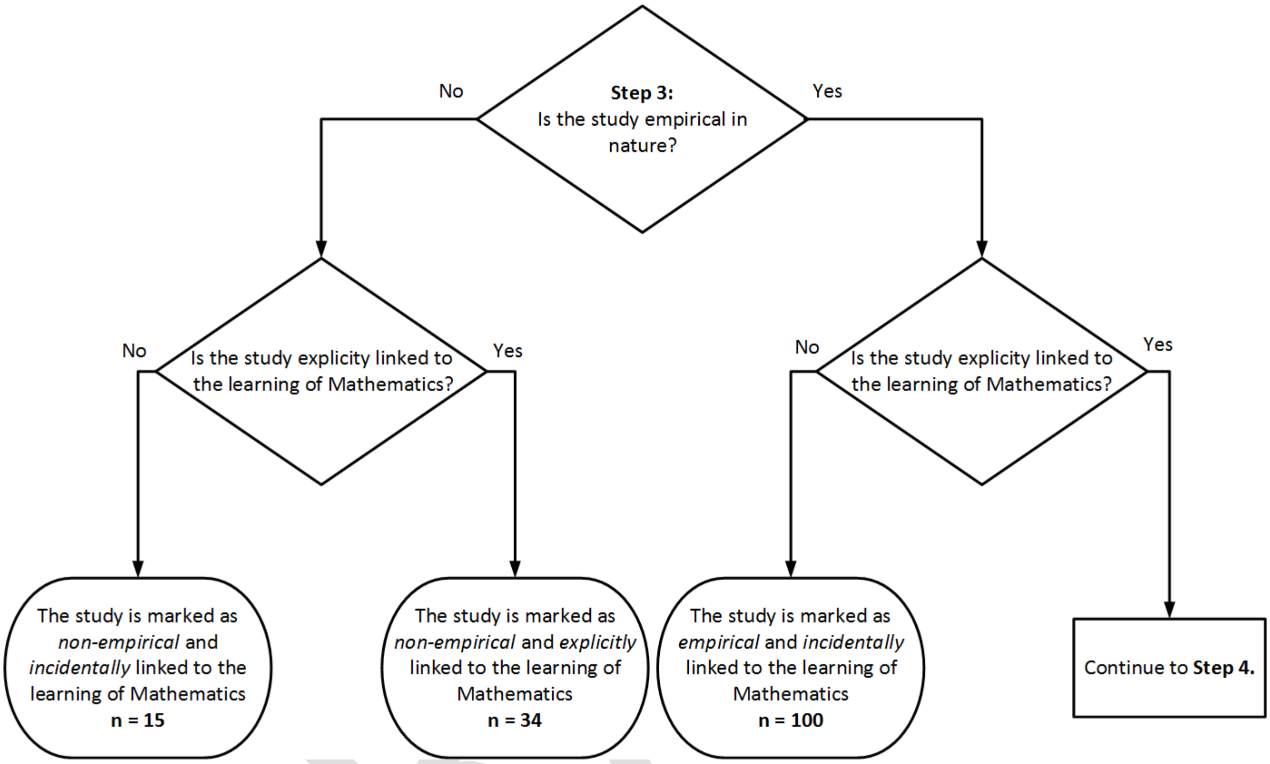

Figure 3: Step 3 of the classification process

Step 4: Activity Approach

Empirical studies that were *explicitly* linked to the learning of mathematics were classified by the type of knowledge that researchers tried to impart when teaching the mathematics concepts: *Procedural*, *Conceptual* or *Both* (where there was evidence of *Procedural* and *Conceptual* knowledge). There were two additional *Activity Approach* classifications used for studies explicitly linked to the learning of mathematics. These were *Not Clear,* for studies where there was an empirical component but insufficient information to determine whether the intervention involved *Procedural* or *Conceptual* knowledge. This step in the classification process is depicted in Figure 4.

