Studies in Educational Evaluation 58 (2018) 132–144

Contents lists available at ScienceDirect

Studies in Educational Evaluation

journal homepage: www.elsevier.com/locate/stueduc

The effect of reciprocal teaching and programmed instruction on learning outcome in computer science education

Andreas Zendler*, Swetlana Reile

University of Education Ludwigsburg, Reuteallee 46, D-71634, Ludwigsburg, Germany

ARTICLE INFO

Keywords:
Computer science education
Instructional methods
Reciprocal teaching
Programmed instruction
Experimental study
Learning outcome

ABSTRACT

Answers to the questions of which instructional methods are suitable for school, what instructional methods should be applied in teaching individual subjects and how instructional methods support the act of learning represent challenges to general education and education in individual subjects. This study focuses on the empirical examination of learning outcome with respect to two instructional methods: programmed instruction and reciprocal teaching. An SPF-2 × 2•2 design is used to control instructional method, time and class context. Learning outcome on search queries is assessed with reference to multi-choice test items. The empirical findings show that learning with programmed instruction performs better than reciprocal teaching.

1. Introduction

Answers to the questions of which instructional methods are suitable for school, what instructional methods should be applied in teaching individual subjects and how instructional methods support the act of learning represent challenges to general education and education in individual subjects. The wide range of instructional methods is almost incomprehensible.

The Center for Teaching and Learning (2018) cites 150 instructional methods, Gugel (2011) more than 2000 methods including their variations. Handbooks describing instructional methods are provided by authors such as Ginnis (2001); Abell and Lederman (2007); Davis (2009); Petty (2009).

A useful definition of method which also represents the conceptual starting point for this study comes from Huber and Hader-Popp: "The word method is understood to mean a clearly defined, conceptually perceivable and independent, if also integrated, component of teaching." (Huber & Hader-Popp, 2007, p. 3)

Empirical findings on the effectiveness of learning are numerous. In his compilation of 800 meta-analyses into which more than 50,000 studies were included Hattie provides information on the influences on learning with respect to six domains: contributions of the person learning, the parental home, the school, the instructor, the curricula and teaching. In particular, the domain of teaching (Hattie, 2009, chapters 9 and 10) provides information on the effectiveness of instructional methods/approaches. High effect sizes ($d > 0.50$) were demonstrated for microteaching ($d = .88$), reciprocal teaching ($d = .74$), feedback ($d = .73$), problem solving ($d = .61$), direct instruction ($d = .59$), mastery learning ($d = .58$), case study ($d = .57$), concept mapping ($d = .57$), peer tutoring ($d = .55$), cooperative (vs. competitive) learning ($d = .54$) and interactive instructional videos ($d = .52$).

There is as yet no standard reference work in neither German nor English literature for computer science education which extensively addresses the application of instructional methods for school (Cornelson, 2018). The German-language literature contains a few brief chapters on the application of learning assignments, group work, learning programs, discovery learning and project teaching (Hartmann, Näf, & Reichert, 2006), problem solving and project teaching (Humbert, 2006), problem-orientation along with modeling and simulation (Hubwieser, 2007), problem solving, experiments and project teaching (Schubert & Schwill, 2012). The English-language literature contains descriptions on the application of "solving problems" (Koffmann & Brinda, 2003), "group work" (Irons, Alexander, & Alexander, 2004), "rich tasks", "concept maps" (Hazzan, Lapidot, & Ragonis, 2011), and "visualizations" (Agneli, Kadijevich, & Schulte, 2013; Fincher & Petre, 2004).

Several articles in the computer science educational magazine LOG IN (2018) are interesting from methodological and practical teaching standpoints. LOG IN already raised awareness of the necessity of new methods in computer science education ten years ago (Seiffert & Koerber, 2003). Among the writings found in the LOG IN heading „Practice & Methodology" there are reports featuring the following instructional methods: direct instruction, inductive approaches, discovery learning, experiment, concept mapping, problem solving, self-directed learning, project teaching, simulation and modeling, and role-play.

* Corresponding author.
E-mail addresses: zendler@ph-ludwigsburg.de (A. Zendler), reile@ph-ludwigsburg.de (S. Reile).

https://doi.org/10.1016/j.stueduc.2018.05.008
Received 31 December 2017; Received in revised form 24 March 2018; Accepted 15 May 2018
0191-491X/ © 2018 Published by Elsevier Ltd.

The search through the English-language magazines and conference reports on computer science education (Journal of Educational Computing Research, Computer Science Education, ACM Transactions on Computing Education, Special Interest Group Computer Science Education Bulletin) provided findings related to computer science education in regard to constructivist teaching activities (Gorp van & Grissom, 2001), the "eXtreme teaching" approach (Andersson & Bendix, 2006), holistic teaching and learning (Thota & Whitfield, 2010), the influence of instructional methods on the design of computer programs (Hung, 2012), the effect of games on motivation in teaching (Freitas de & Freitas de, 2013), the reduction of learning content (Kilpeläinen, 2010), the application of formal modeling (Carro, Herranz, & Mariño, 2013), the effectiveness of two-person team programming (Braught, Wahlks, & Eby, 2011) and the application of the experiment (Schulte, 2012).

The curriculum of ACM Association for Computing Machinery (2013) does not contain any recommendations on the use of instructional methods for computer science classrooms. The *Bildungsstandards Informatik für die Sekundarstufe I* (Computer Science Educational Standards for Lower Secondary Education) (GI, 2018) recommend various instructional methods (e.g. direct instruction, project, group and free work) and learning forms (e.g. subject-related, interdisciplinary, self-directed learning).

