Conceptions and Misconceptions about Computational Thinking among Italian Primary School Teachers

Isabella Corradini
Themis Research Centre
Rome, Italy
isabellacorradini@themiscrime.com

Michael Lodi
University of Bologna
Dep. of Comp. Science and Eng.
Bologna, Italy
michael.lodi2@unibo.it

Enrico Nardelli
University of Roma “Tor Vergata"
Department of Mathematics
Rome, Italy
nardelli@mat.uniroma2.it

ABSTRACT

Many advanced countries are recognizing more and more the importance of teaching computing, in some cases even as early as in primary school. “Computational thinking" is the term often used to denote the conceptual core of computer science or “the way a computer scientist thinks", as Wing put it. Such term - given also the lack of a widely accepted definition - has become a “buzzword" meaning different things to different people. We investigated the Italian primary school teachers' conceptions about computational thinking by analyzing the results of a survey (N=972) conducted in the context of “Programma il Futuro" project. Teachers have been asked to provide a definition of computational thinking and to answer three additional related closed-ended questions. The analysis shows that, while almost half of teachers (43.4%) have included in their definitions some fundamental elements of computational thinking, very few (10.8%) have been able to provide an acceptably complete definition. On a more positive note, the majority is aware that computational thinking is not characterized by coding or by the use of information technology.

KEYWORDS

Computational thinking definition; Informatics education; Conceptions and misconceptions; Primary school teachers

ACM Reference format:
Isabella Corradini, Michael Lodi, and Enrico Nardelli. 2017. Conceptions and Misconceptions about Computational Thinking among Italian Primary School Teachers. In *Proceedings of ICER '17, Tacoma, WA, USA, August 18-20, 2017,* 9 pages.
https://doi.org/10.1145/3105726.3106194

1 INTRODUCTION

1.1 Context

The wide popularity gained by the expression “computational thinking" (CT, from now on) after the Wing's paper [20] risks to spoil the original aim. More and more people are now considering CT a new subject, somehow different or distinct from computer science (“computing” in UK, “informatics” in Europe).

We are convinced this approach is wrong and misleading: in the long run it will do more harm than benefit to informatics. After all, in schools they do not teach “linguistic thinking” or “mathematical thinking”, with specific “body of knowledge” or “assessment methods”. Others, e.g. [7], [2], and [9], share our concerns for the dangers of such an approach.

On the other side, the concept of computational thinking, interpreted as “being able to think like a computer scientist and being able to apply this competence to every field of human endeavor” is sorely needed. In fact, it supports the goal of teaching scientific and cultural aspects of computing in schools, focusing on principles and methods more than on systems and tools. This is required since informatics is the science underlying the digital technology pervading all aspects of contemporary society.

Teachers' conceptions regarding a subject are essential for a proper teaching of the subject itself. We therefore investigated the Italian primary school teachers' conceptions about computational thinking and information technology (IT) and how they relate to computer science concepts and principles.

We conducted our investigation in the context of “Programma il Futuro” project[1] [15], whose goal is to increase awareness of informatics as the scientific basis of digital technologies among teachers in Italian primary and secondary schools. The project has adapted Code.org learning material and has introduced it to Italian schools with the support of a dedicated web site, featuring also an introduction to CT. Response has been enthusiastic in terms of participation: in the first two school-years (2014-15 and 2015-16) more than one million students have been engaged and have completed a total of 10 million hours of informatics in schools [6].

1.2 Purpose of the study

Our research has investigated the knowledge level of CT among Italian primary school teachers. More specifically, we addressed the following research questions:

RQ1 which level of understanding do they have with respect to the concept of computational thinking?
RQ2 how do they perceive the relation between technology and computational thinking?
RQ3 how much do they feel prepared to teach computational thinking?

[1]https://programmailfuturo.it

In this paper we use the term *misconception*. In general, the term indicates an incorrect view based on faulty thinking or understanding[2]. In Computer Science Education research literature, the term is often used in the specific context of learning to program, and refers to an inadequate understanding of fundamental programming concepts (for a review see [19]). In this paper we are not referring to such difficulties, but rather to incorrect ideas about what CT is; so we are using the term in its general sense (like, e.g., in [8]).

2 LITERATURE OVERVIEW

2.1 Definition of CT

2.1.1 Five (of many) definitions. The term "computational thinking" was firstly used by Seymour Papert in his book *Mindstorms* [16] and in a work on Mathematics education [17]. After this expression was revived in 2006 by Jeannette Wing [20], many definitions emerged, mainly to support the introduction of CT in K-12 education. In this paper we discuss five important definitions:

- the "Cuny Snyder Wing" definition of CT [22] (that builds on the informal definition in [20] and the philosophical discussion in [21]);
- the 2011 Operational Definition from International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA) [14];
- the definition proposed by Google in its collection of CT resources [11];
- the definition by Brennan and Resnick [4] about CT in Scratch;
- the definition from UK project Barefoot CAS [5].