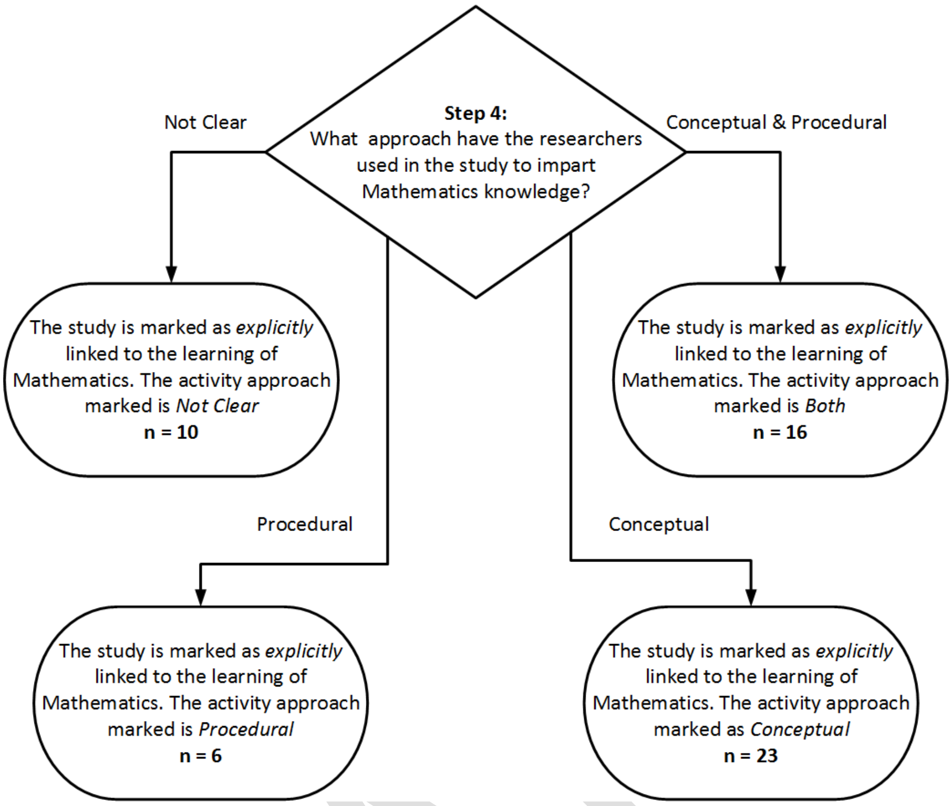

Figure 4: Step 4 in the classification process

It can be difficult to make an exact distinction between *Conceptual* knowledge and *Procedural* knowledge. In this review the *Activity Approach* has been classified according to the distinctions described by Haapasalo and Kadijevich (2000, pg. 141) who defined conceptual knowledge as "knowledge of and a skilful 'drive' along particular networks, the elements of which can be concepts, rules … and even problems … given in various representation forms.", and added that conceptual knowledge "typically requires conscious thinking". Thus, a student demonstrating conceptual knowledge of a mathematical idea will understand the underlying concepts and relationships between these concepts, and will be able to able to explain why a problem involving this idea can be solved, not just how to solve it. An example of study where conceptual knowledge was evident involved an investigation of students' use of ViMAP, a visual programming language, to create agent-based simulations and how this impacted on their conceptual understanding of the relationships between distance, speed and acceleration (Farris and Sengupta 2013). *Procedural* knowledge is defined by Haapasalo and Kadijevich (2000, pg. 141) as "dynamic and successful utilization of particular rules, algorithms or procedures within relevant representation form(s)" and argued that this type of knowledge often "calls for automated and unconscious steps" to be made by students to solve a problem. For example, Xiaoxia and Zhurong (2011) described an activity where high school students calculated the area of a circle by translating the equation into an algorithm written in C program code but did not include evidence on how students conceptualised the relationships between the equation's variables when writing this program.

Some of the studies reviewed also contained evidence of both conceptual and procedural approaches in their activities. Layer et al. (2012) organised a three-week summer camp for high school students that was focused on teaching CS. One of the activities in this summer camp involved teaching students about how mapping and GPS software is implemented. In this activity, students were taught about the "great circle distance formula used to find the shortest distance between two points on the surface of a sphere" Layer et al. (2012, pg. 3), calculated distances by hand using the formula and then implemented the formula in code, an example of a *Procedural* approach. In another activity, the students were taught about different encryption algorithms, devised their own algorithms and paired with other students to test these algorithms, which was an example of a *Conceptual* approach.

Analysis of relevant studies

The 339 studies that were determined to be relevant were classified using the process detailed in the previous section. In this section, our findings from analysis of these results are presented and discussed. The results begin from a broad perspective of the reviewed studies and then focus on the studies that make an explicit link between computational thinking and the learning of mathematics.

Database analysis

The studies relevant to this review originated from six academic databases. The frequency of the different databases of origin for the 339 relevant studies is shown in Figure 5. The three databases that contained most of the reviewed studies were: IEEE Xplore, Springer and ACM Digital Library.

Over half (56%) of the reviewed studies originated from the two databases with a CS focus. Consequently, many of the reviewed studies were conducted by CS academics with an interest in K – 12 Education. These studies often involved introducing K – 12 students and teachers to computational thinking and programming as part of summer camps (Jimenez and Gardner-McCune 2015; Al-Duwis et al. 2013), teacher workshops (Jiangjiang et al. 2015) and as part of K – 12 classes (Nikou and Economides 2014). Additionally, CS academics often ran interventions that aimed to improve students' perceptions of CS and inform them about the potential careers (Al-Duwis et al. 2013; Larkins et al. 2013).

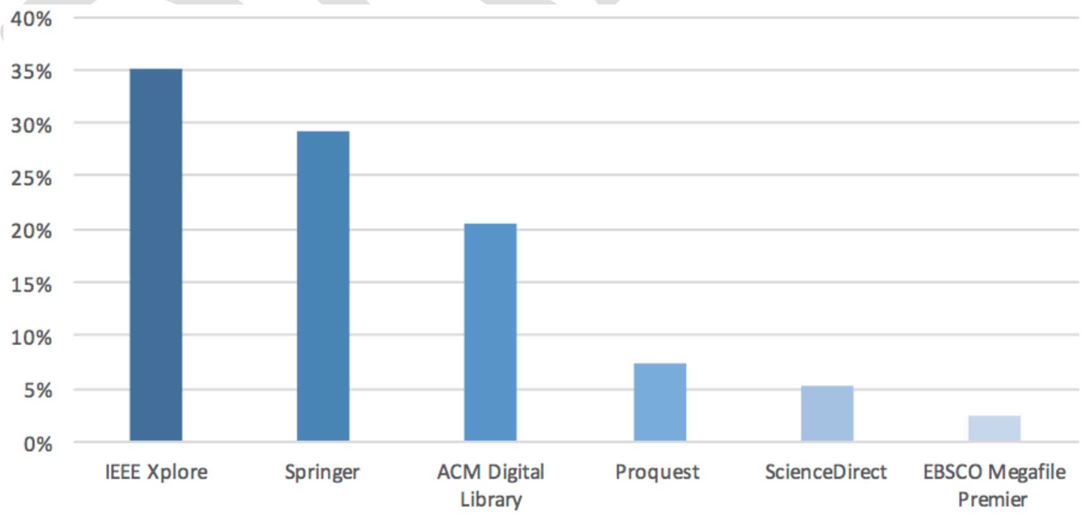

Figure 5: The Databases that relevant results originated from

Studies Explicitly Linking Computational Thinking and Mathematics

The links that the researchers made between computational thinking and the learning of mathematics were classified for the 339 relevant studies, according to whether they were *explicitly, incidentally* or *not at all* linked. The percentages of the studies linked to the learning of mathematics are shown in Figure 6. The majority of the studies (73%) reviewed were not explicitly linked and studies with no link were more common (40%) than those only linked incidentally (33%).

