ChIP: teaching coding in primary schools

Fabio Sorrentino,
Lucio Davide Spano,
Sara Casti,
Alessandro Carcangiu,
Fabrizio Corda,
Gianmarco Cherchi,
Alessio Murru,
Alessandro Muntoni,
Stefano Nuvoli,
Riccardo Scateni
Department of Mathematics and
Computer Science
University of Cagliari
Via Ospedale 72, 09124,
Cagliari, Italy
{cg3hci@gmail.com}

Abstract
In this paper, we introduce the ChIP project, which aims at teaching coding in primary schools by using a series of educational games and hands-on activities. The overall idea is to determine the benefits and costs of this activity and to explore the meaning and the potential impact of learning to code for younger students. We describe the lessons learned from two 60 hours courses that took place in two different primary schools thus detailing the pros and cons of this experience.

Author Keywords
Coding; Learning; Robotics; Primary school;

ACM Classification Keywords
H.5.m. [Information Interfaces and Presentation (e.g. HCI)]: Miscellaneous

Introduction
Nowadays, technologies are acquiring more and more relevance for young students. We find several examples and demonstrations of how these new technologies help learners in extremely different fields [7, 3]. However, simply adopting new technologies in the process of learning does not guarantee better results. Blacke affirms that the technology is theoretically and methodologically neutral: the key point is how the technology is applied [1].

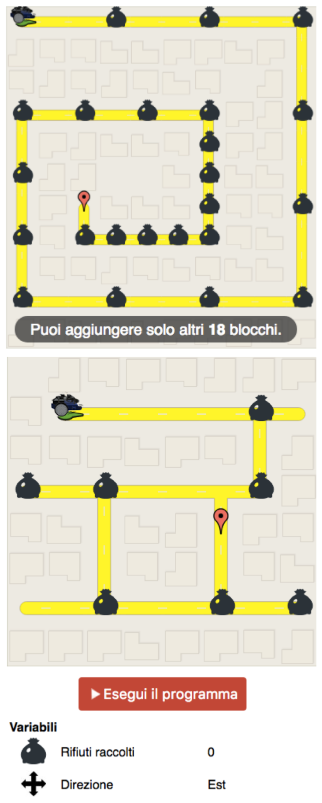

Figure 1: Different levels from the developed Chip game.

In the last years, the debate on when and how starting to teach computer science is becoming increasingly relevant. There are many programming platforms such as Scratch [1], Alice [2], Blockly [3], and Kodu [4] that can be used to teach students coding/programming and Computer Science in general. There are several educational games that exploit these platforms and some of them are oriented to the robotic programming.

Learning and playing are two strictly related aspects, and they are natural components of children's everyday lives. Playing is fun for children, but at the same time, it represents one of the ways they actually learn [5, 2, 6]. Following this general idea, the ChIP (a Children Introduction to Programming) project has three main objectives: (i) to teach coding to pupils through educational games, (ii) to develop their problem-solving and comprehension skills, and (iii) to foster their key skills such as creativity, autonomous learning, and social interaction.

ChIP

The project set as a learning outcome for the students the understanding of the main programming principles: simple statements, program state management (variables), conditional and iterative statements, simple functions.

The project took place in two different primary schools in Cagliari, from March to June. For each lesson were present three coding tutors and one teacher. During the lessons, the tutors actively supported students in their tasks by giving them suggestions and little hints in order to help them in reaching the goal. In some cases, students were allowed to work in pairs.

[1]http://scratch.mit.edu
[2]http://www.alice.org
[3]http://blockly-games.appspot.com
[4]http://www.kodugamelab.com

Course organization
The first classroom (C1) included 29 pupils, 12 boys and 17 girls aged between 9 and 11. The second one (C2) included 25 pupils, 14 boys and 11 girls aged between 6 and 11. In the former, we were able to create a single group since the children's age was homogeneous. In the latter, we created three groups: (a) 3 girls and 5 boys aged between 6 and 7, (b) 5 girls and 3 boys aged between 8 and 9, (c) 3 girls and 6 boys aged between 10 and 11. In both classrooms were present pupils with ADHD (Attention Deficit Hyperactivity Disorder) and the role of teachers was crucial to foster and manage their participation during course activities.

During the lesson, tutors used informal assessment techniques, such as asking questions or simply watching the students' current performance on a task. They walked around while students were working on an activity, and observed the progress made by each pupil. This way allowed tutors to monitor each child's progress on a specific task thus taking notes of the child's current understanding and any areas in which the pupil may need assistance. If the students appeared confused or not interested in the lesson, the tutors adjusted their activities accordingly before reaching the end. In particular, due to the fact that some lessons lasted from two up to four hours (split into two main sessions), we allowed pupils to play videogames or to draw by using MS Paint during the last 20 minutes of the lesson. A short break between the main sessions helped tutors to keep a high attention level of pupils during the activities. Specifically, we followed three methodologies during the entire course: proper lesson, learning-by-doing, and hands-on laboratory. These methodologies have been applied by using different tools: coding with paper, educational games at increasing difficulty on Code.org, assessment games of our developed ChIP platform, robotic programming.