Wing informally defines CT as "thinking like computer scientists" [20] and then more formally as "*the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent*" [22][3]. In her papers she also identifies characteristic elements of CT. In particular she states the most important elements are *abstraction* (the "mental" tool of computing) and *automation* (using computer, the "metal" tool of computer scientists) - she states that "computing is the automation of our abstractions" [21]. In [22] she recognizes important overlapping or inclusions between CT and other types of thinking: logical thinking, algorithmic thinking, parallel thinking, compositional reasoning, pattern matching, procedural thinking, and recursive thinking.

ISTE and CSTA propose an operational definition, targeting specifically K-12 educators. They define CT as "*a problem-solving process that includes (but is not limited to) the following characteristics: Formulating problems in a way that enables us to use a computer and other tools to help solve them; Logically organizing and analyzing data; Representing data through abstractions such as models and simulations; Automating solutions through algorithmic thinking (a series of ordered steps); Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources; Generalizing and transferring this problem-solving process to a wide variety of problems.*" Moreover they state that CT is "*supported and enhanced by a number of dispositions or attitudes*" that includes "*Confidence in dealing with complexity; Persistence in working with difficult problems; Tolerance for ambiguity; The ability to deal with open ended problems; The ability to communicate and work with others to achieve a common goal or solution*" [14]. Finally they propose a CT vocabulary [13], listing a set of CT terms with a brief definition/explanation: *Data Collection; Data Analysis; Data Representation; Problem Decomposition; Abstraction; Algorithms and Procedures; Automation; Simulation; Parallelization.*

Google assumes the same ISTE/CSTA definition but - instead of a vocabulary - lists and (re)defines a series of CT concepts [11], pointing out that they are *mental processes* or *tangible outcomes*: *Abstraction; Algorithm Design; Automation; Data Analysis; Data Collection; Data Representation; Decomposition; Parallelization; Pattern Generalization; Pattern Recognition; Simulation.*

Brennan and Resnick [4] present a *computational thinking framework*, to describe learning and development that take place when designing and programming interactive media with Scratch platform. They state CT involves three dimensions: *computational concepts* designers employ as they program: sequences, loops, parallelism, events, conditionals, operators, and data; *computational practices* designers develop as they program: being incremental and iterative, testing and debugging, reusing and remixing, and abstracting and modularizing; *computational perspectives* designers form about the world around them and about themselves: expressing, connecting, and questioning.

CAS [5] assumes a Wing-like definition: CT is "*learning to think in ways which allow us, as humans, to solve problems more effectively and, when appropriate, use computers to help us do so*" and then states it involves six concepts (*Logic; Algorithms; Decomposition; Patterns; Abstraction; Evaluation*) and five approaches (*Tinkering; Creating; Debugging; Persevering; Collaborating*).

2.1.2 Common aspects. We compared CT elements found in the analyzed definitions, in analogy with what was done in [9].

Those who give a precise definition agree on the fact that CT is a **way of thinking** (thought process) for **problem solving**. They all somehow specify that it is not just problem solving: the formulation and the solution of the problem must be expressed in a way that allows a **processing agent** (a human or a machine) to carry it out.

Apart from the general statement, all definitions list some constitutive elements of CT. These elements are of very different kinds (from thinking habits to specific programming concepts) and many authors group them in categories, but there is no common agreement on the classification.

We classified all the elements in four categories. For each category we list the elements, trying to summarize all aspects stated in the analyzed definitions. Note that many elements are shared between informatics and other scientific disciplines, but computing features a unique combination of them.

- **Mental processes**: mental strategies useful to solve problems.
  - *Algorithmic thinking*: use algorithmic thinking [14, 21, 22] to design a sequence of ordered step (instructions) to solve a problem, achieve a goal or perform a task [4, 5, 11, 13].

[2]Oxford Dictionary, https://en.oxforddictionaries.com/definition/misconception
[3]This definition is attributed to Jan Cuny, Larry Snyder and Jeannette M. Wing, in an unpublished work ("Demystifying Computational Thinking for Non-Computer Scientists", 2010). Moreover, Wing says it was originated by a discussion with Alfred Aho, who provided a very similar (but with a more "algorithmic thinking" flavor) definition [1]