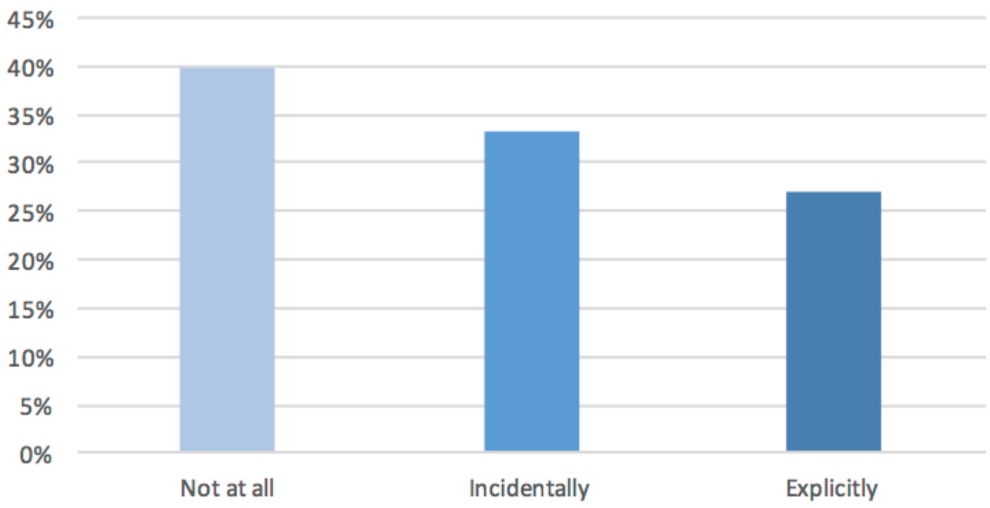

Figure 6: Nature of links in studies between computational thinking and the learning of mathematics

Studies that *incidentally* linked computational thinking to mathematics were more common than those *explicitly* linked. Studies were considered to be *incidentally* linked when there was some evidence of mathematics concepts present in the study's intervention or in the authors' discussion. Fundamental programming concepts like variables and performing number operations, such as addition and subtraction, were often used in these studies introducing students to computational thinking and programming (Werner et al. 2012; Atmatzidou and Demetriadis 2016; McCoid et al. 2013). Generally, the purpose of the application of these concepts, however, was not to impart mathematical content knowledge to participants but to introducing programming concepts. Thus, there were many studies that did incidentally include mathematics concepts, particularly in the domain areas of *Numbers & Operations* and *Algebra*.

Connections to Different Domain Areas of Mathematics

The studies which were found to be *incidentally* or *explicitly* linked to the learning of mathematics present were classified by domain area. The results of this classification are shown in **Error! Reference source not found.**. *Numbers & Operations (*in 87% of *incidentally linked,* in 77% of *explicitly linked)* and *Algebra* (in 80% of *incidentally* linked and 73% of *explicitly* linked), were the most common domain areas present in these studies. Measurement & Functions (in 21% of *incidentally* linked, in 39% of *explicitly* linked) and *Statistics & Probability (*in 14% of *incidentally linked,* in 17% of *explicitly linked)* were rarely present in the studies linked to the learning of mathematics.

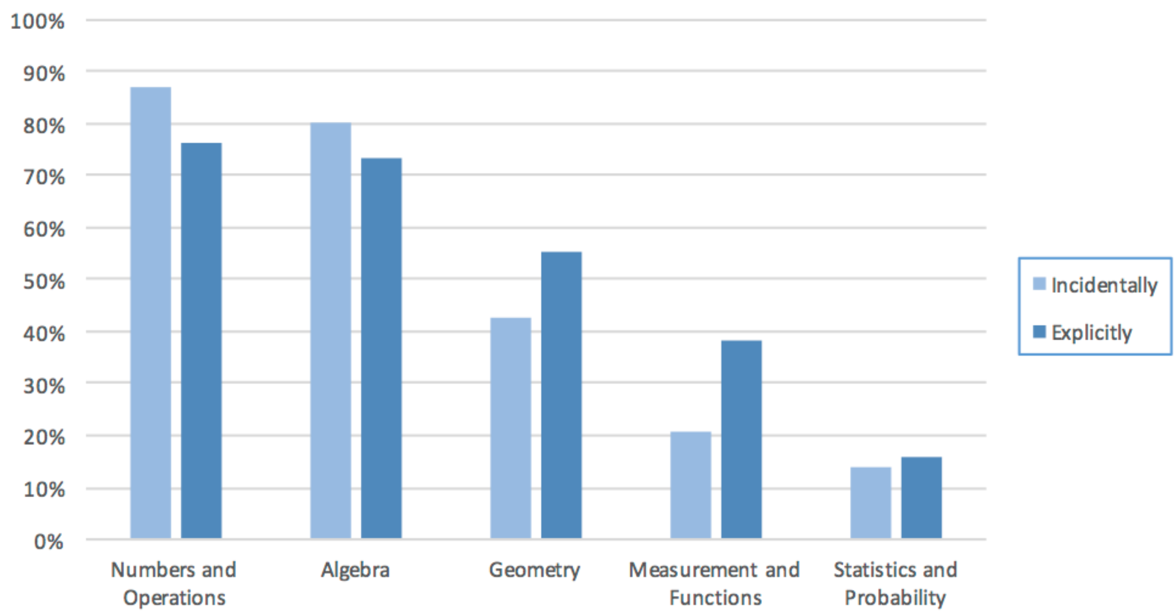

Figure 7: Domain areas present in studies linked to learning of Mathematics

There was often evidence in the studies of multiple mathematics domain areas being present. *Numbers & Operations*, *Algebra* and *Geometry* often appeared together in many of the studies, particularly in the studies where there was an intervention in which students designed games, programmed robots or used turtle geometry environments. Examples of these types of activities in the reviewed studies included designing a maze game in Scratch (Akcaoglu 2014), programming a robot arm to pick up objects (Kurebayashi et al. 2008) and using turtle geometry to draw patterns for physical fabrication (Turbak et al. 2012). These activities involved programming to position and move objects around the Cartesian plane, and thus required the application of variables, operations and geometry.

