# Chapter 12
# Using Dr. Scratch as a Formative Feedback Tool to Assess Computational Thinking

**Peter Rich**
https://orcid.org/0000-0002-0039-966X
*Brigham Young University, USA*

**Samuel Frank Browning**
*AllenComm, USA*

## ABSTRACT

*This study investigated if using Dr. Scratch as a formative feedback tool would accelerate students' Computational Thinking (CT). Forty-one 4th-6th grade students participated in a 1-hour/week Scratch workshop for nine weeks. We measured pre- and posttest results of the computational thinking test (CTt) between control (n = 18) and treatment groups (n = 23) using three methods: propensity score matching (treatment = .575; control = .607; p = .696), information maximum likelihood technique (treatment effect = -.09; p = .006), and multiple linear regression. Both groups demonstrated significantly increased posttest scores over their pretest (treatment = +8.31%; control = +5.43%), showing that learning to code can increase computational thinking over a 2-month period. In this chapter, we discuss the implications of using Dr. Scratch as a formative feedback tool the possibilities of further research on the use of automatic feedback tools in teaching elementary computational thinking.*

## INTRODUCTION

School systems around the world have been adopting and even requiring that computer science (CS) or computational thinking (CT) become part of their curriculum. In Europe, coding integration in the curriculum has seen a large adoption rate, "at the national, regional or local level" (Balanskat & Engelhardt, 2015, p. 9), including 16 countries with two more having plans to integrate coding into their core curriculum by 2020. CT and the fundamentals of coding are also starting to be introduced in K–12

DOI: 10.4018/978-1-7998-1479-5.ch012

schools throughout the United States (Rich, Bly, & Leatham, 2014; Elahi, 2016; Grover & Pea, 2013; K–12 Computer Science Framework, 2016; Repenning, Webb, & Ioannidou, 2010; Smith, 2016). With so many schools adopting coding at earlier ages, there is a need to better understand more and less effective methods to assess the computational ability of younger students.

While assessment in itself is important to measure progress and goal attainment, *formative* assessment can be used by learners themselves to measure the growth of their own CT ability over time and across projects. Educational research has demonstrated that feedback is an important and effective learning intervention. According to Hattie (2015), performance feedback measures are one of the most effective forms of intervention. Hattie's research data is pulled from nearly 1200 meta-analyses and his list has grown to 195 influencers on student achievement. Feedback consistently ranks highly significant in Hattie's meta analyses, currently ranking 15th largest in effect size on student achievement (ES = 0.73; Hattie, 2015; Visible Learning, 2016).

Feedback may be defined as: "the means by which the learner, or any other agent directing the learning process, ascertains whether or not progress is being made toward the end goal, and whether or not the goal has been reached" (Weibell, 2011, p. 361). The goal in measuring progress and providing feedback in CT can be looked at in two ways: (a) the completion of or progress toward solving the problem to which computational thinking processes are applied, or (b) the ability of the student/learner to apply CT processes correctly to any given task/problem. Feedback towards the goal of solving the problem can be given by teachers and peers, or even by the way the learner's program interacts with the problem (e.g., student's program does not solve or partially solves the given problem). Feedback towards the goal of applying CT processes correctly would most likely require a way to understand how the learner was thinking during the creation process. One way to do this would be to analyze the artifact(s) a student/learner creates and what CT processes would have been needed to create those artifacts.

According to a recent research, Scratch (https://scratch.mit.edu/) is one of the most popular ways to teach coding in K-8 throughout the world (Rich at al., 2019). Despite its popularity, Scratch lacks a built-in feedback tool to inform students and teachers of a student's progress in their computational thinking ability. Dr. Scratch was built to fill this need. Dr. Scratch seeks to give feedback on the application of CT processes a student/learner uses to solve a problem. It does this by analyzing a Scratch project's artifact(s) (e.g., coding blocks, sprites, variables) created by the student/learner in their attempt to solve their problem. The reasoning behind Dr. Scratch's analysis is that students who correctly use complex programming blocks will most likely have used the CT concepts and processes required to understand the function of those blocks. For example, a student's understanding of data representation would be categorized as follows: one point for using blocks that modify a sprite's attributes (e.g., orientation & position), two points for using variables, and three points for the use of lists (Moreno-León, Robles, & Román-González, 2015). Dr. Scratch gives a score like this in seven different categories (see Table 1). Dr. Scratch combines the scores from each of the seven categories into one CT score that is shown to the students along with the points given in each section. Dr. Scratch's CT score has "moderate to strong correlations" that are positive and significant with McCabe's Cyclomatic Complexity and Halstead's metrics (Moreno-León, Robles, & Román-González, 2016, p. 1044), common complexity metrics used in software development. Dr. Scratch provides an opportunity to examine how providing formative feedback might influence their computational thinking. The purpose of this study was to better understand if the use of an automated project-analysis tool as formative feedback positively impacts students' computational thinking ability.