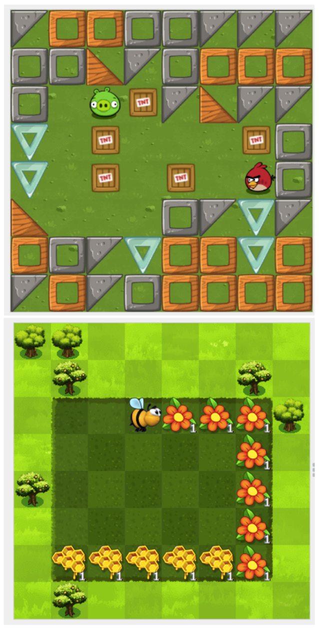

Figure 2: Different levels from Code.org

Teaching methodologies
Since it was the first coding course for the majority of the pupils, they needed the introduction to Computer Science fundamentals before starting the activities. These fundamentals were taught by using everyday life examples, letting pupils understand what is a computer, how it works and how it is possible to program it in order to reach a specific goal. We had a short discussion on what an algorithm was, i.e. a set of steps or instructions that they would create and that they would then execute. This has been done as a group discussion and, lesson by lesson, they were able to understand the meaning of instructions, operators, iterators, conditional statements, functions and more.

During the explanations, tutors used informal assessment techniques, such as asking questions, in order to get the children's current understanding. The learning-by-doing methodology has been largely used during the entire course. It consists in solving educational games by using a laptop. These games were proposed with increasing difficulty. In some cases, pupils were assisted by tutors to reach the problem solution or to produce a better one. During the hands-on sessions, pupils were supported in solving a problem in small groups. At the beginning of the course, they built a robot named Chip (Makeblock mBot[5]) under the supervision of a tutor.

A video report is available at https://youtu.be/wF4Lztzp9Ic.

Teaching tools
We used different tools and techniques for introducing programming concepts to young students in fun and interactive ways. One was coding with paper that was useful for them in learning programming fundamentals at the beginning of the course since the medium did not constrain them and did not added the difficulty of using the desktop computer. We printed and cut a set of simple instructions on paper tags: *turn left, turn right, turn on leds* and more where the young programmers had to find their right order to achieve a task. The created algorithm was then executed to see if the solution was the correct one. During these sessions, we formed groups of two pupils. One was the *programmer*, while the the other one was the *robot*. We also adopted pixel-art techniques to let pupils draw by using a sort of algorithm made up by a list of direction to follow.

The use of educational games by using a laptop was a crucial activity during the lesson. First, they used a subset of Blockly games, then they started to use the Code.org[6] games that are based on Blockly too (see Fig. 2). Blockly is a visual programming language, where users drag and drop blocks together in order to generate code. Code.org offers the opportunity for students to learn computer science by using this approach. At the same time, it offers several statistics and tools to educators.

In Code.org, the students of the first classroom (C1) worked on course 2, 3, and 4. Course 2 is dedicated to students who can read and have no prior programming experience. In this course, students will create programs to solve problems and develop interactive games or stories they can share (recommended for grades 2-5). In course 3 students are lead into programming topics introduced in previous courses to create flexible solutions to more complex problems (recommended for grades 4-5). In course 4 students will learn how to solve puzzles with increased complexity as they learn how to combine several concepts (including loops and functions with parameters) when solving each challenge (recommended for grades 4-8). In C2 it has been

[5]http://www.makeblock.com

[6]https://code.org

necessary for some pupils (those aged between 6 and 7) to start with the course 1 that is designed for early readers.

In combination with these games, pupils used a customised version of the Blockly platform that was developed by our group where the main character is the robot Chip and it has to collect bags of garbage while finding the maze exit (see Fig. 1). This web game [7] has been made *real* by using an mBot robot. It has been programmed by the pupils with the support of tutors. The device mBot is a cheap Arduino-based robotics kit that allows students to assemble it with motors and sensors without requiring students to know anything about electronics. Both sensors and motors are easy to plug with cables. Due to the fact that it is equipped with an Arduino board, it is possible to programming it with the Arduino programming language or by using the Scratch environment mBlock. The device can be remotely controlled by Bluetooth (or WiFi, it depends on the version). We built a small-size maze where the posts and pegs were 3D printed and the rest of the construction was 4mm wooden board and black masking tape.

