Journal of Mathematical Behavior 36 (2014) 20–32

Contents lists available at ScienceDirect

The Journal of Mathematical Behavior

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

Learning mathematics through algorithmic and creative reasoning

Bert Jonsson[a,*], Mathias Norqvist[b,d], Yvonne Liljekvist[e,f], Johan Lithner[c,d]

[a] *Department of Psychology, Umeå University, Sweden*
[b] *Department of Mathematics and Mathematical Statistics, Umeå University, Sweden*
[c] *Department of Science and Mathematics Education, Umeå University, Sweden*
[d] *Umeå Mathematics Education Research Centre, Umeå University, Sweden*
[e] *Department of Mathematics and Computer Science, Karlstad University, Sweden*
[f] *The Centre of Science, Mathematics and Engineering Education Research, Karlstad University, Sweden*

ARTICLE INFO

Keywords:
Mathematical reasoning
Reasoning
Cognitive proficiency
Memory retrieval

ABSTRACT

There are extensive concerns pertaining to the idea that students do not develop sufficient mathematical competence. This problem is at least partially related to the teaching of procedure-based learning. Although better teaching methods are proposed, there are very limited research insights as to why some methods work better than others, and the conditions under which these methods are applied. The present paper evaluates a model based on students' own creation of knowledge, denoted creative mathematically founded reasoning (CMR), and compare this to a procedure-based model of teaching that is similar to what is commonly found in schools, denoted algorithmic reasoning (AR). In the present study, CMR was found to outperform AR. It was also found cognitive proficiency was significantly associated to test task performance. However the analysis also showed that the effect was more pronounced for the AR group.

1. Introduction

The overarching goal in the teaching of mathematics is to help students develop *mathematical competence*; that is the ability to understand, judge, do, and use mathematics across a variety of mathematical situations (Niss, 2007). Basic mathematical competencies include problem-solving abilities (how to solve tasks without knowing a solution method in advance), reasoning ability (the ability to justify choices and conclusions), and conceptual understanding (insights regarding the origin, motivation, meaning, and use of mathematics). In an experimental design the present study primarily addresses whether and how students can develop conceptual understanding *through* mathematical problem solving and mathematical reasoning by engaging in more creative activities than procedure-based learning using predefined algorithms (e.g., Haavold, 2011; Lithner, 2003, 2008). In addition, the mathematical task solving and reasoning are considered in relation to individual variation in cognitive proficiency. The present study is carried out in an experimental design and in that context it is important to point out that the proportion of studies that have been conducted pertaining to mathematics education, and that adopt experimental designs, is rare. During 2012, only 3% of papers published in leading mathematics education journals used experimental designs (Alcock, Gilmore, & Inglis, 2013).

* Corresponding author. Tel.: +46 706777612; fax: +46 7866695.
E-mail addresses: bert.jonsson@psy.umu.se, bert.jonsson@live.se (B. Jonsson).

http://dx.doi.org/10.1016/j.jmathb.2014.08.003

1.1. Learning in mathematics

Much time in mathematics classes is spent learning and rehearsing algorithms, which are supposed to provide students with a quick and reliable way to cope with many of the tasks ahead (Boesen et al., 2014; Hiebert, 2003). There are, however, doubts as to whether these algorithms actually give rise to any deeper understanding of the principles of mathematics, or whether the extensive use of algorithms is counterproductive (Hiebert, 2003). The notion of an *algorithm* includes all pre-specified procedures, that is, finite sequences of executable instructions that allow one to solve a given set of tasks (Brousseau, 1997). The importance of an algorithm is that it can be determined in advance, and the execution of an algorithm is associated with high reliability and speed, which is the strength of using an algorithm when the purpose of a task is only to produce an answer to a particular problem. In many cases, using an algorithm is appropriate; it saves time and prevents miscalculations. In this way, using algorithms provides students with opportunities to solve tasks simply by reusing the procedure that a particular algorithm stands for. However, the use of algorithmic reasoning is, in itself, not an indication of one's conceptual understanding of mathematics (Haavold, 2011). In addition, the reason why an algorithm is regarded as efficient in solving a task (but not for learning) is that it is designed to avoid meaning (Brousseau, 1997). Algorithms are often presented within a classroom context. A typical situation arises whereby the teacher or textbook provides students with a set of mathematical tasks and a template solution method (algorithm); this is then followed by massive repetition of the algorithm, leading to an un-reflected use of the same algorithm (Boesen et al., 2014; Lithner, 2008). The tasks can, therefore, be solved according to the provided template without any conceptual understanding of the actual problem. Sufficient amounts of exposure to the algorithm may also lead to rote learning (the process of learning something by repeating it until it becomes memorized, rather than learning something by understanding the meaning of it; Oxford Advanced Learner's Dictionary); this means that the algorithm can be recalled in its original form without any conceptual understanding of it.