Measurements & Functions and *Statistics & Probability* were the least common domain areas present in the reviewed studies. This could be because *Numbers & Operations* and *Algebra* were often present because they are part of fundamental programming concepts, and because many of the programming environments used in K – 12 involve *Geometry* concepts. *Measurement* was commonly present in studies where participants used and interpreted data collected from sensors, for example collecting temperature readings from a sensor (Phalke and Lysecky 2010; Brady et al. 2014). *Functions* were often present in studies where students created models, particularly those for Physics concepts, such as simulations of Newton's laws of motion (Dukeman et al. 2013; Aiken et al. 2013). *Probability* was common in studies that involved simulations as well. For example, a student calculated "probability to estimate the direction of fire spreading" in a simulation created in AgentSheets (Koh et al. 2013, pg. 599), and students modified probabilities used in a model and observed the effect on the simulation (A. Repenning et al. 2013). *Statistics* were commonly present in studies conducted in the context of science education and usually focused on the use of computational thinking when students interpreted data collected from experiments (Kim 2015; Borne 2010).

Ten of the studies had evidence of all five mathematics domain areas present in them. Six of these studies were explicitly linked to mathematics (7% of all explicitly linked studies) and 4 of these studies were incidentally linked to mathematics (3% of all incidentally linked studies). None of these studies consisted of a single activity that involved all five mathematics domain areas. These studies either involved a combination of activities, which covered different domain areas, that were part of a course or workshop content (Jiangjiang et al. 2015; Chatzinikolakis and Papadakis 2014; Bojic and Arratia 2015), or the studies discussed the integration of computational and programming into mathematics curricula (Sengupta et al. 2013; Reeping and Reid 2015; Dagienė 2008).

Activity Approaches in Empirical Studies

Studies that were *explicitly* linked to mathematics in Step 3 were subsequently classified by the approach used in the activities conducted each study, the *Activity Approach*. The results of this classification are illustrated in Figure 8. More than a third of the studies were non-empirical (38% of the explicitly linked studies). The authors of these studies would often discuss the relationship between mathematics and computational thinking, as well as high-level mappings of mathematics concepts to activities, but they did not collect data to support their arguments.

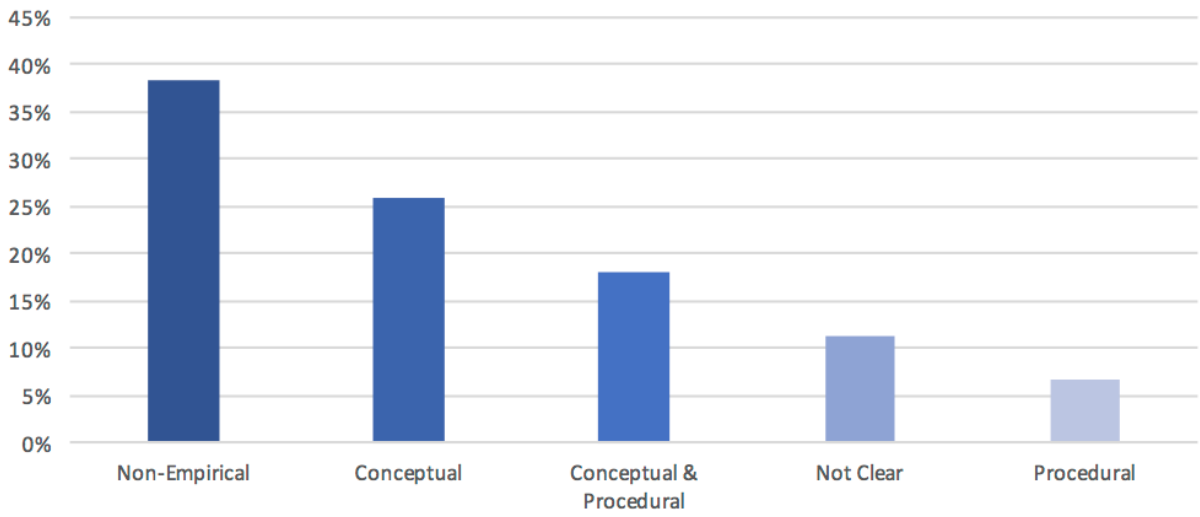

Figure 8: Activity approaches of studies explicitly linked to Mathematics

Studies that involved an intervention with a focus on imparting conceptual knowledge were the second most common (26%). The central vision of Papert's work with Logo was to give students another way of reasoning about and conceptualising mathematics (Papert 1996). The legacy of Papert's vision was evident in many of the reviewed studies that were *explicitly* linked, particularly those where the authors incorporated activities that aimed to increase students' conceptual understanding of *Geometry* through computational thinking and programming (see, for example Kyriakides et al. 2015).

Studies that were classified with a *Not Clear Activity Approach* had an empirical component but did not provide enough information about the intervention or activities to determine the *Activity Approach*. Often these were studies that only briefly outlined the activities presented to teachers and/or students at a workshop and reported on perceptions of CS (Sullivan et al. 2015; Ahamed et al. 2010)

Empirical Studies with Evidence of Impact on Students' Outcomes

The empirical studies that were *explicitly* linked to the learning of mathematics were reviewed to identify which had included evidence of participants' learning outcomes in mathematics (n = 54). Only studies where researchers had used quantitative methods were considered for this last part of the analysis. It is important to emphasise here that we consider non-quantitative studies essential to the development of the field, but we found that they all dealt with small samples and therefore would not be considered representative by government agencies in terms of broad educational reform. The research methods used in each of these studies are shown in **Error! Reference source not found.**.

Table 2: Research methods used in empirical studies explicitly linked to the learnign of mathematics

Research Methods	Number of Studies
Quantitative Only	20
Qualitative Only	18
Mixed Methods	15

The 35 studies that contained quantitative analysis were examined to identify the type of evidence that was gathered, the results of which are shown in Figure 9. The most common type of evidence gathered in the studies was students' perceptions of CS and related careers. Six of the studies were focused on an evaluation of a tool, course or workshop. For example, Ruutmann (2014) surveyed Estonian secondary school STEM in order to evaluate teacher education courses that incorporated mathematics, programming and computational thinking. Self-reported learning outcomes and attitudes were also reported in three of the studies, for example Ke (2014) found positive changes in students' dispositions towards mathematics as a result of creating games.