  - *Logical thinking*: use logical thinking [22] and reasoning to make sense of things, establish and check facts [5].
  - *Problem Decomposition*: split a complex problem in simpler subproblems to solve it more easily [5, 11, 13]; modularize [4]; use compositional reasoning [21].
  - *Abstraction*: get rid of useless details to focus on relevant information/ideas [4, 5, 11, 13, 22].
  - *Pattern recognition*: discover and use regularities in data, problems [5, 11, 22].
  - *Generalization*: use discovered similarities to make predictions or to solve more general problems [5, 11].
- **Methods**: operational approaches widely used by computer scientists.
  - *Automation*: automate the solutions [14, 21]; use a computer or a machine to do repetitive tasks [11, 13].
  - *Data Collection, Analysis and Representation*: gather information/data, make sense of them by finding patterns, represent them properly [11, 13]; store, retrieve and update values [4].
  - *Parallelization*: carry out tasks simultaneously to reach a common goal [4, 11, 13], use parallel thinking [22].
  - *Simulation*: represent data and (real world) processes through models [11, 14], run experiments on models [13].
  - *Evaluation*: implement and analyze solutions [14] to judge them [5], in particular for what concerns effectiveness, efficiency in terms of time and resources [14].
  - *Programming*: use some common concepts in programming (eg. loops, events, conditionals, mathematical and logical operators [4]).
- **Practices**: typical practices used in the implementation of computing machinery based solutions.
  - *Experimenting, iterating, tinkering*: in iterative and incremental software development, one develops a project with repeated iterations of a design-build-test cycle, incrementally building the final result [4]; tinkering means trying things out using a trial and error process, learning by playing, exploring, and experimenting [5].
  - *Test and debug*: verify that solutions work by trying them out [4]; find and solve problems (bugs) in a solution/ program [5].
  - *Reuse and remix*: build your solution on existing code, projects, ideas [4].
- **Transversal skills**: general ways of seeing and operating in the world; useful life skills enhanced by thinking like a computer scientist.
  - *Create*: design and build things [5], use computation to be creative and express yourself [4].
  - *Communicate and collaborate*: connect with others and work together to create something with a common goal and to ensure a better solution [4, 5, 14].
  - *Reflect, learn, meta-reflect*: use computation to reflect and understand computational aspects of the world [4].
  - *Be tolerant for ambiguity*: deal with non-well specified and open-ended real-world problems [14].
  - *Be persistent when dealing with complex problems*: be confident in working with difficult or complex problems [14], persevering, being determined, resilient and tenacious [5].

2.2 Related work

A few works investigated teachers' conceptions about CT, computing and their relation with IT.

Yadav and colleagues conducted two experiments [23, 24] to asses pre-service teachers' "attitudes towards and understanding of computational thinking" and how they changed after attending a CT module in a course of an Education major. Both a pre and a post questionnaire were used in these studies. The first one (N=100) did not have a control group, that was introduced in the second study (N = 294, 141 in the treatment group and 153 in the control group). In both experiments, results showed such module was effective to influence teachers' understanding of CT and to improve their positive attitudes toward CT and its integration into the classroom.

Bower and Falkner [3] conducted a pilot survey on 44 pre-service teachers, investigating their awareness of CT, conceptions regarding the term, use of IT and pedagogical strategies for CT development, and confidence in teaching CT.

Duncan and colleagues [9] report the post-lesson feedbacks from 13 primary school teachers (with no previous experience in teaching computer science) participating in an ongoing study on teaching CT in New Zealand. They report about teacher confidence, level of difficulty of the lessons, common themes emerged in the answers, and teachers' misconceptions.

3 METHODS

3.1 Instrument

Periodical surveys are conducted in "Programma il Futuro" project by means of on-line questionnaires collecting quantitative and qualitative data.

We investigated our research questions in this context. A questionnaire, with some additional questions relevant to the current research, was sent in December 2016, after the CS Educational Week, to all 24,939 teachers enrolled into the project. They filled it out anonymously and we received 3,593 answers up to the end of January 2017.

Teachers belong to all level of schools, from kindergarten to higher secondary schools. Some of them participated this school-year to the project for the first time, others for the second or third time.

For our research, we asked teachers to complete - if they wished - the sentence

Q1 "*In my view computational thinking is…*"

that is we asked them to provide their definition of CT.

We also asked teachers to choose their level of agreement, on a 4-point Likert scale, with the following statements:

Q2 *Being able to use technological devices means having developed computational thinking competences*

Q3 *Computational thinking competences can be adequately developed in primary schools without using technological devices*

We finally asked teachers to grade (4-point Likert scale)

Q4 *How much do you feel prepared to develop computational thinking in your students?*

and to indicate the

Q5 *Most important initiatives to improve your preparation*

by choosing up to 3 answers among:

- training
- availability of technology
- organizational support
- methodological guidelines
- learning objectives and teaching content

In the current study we focus only on answers from primary schools teachers who participated this school-year for the first time (N=972).

3.2 Sample description

We provide here a description of the sample, 93,7% of which are women, apparently not far from the national value (96,4%) for primary school teachers. But this implies 6,3% are men, which is almost the double of the national value (3,6%): this appears to be a confirmation of the current situation where men are more attracted to computing than women.

Figure 1 shows the distribution of teaching seniority in our sample, while figure 2 shows age distribution.

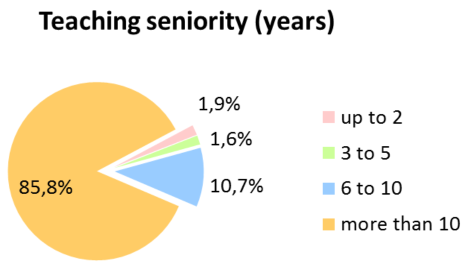

Figure 1: Teacher seniority in years.

Both of them show our sample is made, to a very large extent (>80%), by mature and experienced teachers. This grounds our findings on a reliable base of subjects, but on the other side indicates most probably they have not received any formal or structured training in informatics.

3.3 Procedures

3.3.1 Quantitative analysis. We used standard descriptive statistical methods to analyze closed-ended answers (Q2 to Q5). More specifically, we computed the frequency distribution of these answers.

3.3.2 Qualitative analysis. Among the 972 answers, we filtered out those (116) that did not provide a definition and also those (77) that were completely out of scope (e.g.: they answered “interesting" or “useful").