Conclusion and Future Work

As shown in Fig. 3 and 4, it has been reached a really good result in term comprehension of the proposed topics in both C1 and C2 classrooms. Specifically, T2.1 corresponds to information representation, T2.2 to simple instructions, T2.3 to conditional instructions, T2.4 to iterative instructions and T2.5 to functions and procedures. In both classrooms, a critical topic was the T2.5, due to the fact that it requires a good ability of abstraction that is not yet developed during primary school age. In fact, in the concrete operational stage (from age 7 to 11), they begin to think more logically and they tend to struggle with abstract and hypothetical

[7] http://spano.sc.unica.it/chip/appengine/index.html?lang=it

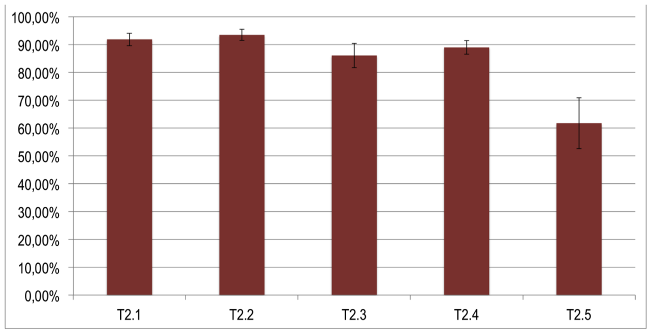

Figure 3: C1 - Percentage of successfully solved exercises, aggregated by topic. Bars show the standard error around the central value.

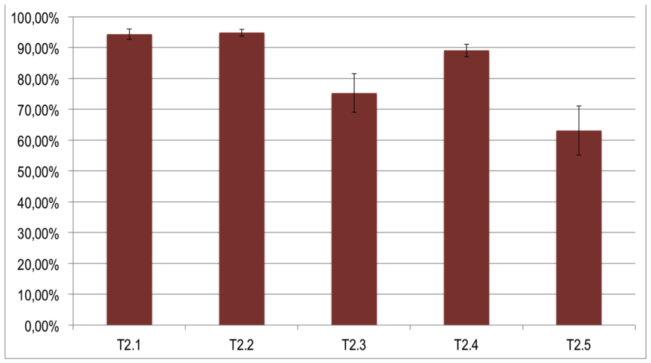

Figure 4: C2 - Percentage of successfully solved exercises, aggregated by topic. Bars show the standard error around the central value.

concepts [4]. This topic was hard to do for its graphical representation too. The block used for represent a function or a procedure and its position inside the working view can confuse the student on which block will be executed for first. This would require more research in order to find more suitable representations for children.

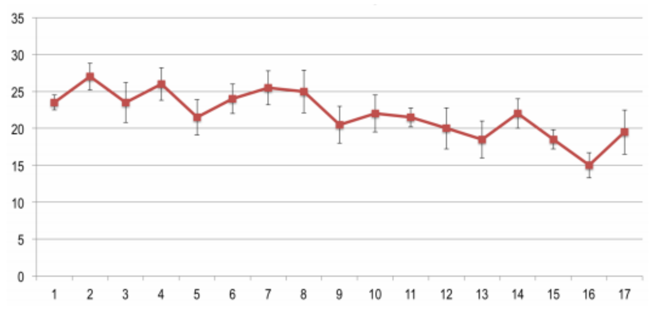

Figure 5: Number of tutors interventions per hour for each lesson. Bars show the standard error around the central value.

During the course, tutors have counted the number of hints given to students (see Fig. 5). The main goal of this activity, that is shown in Figure 5, was to check if these hints produced a more autonomous problem-solving capability. The data shows that lesson after lesson this number follows a decreasing trend, due to two main motivation: the former is the progressive comprehension of topics and the acquisition of the computational thinking. The latter is the established spontaneous collaboration made between classmates.

Acknowledgements

The authors are grateful to the children, teachers and the class presidents of both Istituto Comprensivo n. 4 in Quartu Sant'Elena (CA) and Primo Circolo Didattico Selargius (CA) who participated in our study.

References

[1] Robert J Blake. 2013. *Brave new digital classroom: Technology and foreign language learning*. Georgetown University Press.

[2] Pat Broadhead. 2006. Developing an understanding of young children's learning through play: the place of observation, interaction and reflection. *British Educational Research Journal* 32, 2 (2006), 191–207.

[3] Andrea Casu, Lucio Davide Spano, Fabio Sorrentino, and Riccardo Scateni. 2015. RiftArt: Bringing Masterpieces in the Classroom through Immersive Virtual Reality.. In *Eurographics Italian Chapter Conference*. 77–84.

[4] Jean Piaget. 2000. Piaget's theory of cognitive development. *Childhood cognitive development: The essential readings* (2000), 33–47.

[5] Maria Roussou. 2004. Learning by doing and learning through play: an exploration of interactivity in virtual environments for children. *Computers in Entertainment (CIE)* 2, 1 (2004), 10–10.

[6] Ingrid Pramling Samuelsson and Maj Asplund Carlsson. 2008. The playing learning child: Towards a pedagogy of early childhood. *Scandinavian journal of educational research* 52, 6 (2008), 623–641.

[7] Fabio Sorrentino. 2016. *New HCI techniques for better living through technology*. Ph.D. Dissertation. University of Cagliari.