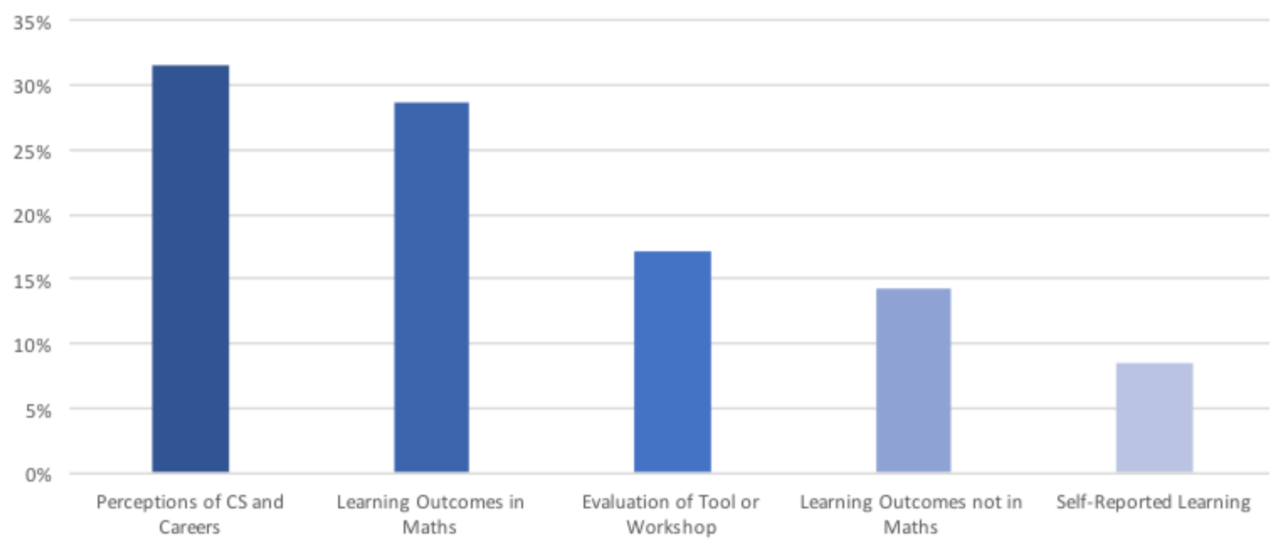

Figure 9: Type of evidence gathered in quantitative studies

There were only ten studies that reported participants' learning outcomes in mathematics. None of these studies involved the observation of long-term learning outcomes or conducted an experiment with randomisation of participants. Four of these studies contained only descriptive statistics, which were used to report on students' results in assessments. Six of these 10 studies had both descriptive and inferential statistics, two of which were focused on investigating the correlation between students' scores on computational thinking assessments and mathematics test scores (Oliveira et al. 2014; Lewis and Shah 2012).

On these ten studies, four involved an intervention, contained both descriptive and inferential statistics, and reported on a change in learning outcomes in mathematics. One of these studies reported on a change in one group of student's understandings of fractal geometry by comparing results from pre-test and post-test questionnaires (Wilkerson-Jerde 2014). Another study compared students' performance on pre-test and post-test programming quizzes by gender (Kalelioğlu 2015).

Only two of these four studies had a research design with control and experimental groups: Boyce et al. (2011) and Calao et al. (2015). Boyce et al. (2011) compared the impact of a game and online tool on students' understanding of co-ordinate geometry. The game, titled BeadLoom Game (BLG), was a gamified version of the online tool, titled Virtual BeadLoom (VBL). In BLG and VBL students could use iteration to create virtual bead art that were drawn on a Cartesian plane. The study reported results from two experiments conducted at summer camps held in the USA. The first of these experiments had 21 middle school students as participants and the second had 22 high school students. The experiments were conducted using a "switching replications experimental design" (Boyce et al. 2011, pg. 245): the participants were split into two groups and used received both BLG and VBL interventions. The students were given a pre-test and post-test, as well as a test in the middle of the experiment, before the groups switched between using BLG and VBL. Boyce et al. (2011) were mainly concerned with the comparison of the effectiveness of the game and tool, and concluded that adding gaming elements to the tool resulted in a significant increase in students' motivation and enjoyment, as well as their understanding of co-ordinate geometry. The focus of the study was largely on students' motivation and enjoyment, rather than the effect of incorporating computational thinking in the teaching of mathematics. The results from this study do indicate, however, that mathematics concepts can be combined with computing concepts (iteration) and have a positive effect on students' learning outcomes and motivation.

The second quasi-experimental study, Calao et al. (2015), examined the effect of learning to code in Scratch on students' understanding of mathematical processes. This study was conducted with a quasi-experimental design, and used a pre-test and post-test to measure students' learning in four different mathematical processes: "*Modelling*", "*Reasoning*", "*Problem solving*" and "*Exercising*" (p. 20). There were 42 participants in the study, who were all 6th Grade (11 – 12 years old) Colombian students, that were divided into a control group (n = 18) and an experimental group (n = 24). The experimental group were given an intervention for 3 months that provided students an introduction to programming concepts in Scratch and the opportunity for them to develop their own games and simulations. There was no change in the approaches used to teach mathematics to the control group, which the authors refer to as the "traditional" approach (p. 24). Calao et al. (2015) found that the post-test scores of students in the experimental group were significantly higher than those in the control group, indicating that learning to code in Scratch was more effective for teaching mathematical processes than the 'traditional' approach. There are two potential caveats in this study, however. Firstly, the distribution of the difference scores (post-test – pre-test) are not reported or compared, so it's not clear if the high mean score in the experimental group was skewed by high performing students. Secondly, the content of the post-test assessment and its relation to the material delivered to the experimental group are not explained in the study. One explanation for the high scores in the experimental group could be that the post-test had questions directly related to the skills students learned in the intervention, but Calao et al. (2015) do not elaborate this in their discussion. Despite these caveats the results of the study do provide some evidence that students learning computational thinking through coding in Scratch can have positive effects on their ability to use mathematical processes.

These findings indicate that there is a lack of quasi-experimental research that *explicitly* links the learning of mathematics (as evidenced by student academic outcomes) with computational thinking. We believe that this is an area that needs to be investigated to build an evidence base for combining mathematics and computational thinking in practice.

Conclusions

The introduction of computational thinking skills into K – 12 curricula presents many challenges and opportunities for educators committed to improving students' understanding of mathematics. The analysis of existing literature presented in this paper signals areas of research that could potentially be broadened in order to provide teachers, outreach providers and professional development agencies an evidence base for developing digital learning experiences in mathematics that create for children a "range of opportunities to engage as a bricoleur or bricoleuse in activities with scientific and mathematical content" (Papert 1993, p. 145).