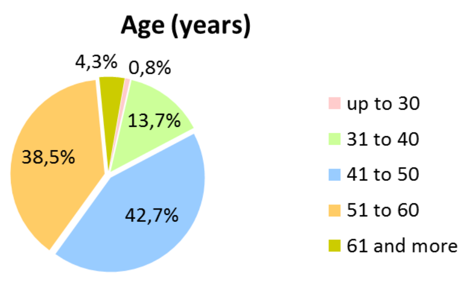

Figure 2: Age of teachers in years.

We then proceeded to identify, by reading and discussing, the conceptual categories present in the remaining 779 definitions.

In a first phase each of us independently analyzed the definitions and proposed a set of conceptual categories to classify them. We used a mixed approach: some categories were defined "a priori", on the basis of literature overview and related work described in section 2, others were grounded on the definitions themselves. We then met to examine the proposed sets of categories and through discussion we agreed to a preliminary set.

We then manually assigned each answer to one or more category, if the statement either declared CT was of the same nature as the category or stated CT had relations to or was useful for the category.

For this process the set of answers was split in three, and each of us assigned answers in his/her set to one or more category. During this process proposals for modifications to categories emerged. Then we met again and jointly examined both these proposed modifications and assignments. Through discussion, we came to agree on the final set of 17 categories (described in subsection 5.1) and the final assignment of each definition to one or more category.

3.3.3 Measuring CT knowledge. To be able to measure the *level* of teachers' knowledge about CT we used the following procedure. We assigned a weight (see discussion in subsection 5.1) to each category according to its relevance (in our view) for CT definition, in the light of the main definitions known in the literature (see subsection 2.1). Finally, the *level* of an answer was computed as the sum of weights of categories it is assigned to.

4 QUANTITATIVE RESULTS

4.1 Technology and computational thinking

The distribution of agreement with the two statements (Q2, Q3) investigating relations between computational thinking and technological devices are respectively shown in figures 3 and 4.

It is positive that almost half of the teachers *disagree* with Q2, and just 17.1% *agree*s or *strongly agree*s with the statement. This shows Italian primary schools teachers have a sufficiently clear understanding that computing is not the same thing as using IT devices. This is supported by the qualitative analysis results discussed in section 5. We observe the results discussed in [3, 23, 24] appear to show a much higher level of misconceptions regarding CT in teachers but we note that those analysis were conducted on

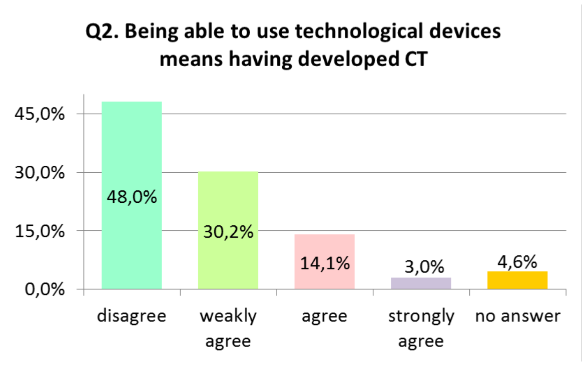

Figure 3: Technological devices and CT.

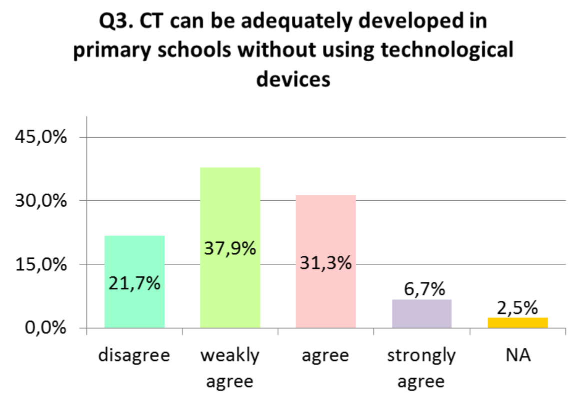

Figure 4: CT without technological devices.

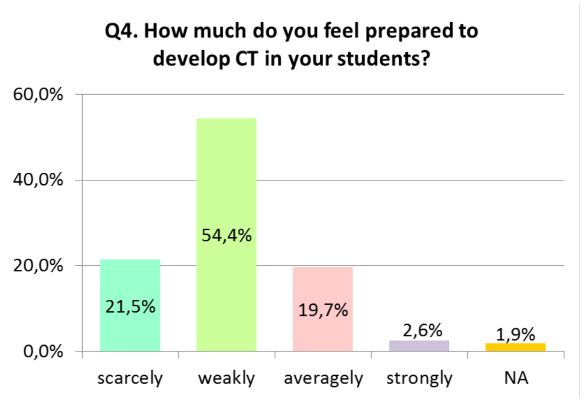

Figure 5: Teachers self-perception of their preparation.