For the assessment of instructional methods, a booklet (Zendler & Klaudt, 2015b) was developed at the Department of Mathematics and Computer Science of the University of Education Ludwigsburg, which contains the following 20 instructional methods for computer science education: case study, computer simulation, concept mapping, direct instruction, discovery learning, experiment, Leittext method, jigsaw method, learning stations, learning by teaching, learning tasks, models method, portfolio method, presentation, problem-based learning, programmed instruction, project work, reciprocal teaching, role-play, and web quest. All methods are described according to a uniform schema that provides information on the use of the individual methods in computer science education.

The heat map seen in Fig. 1 contains the means for the 20 instructional methods with respect to the six knowledge processes: *build, process, apply, transfer, assess*, and *integrate* (Zendler & Klaudt, 2015b). The heat map also contains the grand means of means of the knowledge processes for the instructional methods. The instructional methods are sorted in accordance with these grand means.

Fig. 1 shows that problem-based learning was assessed by the computer science teachers as the best method for supporting the act of learning in computer science education; this method is followed by five additional instructional methods: learning tasks, discovery learning, computer simulation, project work, and direct instruction.

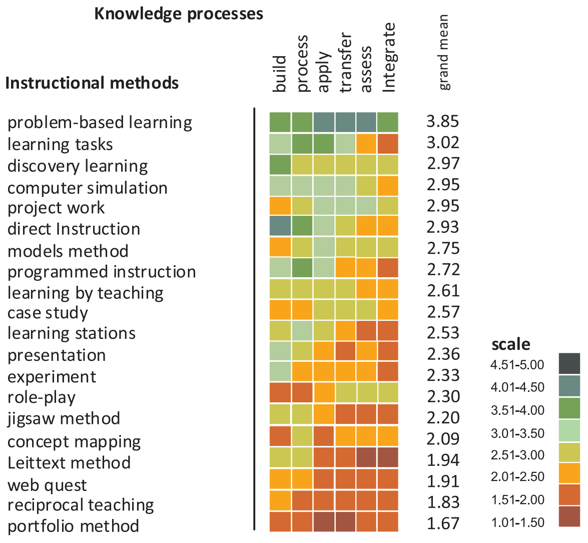

Fig. 1. Means of the instructional methods visualized for the knowledge processes.

In a more detailed observation the heat map reveals that problem-based learning is distinguished by high values (> 3.50) for all knowledge processes. Learning tasks is characterized by high values for the knowledge processes of *process* and *apply*. Discovery learning demonstrates high values for the knowledge process *build*. Particularly high values (> 4.00) for the knowledge process *build* are shown by direct, which additionally has relatively high values (> 3.00) for the knowledge processes of *process* and *apply*. Whereas computer simulation is characterized by relatively high values for the first four knowledge processes, project work is notable for relatively high values with the knowledge processes *apply*, *transfer* and *assess*. The following instructional methods in the heat map are also noteworthy: the models method due to its relatively high values in the knowledge process *apply*, programmed instruction due to its relatively high values in the knowledge processes *build*, *process* and *apply*, learning stations due to its relatively high values in the knowledge process of *process*, and finally presentation and the experiment method due to their relatively high values in the knowledge process *build.* The following instructional methods had relatively low values in all of the knowledge processes (< 3.00): learning by teaching, case study, the jigsaw method, concept mapping and the Leittext method. Web quest, reciprocal teaching and the portfolio method were rated as relatively poor (< 2.50) in all of the knowledge processes.

Reciprocal teaching and programmed instruction are two instructional methods that showed high effect sizes in the Hattie study, as evidenced by a large number of studies in different subjects, but not in computer science. On the other hand, the assessments of computer science teachers show a very different picture: While computer science teachers assessed program instruction well, especially for the knowledge processes of *build* and of *process*, reciprocal teaching was rated poor in all knowledge processes. In order to verify and validate the assessments by the computer science teachers, reciprocal teaching and programmed instruction must be tested in authentic teaching scenarios, which is the objective of this study.

1.1. Reciprocal teaching

Reciprocal teaching is a dialogical instructional method between teachers and learners which serves as a tool in understanding the meaning of texts. This instructional method is attributable to Palinscar and Brown (1984).

Reciprocal teaching is based on the principle of learning by teaching. Reciprocal teaching is compatible with the assumptions of the cognitive learning theory in which learning takes place through the construction and conversion of cognitive structures. The role of the learner is to actively process information, to organize and to reorganize it. Depending on the learning progress, the learning path is controlled by the teacher or self-controlled by the learners themselves (see Zendler, 2018).

Rosenshine and Meister (1994) have presented a first summary of studies on reciprocal teaching. In their study, a co-operative variant of reciprocal teaching was superior to conventional methods for reading, using standardized tests ($d = 0.32$) and self-constructed tests ($d = 0.88$) (Rosenshine & Meister, 1994). In the two meta-analyses of Hattie (2009), reciprocal teaching has an effect size of $d = 0.74$, and was among the top ten instructional methods. The use of reciprocal teaching is mainly positive concerning reading comprehension and in students' self-efficacy. Similar effects have been demonstrated for natural sciences (Palincsar, Brown, & Campione, 1996; Schneeberger, 2009).

The basic structure of this instructional method is as follows: The instructor and each student from a small group take turns leading the discussion on a particular section of text in a joint attempt to understand it. This involves moving through four steps in an iterative