In the present study, we define using memorized or well-rehearsed procedures (such as algorithms) without reflecting on their meaning as algorithmic learning. An important note is that using well-rehearsed procedures or engaging in rote learning can be an efficient way to learn facts such as multiplication tables (Caron, 2007). In a similar way, using algorithms can reduce the cognitive demands of complicated calculations (Haavold, 2011), and thus also the cognitive load on our working memory (Raghubar, Barnes, & Hecht, 2010).

The components and capacity of working memory refer to the ability to process and store information simultaneously (e.g., Baddeley, 2010). Students could, therefore, be aided by using algorithms that reduce the cognitive load, thereby freeing resources for more advanced problem solving to occur (Merriënboer & Sweller, 2005). However, if all or most learning is done using routine procedures, it can lead to algorithmic reasoning that is based on superficial features of the algorithm, and not on the intrinsic properties of the tasks at hand (Hiebert, 2003; Lithner, 2003); as a result, there is a risk that mathematical competences are not well developed. In spite of being efficient in the short term – in the sense that students can quickly solve new practice tasks, as long as there are templates to use and memorize – there are many studies showing that procedure based teaching models fail to enhance students' long-term development in basic mathematical competencies (see Hiebert, 2003 for an overview). Several other concepts are used in the literature to capture similar phenomena related to the dichotomy between superficial versus deep/true/conceptual mathematical learning. In the seminal book "*Conceptual Knowledge and Procedural Knowledge*" (Hiebert, 1986), Hiebert and Lefevre defined conceptual knowledge as a form of knowledge that is rich in informational relationships, and linked in a network where the connections within the network are as important as the discrete pieces of information themselves. Procedural knowledge was defined in terms of a person's ability to become familiar with the symbols and conventions of mathematics, while having access to the rules or procedures required to solve mathematical problems (Hiebert & Lefevre, 1986). However, Star (2005) argued that conceptual knowledge does not necessarily need to have a rich informational relationship. For example, a child's conceptual knowledge can be less sophisticated and differently connected than that of an adult, but it is still regarded as conceptual knowledge. In a study by Rittle-Johnson and Alibali (1999) it was argued that 'conceptual instructions' (children were told the underlying principle behind the problem solution) to greater extent than procedural based instructions (being taught the procedure) influence conceptual understanding. However the results also indicated that the relationship is bidirectional, se also Rittle-Johnson, Siegler, and Alibali (2001) and Schneider, Rittle-Johnson, and Star (2011). In the present study, no 'conceptual instructions' such as the underlying principles are provided. The key issue in the present study is allowing for mathematical "struggle" in adidactical situations (no teacher support) with tasks that are designed to facilitate students' *own construction* of solutions.

1.2. The importance of a productive "struggle"

In order for students to obtain desirable learning outcomes, "the students need to be engaged in activities where they have to 'struggle' (in a productive sense of that word) with important mathematics" (Niss, 2007, p. 1304). At the same time, a delicate balance must be maintained in order to prevent these struggles from becoming obstacles, rather than promoters of learning. Hiebert and Grouws (2007) concluded in a mathematics education research review that this 'struggle' is necessary in order to enhance students' development of conceptual understanding of the principles involved in mathematics. Still, little is known about how this idea of a 'struggle' translates into specific activities that are useful in the teaching of the subject, and in what way these activities are linked to learning outcomes (Niss, 2007). However, support for the argumentation of learning outcomes can be found in the field of memory research, where several studies have shown that more 'struggle' in terms of more effortful retrieval is effective for later performances on subsequent tasks (e.g., Pyc & Rawson, 2009); these are