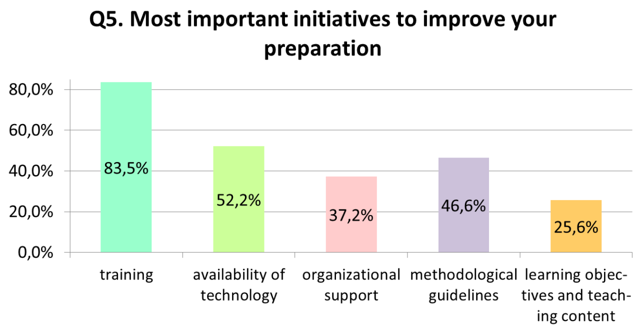

Figure 6: Most important initiative to improve teachers' preparation.

a different (smaller) sample (pre-service teachers) operating in a different culture (USA or Australia).

A positive insight is also given by answers to Q3. In fact, only less than a quarter (21.7%) of teachers thinks an adequate development of CT requires the use of technological devices, while 38.0% *agree*s or *strongly agree*s with Q3. It has also to be noted that spontaneous skill transfer among domain is an unsupported claim [12, 18].

4.2 Teachers' preparation

Self-perception of teachers with respect to their level of preparation to develop CT competences in their students (Q4, Q5) is shown in figure 5.

It is apparent that a large majority does not feel adequately prepared. This is coherent with the following facts regarding Italian schools:

- preparation of primary school teachers is not focused on specific disciplines but has a broad scope
- there is not a specific training program in computing for school teachers of primary and lower secondary levels

Figure 6 shows which initiatives teachers consider most important to improve their preparation (they could choose up to 3 items). Training is by far the most chosen one, which we feel is depending on the nature of our sample. This choice has also a rational support in the positive training effects noted in [23, 24]. A bit more worrying, in our view, is that slightly more than half of the teachers does not feel the need for *methodological guidelines* and just one quarter considers *learning objectives and teaching content* important to improve their preparation.

5 QUALITATIVE RESULTS

5.1 Categories

We now describe the 17 categories emerged from our analysis. We present them in four classes and indicate between parentheses the weight assigned to each category in a class for the purpose of the procedure described in 3.3.2.

- **Fundamental** (+2) - these categories express elements absolutely necessary in any definition of CT.

`PSOL`	Problem solving: action(s) or process(es) leading to solve a problem, to reach a goal, to face a complex situation.
`MENT`	Mental process or tool: a cognitive ability, a mental competence.
`ALGO`	Algorithmic thinking: devising an algorithm to solve a problem; defining an effective method or strategy or plan; solving a problem by means of a sequence of elementary steps.

- AUTO Giving instructions/automation: instructing some agent to solve a problem; providing a procedure to an information processing agent.
- METH Using/learning informatics methods: the ability of using informatics concepts and methods; learning informatics.

- **Important** (+1) - these categories express elements that are important for a definition of CT but are not fundamental.
  - DECO Problem decomposition: splitting a complex problem in simpler subproblems to solve it more easily.
  - LOGI Logical thinking: logical or reasoning or analytical skills.
  - ABST Abstraction: focusing on common characteristic of general value; reusing a solution in other situations; devising a solution for a more general situation.
  - CODE Write programs: writing programs; coding.
- **Part-of** (0) - these categories express elements that are somehow present in definitions of CT reported in the literature; in some sense they are not necessary for a well-formed definition of CT.
  - MCOG Meta-cognition: reflecting about thinking or learning; learning to learn.
  - TRAN Transversal competence: e.g. fourth skill, transversal skill, life skill, useful in other fields, of general use, useful for teaching and learning.
  - CREA Creative thinking: being able to find creative or original solutions to problems; creativity.
  - UNIT Understanding information technology: understanding how information technology devices work; understanding science behind IT.
  - LANG Programming language: a language to communicate with IT devices.
  - ITER Iterative development: operating by means of successive refinements, possibly based on trial and error or testing and debugging.
- **Misleading** (-1) - these categories express elements whose presence in the definition of CT takes away from a correct understanding.
  - THPC "Think" like a computer: act mechanically like a machine, not behaving like a human.
  - UDEV Using IT: being able to use information technology devices and programs as an end-user.

5.2 Analysis of category distribution

The distribution of assignments of definitions to categories is shown in figure 7, where different colors code different classes (remember each definition was classified under one or more category).

To better understand the distribution and its analysis, note that "Programma il Futuro" website provides some introductory information[4], with a discussion on the role of computer science in the digital society and the importance of informatics as an autonomous scientific discipline, based on [10]. It also informally describes what CT is ("*Computational thinking is a mental process for problem-solving with distinctive techniques and general intellectual practices*")[5].

[4]https://programmailfuturo.it/progetto/perche-partecipare
[5]https://programmailfuturo.it/progetto/cose-il-pensiero-computazionale

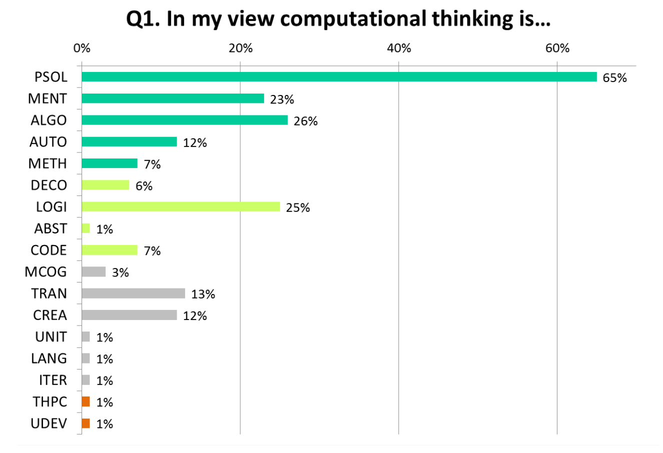

Figure 7: Frequency of each category in Q1.