Many of the studies were found in specialised computer science databases and conducted by researchers in the computer science field as part of outreach or summer camps that involved teaching students computational thinking, usually through the introduction of programming languages. These studies were often centred on improving students' perceptions of CS and related careers or evaluating feedback from participants, rather than the effect of the intervention on students' learning. There appears to be a dearth of research originating from academics from an educational research background. To us, this suggests that there is an opportunity for more multidisciplinary research to be conducted, which is informed by both computer science and education literature.

Studies that *explicitly* linked the learning of mathematics concepts with computational thinking were uncommon in the reviewed literature. There was often evidence of concepts involving numbers, operations or algebra being imparted in the studies reviewed, but this was usually with the intention of introducing programming concepts. This seems to suggest, and is confirmed by prior work conducted by the likes of Papert, Noss, Hoyles, Harel, Wilensky, Resnick, etc., that there are opportunities to investigate explicit ways with which to enhance the understanding of mathematics concepts using the computational thinking. In particular, more research needs to be conducted into ways of using computational thinking for the teaching concepts in the domain areas of probability, statistics, measurement and functions.

Non-empirical studies that *explicitly* linked the learning of mathematics to computational thinking were common in the reviewed studies. These studies often contained discussion of potential mappings between these two concepts but most did not include concrete ideas or practices that could be applied by K – 12 educators in domain areas other than geometry. This is another literature void that could be addressed by future research in the field.

Empirical studies that explicitly link the learning of mathematics to computational thinking and that reported on students' learning outcomes in mathematics were rare in the reviewed literature. Those studies that did report on students' learning outcomes in mathematics were often short-term, one-off studies with no long-term learning outcomes. The authors of this scoping review are aware that an independently evaluated quasi-empirical study, ScratchMaths, is currently underway in the UK being the first of its kind (Benton et al. 2017). Studies explicitly targeting the learning of mathematics such as ScratchMaths, with a focus on domain areas less explored to date, and conducted by multidisciplinary teams, need to occur in other parts of the world if we are to provide empirical evidence of the transfer effects of programming to the learning of mathematics.

References

Ackermann, E. (2001). Piaget's constructivism, Papert's constructionism: What's the difference. *Future of learning group publication, 5*(3), 438.

Agalianos, A., Whitty, G., & Noss, R. (2006). The social shaping of Logo. *Social studies of science, 36*(2), 241-267.

Ahamed, S. I., Brylow, D., Ge, R., Madiraju, P., Merrill, S. J., Struble, C. A., et al. (2010). *Computational thinking for the sciences: a three day workshop for high school science teachers.* Paper presented at the Proceedings of the 41st ACM technical symposium on Computer science education, Milwaukee, Wisconsin, USA,

Aiken, J. M., Caballero, M. D., Douglas, S. S., Burk, J. B., Scanlon, E. M., Thoms, B. D., et al. (2013). Understanding student computational thinking with computational modeling. *AIP Conference Proceedings, 1513*(1), 46-49, doi:10.1063/1.4789648.

Akcaoglu, M. (2014). Learning problem-solving through making games at the game design and learning summer program. *Educational Technology, Research and Development, 62*(5), 583-600, doi:10.1145/2159365.2159366 .

Al-Duwis, M., Al-Khalifa, H. S., Al-Razgan, M. S., Al-Rajebah, N., & Al-Subaihin, A. (2013). *Increasing high school girls awareness of computer science through summer camp.* Paper presented at the Global Engineering Education Conference (EDUCON), 2013 IEEE, 13-15 March 2013

Arksey, H., & O'Malley, L. (2005). Scoping studies: towards a methodological framework. *International journal of social research methodology, 8*(1), 19-32.

Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems, 75, Part B*, 661-670, doi:http://dx.doi.org/10.1016/j.robot.2015.10.008.

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., Newton, H., Andreae, P., & Robins, A. (2012a). The introduction of Computer Science to NZ High Schools — an analysis of student work. 5-15.

Bell, T., Rosamond, F., & Casey, N. (2012b). Computer Science Unplugged and Related Projects in Math and Computer Science Popularization. In H. Bodlaender, R. Downey, F. Fomin, & D.

Marx (Eds.), *The Multivariate Algorithmic Revolution and Beyond* (Vol. 7370, pp. 398-456, Lecture Notes in Computer Science): Springer Berlin Heidelberg.

Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2016). *Building mathematical knowledge with programming: insights from the ScratchMaths project.* Paper presented at the Constructionism 2016, Bangkok, Thailand, February 1 - 6 2016

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*.

Bojic, I., & Arratia, J. F. (2015). *Teaching K-12 students STEM-C related topics through playing and conducting research.* Paper presented at the Frontiers in Education Conference (FIE), 2015. 32614 2015. IEEE, 21-24 Oct. 2015

Borne, K. D. (2010). Astroinformatics: data-oriented astronomy research and education. *Earth Science Informatics, 3*(1), 5-17, doi:10.1007/s12145-010-0055-2.

Boyce, A. K., Campbell, A., Pickford, S., Culler, D., & Barnes, T. (2011). *Experimental evaluation of BeadLoom game: how adding game elements to an educational tool improves motivation and learning.* Paper presented at the Proceedings of the 16th annual joint conference on Innovation and technology in computer science education, Darmstadt, Germany,

Brady, C., Holbert, N., Soylu, F., Novak, M., & Wilensky, U. (2014). Sandboxes for Model-Based Inquiry. *Journal of Science Education and Technology, 24*(2), 265-286, doi:10.1007/s10956-014-9506-8.

Brennan, K., & Resnick, M. New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada, 2012* (pp. 1-25)

Brown, N. C. C., Sentance, S., Crick, T., & Humphreys, S. (2014). Restart: The Resurgence of Computer Science in UK Schools. *Trans. Comput. Educ., 14*(2), 1-22, doi:10.1145/2602484.