This may explain why two thirds of the answers identified *problem solving* as an element of the definition of CT.

We note that some categories have a surprisingly high frequency relatively to their importance (in our view) for the definition of CT: *logical thinking*, *transversal competence*, and *creative thinking*. A possible motivation is that a Google query in Italian about computational thinking returns these terms in the first few results.

Also, it is somewhat surprising the low frequency of use of *abstraction* to characterize CT, given the very strong stance taken by Wing in respect to it. We think the conceptual difficulty inherent with the role of abstraction in informatics may explain this outcome.

It is worth noting that *writing programs* has a relatively low frequency: this shows that not so many teachers make the mistake of equating coding and CT.

5.3 Analysis of answer values distribution

5.3.1 Approach. Since all definitions (with their constitutive elements) of CT considered in subsection 2.1, if classified and evaluated with our procedure in 3.3.3, have a value of at least 8, we decided to use this as the threshold to identify the class of "good definitions". We also set the "acceptable definition" threshold at 6. In fact, to reach 6, an answer must have been labeled with categories defined in subsection 5.1 so that it falls within one of the following cases:

(c1) at least 3 fundamental
(c2) 2 fundamental and at least 2 important
(c3) 1 fundamental and 4 important

In other words, there is no way for an answer to be evaluated as an "acceptable definition" if it does not have at least 1 fundamental. But 1 or 2 fundamental alone are not enough, if they are not accompanied by a large enough number of important.

We consider all definitions whose value is 5 or less as misconceptions.

5.3.2 Outcome. Our procedure evaluates just 8 of the 779 answers as "good definitions". The number of those being acceptable

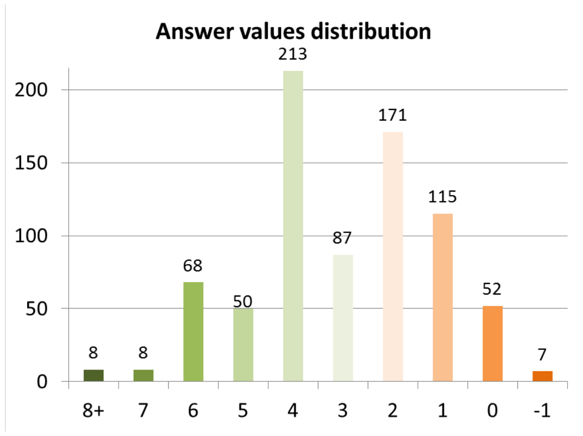

Figure 8: Answer values distribution.

but not good are 76, resulting in a total of just about 10.8% of all answers being at least acceptable. This result appears to be correlated with the feeling of a weak preparation reported in figure 5.

Also, all of not acceptable answers with a value of 5 and 96% of those with a value of 4 have at least 2 fundamental. This leads to a comforting 43.4% of answers that features the presence of at least two fundamental components for a CT definition.

The 695 not acceptable answers (i.e., the misconceptions) are roughly evenly split among those with a value at least 3, and those with a value less than 3: see in figure 8 the overall distribution.

Moreover, we investigated the frequency with which each couple of categories appeared in the definition. We report in table 1 the 27 most frequent couples with at least 2% frequency. This table

Table 1: Most frequent (%) couples of categories.

	MENT	ALGO	AUTO	METH	DECO	LOGI	CODE	TRAN	CREA
PSOL	17	22	8	5	5	10	2	6	8
ALGO	4				2	4		2	
AUTO	2	2				2	2		
METH	4							2	
LOGI	2				2			2	
TRAN	5								
CREA	2	3				4		2	

therefore shows the frequency with which (at least) both categories have labeled answers, that is their frequency of co-occurrence. Row and column headings appear in the same order as in subsection 5.1 and, to make the table more compact, not all categories are listed.

Note that PSOL plays a leading role, which is understandable given two thirds of definitions have received its label. A positive element is the relatively high frequency of co-occurrence of PSOL with ALGO (22%): this can be interpreted as an evidence that that about 11% of answers (22%-10.8%), even if not acceptable, are characterized by a sound (even if incomplete) description of computing.

5.4 Conceptions and misconceptions regarding CT

Examining all the answers that are at least acceptable we observe just 29 distinct sets. Table 3 on next page shows the count, the value according to our procedure in 3.3.3, and the constituent categories of each set.

Table 2: Distinct sets and counts of not acceptable **answers**

Value	Count	Labels
...		
4	88	PSOL, ALGO
4	51	PSOL, MENT
4	28	PSOL, AUTO
...		
3	24	PSOL, LOGI
3	11	PSOL, DECO
...		
2	80	PSOL
2	19	PSOL, CREA
2	11	MENT
...		
1	67	LOGI
1	13	CODE
1	11	LOGI, CREA
...		
0	26	TRAN
...		

There is no example of a set belonging to case (c3), see 5.3.1, and just 4 lines of the table show sets belonging to case (c2), explicitly indicated in the table, meaning the overwhelming majority of acceptable answers belongs to case (c1).

Moreover, three different sets have a high count (marked with a * in the "Case" column): {PSOL, MENT, METH}, {PSOL, MENT, ALGO}, and {PSOL, MENT, METH TRAN}. We think this is a positive result since these answers are all instances of case (c1), even if many examples in these sets are clearly molded after the information provided in “Programma il Futuro” website.

The most frequent not acceptable answers are shown in table 2.

A large number of misconceptions is characterized either by PSOL alone or by its coupling with exactly one of these categories: MENT, LOGI, DECO, CREA (first 7 lines of the table). In all these cases a very partial view of informatics emerges, given the absence of categories describing the *information-processing agent*. A similar situation happens for MENT and LOGI (next 2 lines). This reinforces our concerns that considering CT as a subject somewhat distinct from computing may give raise to misconceptions about IT.

Another misconception is shown by the relatively high count of TRAN alone (last line), which shows the evidence of a view of CT as an instrumental discipline, not important in itself. This is possibly deriving from attempts to convince teacher of the importance of CT by focusing mainly on its value for other disciplines and as a general learning tool.

Table 3: Distinct sets and counts of acceptable **answers**

Value	Count	Case	Labels										
12	1		PSOL	MENT	ALGO	AUTO	METH	DECO	LOGI				
11	1		PSOL	MENT	ALGO		METH	DECO	LOGI	ABST		CREA	
9	1		PSOL	MENT	ALGO		METH			ABST			
8	3		PSOL	MENT	ALGO	AUTO							
8	1		PSOL	MENT	ALGO		METH				TRAN	CREA	
8	1		PSOL	MENT	ALGO		METH						
7	2		PSOL	MENT	ALGO			DECO					
7	2		PSOL		ALGO	AUTO			LOGI				
7	1		PSOL	MENT			METH		LOGI		TRAN	CREA	
7	1		PSOL	MENT	ALGO					ABST	TRAN		
7	1		PSOL		ALGO		METH		LOGI				
7	1		PSOL		ALGO	AUTO				ABST			
6	17	*	PSOL	MENT			METH						
6	11	*	PSOL	MENT	ALGO								
6	10	*	PSOL	MENT			METH				TRAN		
6	7		PSOL		ALGO	AUTO							
6	6		PSOL	MENT		AUTO							
6	2	c2	PSOL		ALGO			DECO	LOGI			CREA	
6	2	c2	PSOL		ALGO			DECO	LOGI				
6	2		PSOL	MENT	ALGO						TRAN	CREA	
6	2		PSOL	MENT	ALGO						TRAN		
6	2		PSOL	MENT	ALGO							CREA	
6	1	c2	PSOL			AUTO		DECO	LOGI				
6	1		PSOL	MENT		AUTO					TRAN		
6	1		PSOL	MENT		AUTO							ITER
6	1		PSOL		ALGO	AUTO					TRAN		ITER
6	1	c2	PSOL		ALGO				LOGI	ABST			
6	1		PSOL		ALGO		METH						
6	1		PSOL			AUTO	METH						

6 CONCLUSIONS AND FURTHER WORK

Outcome of our work shows the vast majority of Italian primary school teachers has not a sound and complete conception about CT (RQ1).

This negative finding is somewhat balanced by the evidence regarding teachers in relation to information technology (IT). In fact, it is sufficiently clear to them that (1) computer science and the use of IT are two distinct fields, and (2) IT devices are not absolutely needed to develop CT competences in students (RQ2).

Finally, teachers feel themselves not enough prepared to develop CT competences in their students and identify in specific training the most important initiative (RQ3).

What we reported in this paper is only a first analysis of the situation in Italy regarding CT in schools in relation to "Programma il Futuro". We plan to complete the analysis by considering also answers from teachers at all school levels, and by investigating possible differences between teachers newly came to the project and those involved since the beginning.

ACKNOWLEDGMENTS

We greatly thank teachers and students involved in Programma il Futuro project (coordinated by EN and Giorgio Ventre) and Code.org for its cooperation.

We acknowledge the financial support of TIM; Engineering; CA Technologies, Cisco, De Agostini Scuola; SeeWeb. Other companies have financially supported the project during the first two school-years only: Samsung Italia; Microsoft Italia; Hewlett-Packard; Oracle; Facebook.

Rai Cultura, the culture department of Italian national public broadcasting company, is a media partner of the project since February 2017.

ML greatly thanks University of Bologna for funding his research and his Ph.D. advisor, Simone Martini, for fruitful discussions and support.

We greatly thank anonymous referees for insightful comments and suggestions.