Calao, L., Moreno-León, J., Correa, H., & Robles, G. (2015). Developing Mathematical Thinking with Scratch. In G. Conole, T. Klobučar, C. Rensing, J. Konert, & É. Lavoué (Eds.), *Design for Teaching and Learning in a Networked World* (Vol. 9307, pp. 17-27, Lecture Notes in Computer Science): Springer International Publishing.

Chatzinikolakis, G., & Papadakis, S. (2014). *Motivating K-12 students learning fundamental Computer Science concepts with App Inventor.* Paper presented at the Interactive Mobile Communication Technologies and Learning (IMCL), 2014 International Conference on, 13-14 Nov. 2014

Dagienė, V. (2008). Teaching Information Technology and Elements of Informatics in Lower Secondary Schools: Curricula, Didactic Provision and Implementation. In R. Mittermeir, & M. Sysło (Eds.), *Informatics Education - Supporting Computational Thinking* (Vol. 5090, pp. 293-304, Lecture Notes in Computer Science): Springer Berlin Heidelberg.

Dukeman, A., Caglar, F., Shekhar, S., Kinnebrew, J., Biswas, G., Fisher, D., et al. (2013). Teaching Computational Thinking Skills in C3STEM with Traffic Simulation. In A. Holzinger, & G. Pasi (Eds.), *Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data* (Vol. 7947, pp. 350-357, Lecture Notes in Computer Science): Springer Berlin Heidelberg.

Falkner, K., Vivian, R., & Falkner, N. (2014). *The Australian Digital Technologies Curriculum: Challenge and Opportunity.* Paper presented at the Australasian Computing Education, Auckland, New Zealand,

Farris, A. V., & Sengupta, P. (2013). *On the aesthetics of children's computational modeling for learning science.* Paper presented at the Proceedings of the 12th International Conference on Interaction Design and Children, New York, New York, USA,

Feurzeig, W., Papert, S., Bloom, M., Grant, R., & Solomon, C. (1969). *Programming-Languages as a Conceptual Framework for Teaching Mathematics. Final Report on the First Fifteen Months of the LOGO Project* (Vol. Accessed from http://nla.gov.au/nla.cat-vn5207614). [Washington, D.C.]: Distributed by ERIC Clearinghouse.

Fisher, L. M. (2016). A decade of ACM efforts contribute to computer science for all. *Communications of the ACM, 59*(4), 25-27, doi:10.1145/2892740.

Haapasalo, L., & Kadijevich, D. (2000). Two Types of Mathematical Knowledge and Their Relation. *Journal für Mathematik-Didaktik, 21*(2), 139-157, doi:10.1007/bf03338914.

Hoyles, C., & Noss, R. (1992). *Learning mathematics and logo*: MIT Press.

Jiangjiang, L., Wilson, J., Hemmenway, D., Yingbo, X., & Cheng-Hsien, L. (2015). *Oh SNAP! A one-week summer computing workshop for K-12 teachers.* Paper presented at the Computer Science & Education (ICCSE), 2015 10th International Conference on, 22-24 July 2015

Jimenez, Y., & Gardner-McCune, C. (2015). *Using App inventor & history as a gateway to engage African American students in computer science.* Paper presented at the Research in Equity and Sustained Participation in Engineering, Computing, and Technology (RESPECT), 2015, 13-14 Aug. 2015

Kafai, Y. B., & Burke, Q. (2015). Constructionist Gaming: Understanding the Benefits of Making Games for Learning. *Educational Psychologist, 50*(4), 313-334, doi:10.1080/00461520.2015.1124022.

Kafai, Y. B., & Harel, I. (1991). Children's learning through consulting: When mathematical ideas, programming knowledge, instructional design, and playful discourse are intertwined. In I. Harel, & S. Papert (Eds.), *Constructionism* (pp. 85-100). Norwood, NJ: Ablex.

Kalelioğlu, F. (2015). A new way of teaching programming skills to K-12 students: Code.org. *Computers in Human Behavior, 52*, 200-210, doi:http://dx.doi.org/10.1016/j.chb.2015.05.047.

Kalelioglu, F., Gülbahar, Y., & Kukul, V. (2016). A Framework for Computational Thinking Based on a Systematic Research Review. *Baltic Journal of Modern Computing, 4*(3), 583.

Ke, F. (2014). An implementation of design-based learning through creating educational computer games: A case study on mathematics learning during design and computing. *Computers & Education, 73*, 26-39, doi:http://dx.doi.org/10.1016/j.compedu.2013.12.010.

Kim, H. (2015). Inquiry-Based Science and Technology Enrichment Program for Middle School-Aged Female Students. *Journal of Science Education and Technology*, 1-13, doi:10.1007/s10956-015-9584-2.

Koh, K. H., Repenning, A., Nickerson, H., Endo, Y., & Motter, P. (2013). *Will it stick?: exploring the sustainability of computational thinking education through game design.* Paper presented at the Proceeding of the 44th ACM technical symposium on Computer science education, Denver, Colorado, USA,

Kurebayashi, S., Aoki, H., Kamada, T., Kanemune, S., & Kuno, Y. (2008). Proposal for Teaching Manufacturing and Control Programming Using Autonomous Mobile Robots with an Arm. In R. Mittermeir, & M. Sysło (Eds.), *Informatics Education - Supporting Computational Thinking* (Vol. 5090, pp. 75-86, Lecture Notes in Computer Science): Springer Berlin Heidelberg.

Kyriakides, A. O., Meletiou-Mavrotheris, M., & Prodromou, T. (2015). Mobile technologies in the service of students' learning of mathematics: the example of game application A.L.E.X. in the context of a primary school in Cyprus. *Mathematics Education Research Journal*, 1-26, doi:10.1007/s13394-015-0163-x.

Larkins, D. B., Moore, J. C., Rubbo, L. J., & Covington, L. R. (2013). *Application of the cognitive apprenticeship framework to a middle school robotics camp.* Paper presented at the Proceeding of the 44th ACM technical symposium on Computer science education, Denver, Colorado, USA,

Layer, R., Sherriff, M., & Tychonievich, L. (2012). *"Inform, Experience, Implement" - Teaching an intensive high school summer course.* Paper presented at the Frontiers in Education Conference (FIE), 2012, 3-6 Oct. 2012

Lewis, C. M., & Shah, N. (2012). *Building upon and enriching grade four mathematics standards with programming curriculum.* Paper presented at the Proceedings of the 43rd ACM technical symposium on Computer Science Education, Raleigh, North Carolina, USA,

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, doi:http://dx.doi.org/10.1016/j.chb.2014.09.012.