REFERENCES

[1] Alfred V. Aho. 2011. Ubiquity Symposium: Computation and Computational Thinking. *Ubiquity* 2011, January, Article 1 (Jan. 2011). https://doi.org/10.1145/1922681.1922682

[2] Michal Armoni. 2016. COMPUTING IN SCHOOLS: Computer Science, Computational Thinking, Programming, Coding: The Anomalies of Transitivity in K-12 Computer Science Education. *ACM Inroads* 7, 4 (Nov. 2016), 24–27. https://doi.org/10.1145/3011071

[3] Matt Bower and Katrina Falkner. 2015. Computational Thinking, the Notional Machine, Pre-service Teachers, and Research Opportunities. In *Proceedings of the 17th Australasian Computing Education Conference (ACE 2015)*, Vol. 27. 30.

[4] Karen Brennan and Mitchel Resnick. 2012. 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*. 1–25.

[5] Barefoot CAS. 2014. Computational Thinking. (2014). Retrieved April 4, 2017 from http://barefootcas.org.uk/wp-content/uploads/2014/10/Computational-thinking-Barefoot-Computing.pdf

[6] Isabella Corradini, Michael Lodi, and Enrico Nardelli. 2017. Computational Thinking in Italian Schools: Quantitative Data and Teachers' Sentiment Analysis after Two Years of "Programma il Futuro" Project. In *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '17)*. ACM, New York, NY, USA. https://doi.org/10.1145/3059009.3059040

[7] Peter J. Denning. 2009. The Profession of IT: Beyond Computational Thinking. *Commun. ACM* 52, 6 (June 2009), 28–30. https://doi.org/10.1145/1516046.1516054

[8] Peter J. Denning, Matti Tedre, and Pat Yongpradit. 2017. Misconceptions About Computer Science. *Commun. ACM* 60, 3 (Feb. 2017), 31–33. https://doi.org/10.1145/3041047

[9] Caitlin Duncan, Tim Bell, and James Atlas. 2017. What Do the Teachers Think?: Introducing Computational Thinking in the Primary School Curriculum. In *Proceedings of the Nineteenth Australasian Computing Education Conference (ACE '17)*. ACM, New York, NY, USA, 65–74. https://doi.org/10.1145/3013499.3013506

[10] Informatics Europe and ACM Europe. 2013. Informatics education: Europe cannot afford to miss the boat. (2013). Retrieved April 4, 2017 from http://europe.acm.org/iereport/ACMandIEreport.pdf

[11] Google. 2017. Exploring Computational Thinking. (2017). Retrieved April 4, 2017 from http://g.co/exploringct

[12] Mark Guzdial. 2015. *Learner-Centered Design of Computing Education: Research on Computing for Everyone*. Morgan & Claypool Publishers. http://dx.doi.org/10.2200/S00684ED1V01Y201511HCI033

[13] ISTE and CSTA. 2011. Computational Thinking teacher resources. (2011). Retrieved April 4, 2017 from https://c.ymcdn.com/sites/www.csteachers.org/resource/resmgr/472.11CTTeacherResources_2ed.pdf

[14] ISTE and CSTA. 2011. Operational Definition of Computational Thinking for K-12 Education. (2011). Retrieved April 4, 2017 from https://c.ymcdn.com/sites/www.csteachers.org/resource/resmgr/CompThinkingFlyer.pdf

[15] Enrico Nardelli and Giorgio Ventre. 2015. Introducing Computational Thinking in Italian Schools: A First Report on "Programma Il Futuro" Project. In *INTED2015 Proceedings (9th International Technology, Education and Development Conference)*. IATED, 7414–7421.

[16] Seymour Papert. 1980. *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books, Inc., New York, NY, USA.

[17] Seymour Papert. 1996. An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning* 1, 1 (1996), 95–123. https://doi.org/10.1007/BF00191473

[18] Roy D. Pea and D.Midian Kurland. 1984. On the cognitive effects of learning computer programming. *New Ideas in Psychology* 2, 2 (1984), 137 – 168. https://doi.org/10.1016/0732-118X(84)90018-7

[19] Juha Sorva. 2013. Notional Machines and Introductory Programming Education. *Trans. Comput. Educ.* 13, 2, Article 8 (July 2013), 31 pages. https://doi.org/10.1145/2483710.2483713

[20] Jeannette M. Wing. 2006. Computational Thinking. *Commun. ACM* 49, 3 (March 2006), 33–35. https://doi.org/10.1145/1118178.1118215

[21] Jeannette M. Wing. 2008. Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 366, 1881 (Oct 2008), 3717–3725. https://doi.org/10.1098/rsta.2008.0118

[22] Jeannette M. Wing. 2010. Computational Thinking: What and Why? *Link Magazine* (2010).

[23] Aman Yadav, Chris Mayfield, Ninger Zhou, Susanne Hambrusch, and John T. Korb. 2014. Computational Thinking in Elementary and Secondary Teacher Education. *Trans. Comput. Educ.* 14, 1, Article 5 (March 2014), 16 pages. https://doi.org/10.1145/2576872

[24] Aman Yadav, Ninger Zhou, Chris Mayfield, Susanne Hambrusch, and John T. Korb. 2011. Introducing Computational Thinking in Education Courses. In *Proceedings of the 42Nd ACM Technical Symposium on Computer Science Education (SIGCSE '11)*. ACM, New York, NY, USA, 465–470. https://doi.org/10.1145/1953163.1953297