McCoid, S., Freeman, J., Magerko, B., Michaud, C., Jenkins, T., McKlin, T., et al. (2013). EarSketch: An integrated approach to teaching introductory computer music. *Organised Sound, 18*(2), 146-160, doi:http://dx.doi.org/10.1017/S135577181300006X.

Mensing, K., Mak, J., Bird, M., & Billings, J. (2013). *Computational, model thinking and computer coding for U.S. Common Core Standards with 6 to 12 year old students.* Paper presented at the Emerging eLearning Technologies and Applications (ICETA), 2013 IEEE 11th International Conference on, 24-25 Oct. 2013

Nikou, S. A., & Economides, A. A. (2014). *Transition in student motivation during a scratch and an app inventor course.* Paper presented at the Global Engineering Education Conference (EDUCON), 2014 IEEE, 3-5 April 2014

Oliveira, O. L., Nicoletti, M. C., & Cura, L. M. d. V. (2014). *Quantitative correlation between ability to compute and student performance in a primary school.* Paper presented at the Proceedings of the 45th ACM technical symposium on Computer science education, Atlanta, Georgia, USA,

Papert, S. (1972). Teaching children to be mathematicians versus teaching about mathematics. *International journal of mathematical education in science and technology, 3*(3), 249-262.

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

Papert, S. (1990). *A critique of technocentrism in thinking about the school of the future*: Epistemology and Learning Group, MIT Media Laboratory.

Papert, S. (1993). *The children's machine: Rethinking school in the age of the computer*. New York: Basic Books.

Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning, 1*(1), 95-123.

Phalke, A., & Lysecky, S. (2010). Adapting the eBlock Platform for Middle School STEM Projects: Initial Platform Usability Testing. *Learning Technologies, IEEE Transactions on, 3*(2), 152-164, doi:10.1109/TLT.2009.41.

Pinto, A., & Escudeiro, P. (2014). *The use of Scratch for the development of 21st century learning skills in ICT.* Paper presented at the Information Systems and Technologies (CISTI), 2014 9th Iberian Conference on, 18-21 June 2014

Reeping, D., & Reid, K. (2015). *Viewing K-12 mathematics and science standards through the lens of the first-year introduction to engineering course classification scheme.* Paper presented at the Frontiers in Education Conference (FIE), 2015. 32614 2015. IEEE, 21-24 Oct. 2015

Repenning, A., Basawapatna, A., & Klymkowsky, M. (2013). *Making educational games that work in the classroom: A new approach for integrating STEM simulations.* Paper presented at the Games Innovation Conference (IGIC), 2013 IEEE International, 23-25 Sept. 2013

Repenning, A., Webb, D., & Ioannidou, A. (2010). *Scalable game design and the development of a checklist for getting computational thinking into public schools.* Paper presented at the Proceedings of the 41st ACM technical symposium on Computer science education, Milwaukee, Wisconsin, USA,

Ruutmann, T. (2014). *Optional STEM courses for secondary schools designed and implemented for enhancement of K-12 technology education in order to excite students' interest in technology and engineering education.* Paper presented at the Interactive Collaborative Learning (ICL), 2014 International Conference on, 3-6 Dec. 2014

Scherer, R. (2016). Learning from the Past–The Need for Empirical Evidence on the Transfer Effects of Computer Programming Skills. *Frontiers in Psychology, 7*, 1390, doi:10.3389/fpsyg.2016.01390.

Sengupta, P., Kinnebrew, J., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. [Article]. *Education & Information Technologies, 18*(2), 351-380, doi:10.1007/s10639-012-9240-x.

Sullivan, K., Byrne, J. R., Bresnihan, N., O'Sullivan, K., & Tangney, B. (2015). *CodePlus — Designing an after school computing programme for girls.* Paper presented at the Frontiers in Education Conference (FIE), 2015. 32614 2015. IEEE, 21-24 Oct. 2015

Tisue, S., & Wilensky, U. Netlogo: A simple environment for modeling complexity. In *International conference on complex systems, 2004* (Vol. 21, pp. 16-21): Boston, MA

Turbak, F., Sandu, S., Kotsopoulos, O., Erdman, E., Davis, E., & Chadha, K. (2012). *Blocks languages for creating tangible artifacts.* Paper presented at the Visual Languages and Human-Centric Computing (VL/HCC), 2012 IEEE Symposium on, Sept. 30 2012-Oct. 4 2012

Vivian, R., Falkner, K., & Falkner, N. (2014). Addressing the challenges of a new digital technologies curriculum: MOOCs as a scalable solution for teacher professional development. *2014, 22*, doi:10.3402/rlt.v22.24691.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., et al. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology, 25*(1), 127-147, doi:10.1007/s10956-015-9581-5.

Werner, L., Denner, J., Campe, S., & Kawamoto, D. C. (2012). *The fairy performance assessment: measuring computational thinking in middle school.* Paper presented at the Proceedings of the 43rd ACM technical symposium on Computer Science Education, Raleigh, North Carolina, USA,

Wilensky, U., & Resnick, M. (1999). Thinking in levels: A dynamic systems approach to making sense of the world. *Journal of Science Education and Technology, 8*(1), 3-19.

Wilkerson-Jerde, M. (2014). Construction, categorization, and consensus: student generated computational artifacts as a context for disciplinary reflection. *Educational Technology Research & Development, 62*(1), 99-121, doi:10.1007/s11423-013-9327-0.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM, 49*(3), 33-35, doi:10.1145/1118178.1118215.

Wing, J. M. (2016). Computational thinking, 10 years later

Xiaoxia, W., & Zhurong, Z. (2011). *The research of situational teaching mode of programming in high school with Scratch.* Paper presented at the Information Technology and Artificial Intelligence Conference (ITAIC), 2011 6th IEEE Joint International, 20-22 Aug. 2011