Cognition 149 (2016) 40–55

Contents lists available at ScienceDirect

Cognition

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

Non-formal mechanisms in mathematical cognitive development: The case of arithmetic

David W. Braithwaite [a,*], Robert L. Goldstone [b], Han L.J. van der Maas [c], David H. Landy [b]

[a] *Carnegie Mellon University, Pittsburgh, PA, United States*
[b] *Indiana University, Bloomington, Indiana, United States*
[c] *University of Amsterdam, Amsterdam, Netherlands*

ARTICLE INFO

Article history:
Received 6 August 2014
Revised 27 August 2015
Accepted 5 January 2016
Available online 12 January 2016

Keywords:
Mathematical cognitive development
Concrete to abstract shift
Arithmetic
Syntax
Perception
Mathematics education

ABSTRACT

The idea that cognitive development involves a shift towards abstraction has a long history in psychology. One incarnation of this idea holds that development in the domain of mathematics involves a shift from non-formal mechanisms to formal rules and axioms. Contrary to this view, the present study provides evidence that reliance on non-formal mechanisms may actually increase with age. Participants – Dutch primary school children – evaluated three-term arithmetic expressions in which violation of formally correct order of evaluation led to errors, termed foil errors. Participants solved the problems as part of their regular mathematics practice through an online study platform, and data were collected from over 50,000 children representing approximately 10% of all primary schools in the Netherlands, suggesting that the results have high external validity. Foil errors were more common for problems in which formally lower-priority sub-expressions were spaced close together, and also for problems in which such sub-expressions were relatively easy to calculate. We interpret these effects as resulting from reliance on two non-formal mechanisms, perceptual grouping and opportunistic selection, to determine order of evaluation. Critically, these effects reliably increased with participants' grade level, suggesting that these mechanisms are not phased out but actually become more important over development, even when they cause systematic violations of formal rules. This conclusion presents a challenge for the shift towards abstraction view as a description of cognitive development in arithmetic. Implications of this result for educational practice are discussed.

1. Introduction

The idea that cognitive development involves a shift towards abstraction has a long history in psychology (Gentner & Toupin, 1986; Gentner, 1988, 2003; Keil & Batterman, 1984; Keil, 1989; Piaget, 1952; Rattermann & Gentner, 1998; Vygotsky, 1962). This shift supposedly involves decreasing reliance on perceptual features and details of context, and increasing reliance on abstract features and context-free rules. In academic disciplines such as mathematics and physics, the development of expertise as a result of education and experience has also been described in terms of a shift towards abstraction (Chi, Feltovich, & Glaser, 1981; Chi & VanLehn, 2012; De Lima & Tall, 2008; Novick, 1988; Tall, 1995, 2008). However, some researchers have challenged the notion of a shift towards abstraction on both theoretical (Keil, Smith, Simons, & Levin, 1998) and empirical (Bulloch & Opfer, 2009) grounds, or even proposed that a shift in the opposite direction may occur (Simons & Keil, 1995; Varma & Schwartz, 2011).

The present study provides evidence that in the domain of symbolic arithmetic, the influence on performance of formally extraneous perceptual and contextual details increases with age and experience, suggesting that development in this domain cannot be fully characterized in terms of a shift towards abstraction. Arithmetic is an attractive domain for investigating this issue for at least two reasons. First, there exist explicit formal rules constraining correct performance in arithmetic, so it is natural to suppose that arithmetic competence consists precisely in following these rules, and that the development of such competence involves a shift towards such formal rule-governed behavior. Thus, one might expect arithmetic to be a likely domain for showing a developmental shift towards abstraction. Secondarily, arithmetic is of immense practical importance, due to its direct utility in a wide range of

---

* Corresponding author at: Department of Psychology, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, United States.
E-mail address: baixiwei@gmail.com (D.W. Braithwaite).

http://dx.doi.org/10.1016/j.cognition.2016.01.004
0010-0277/© 2016 Elsevier B.V. All rights reserved.

other domains as well as its foundational role in higher mathematics. Understanding the nature of learning and development in arithmetic has the potential to inform instructional design and thereby improve educational outcomes.

We focus more specifically on the evaluation of complex arithmetic expressions – that is, expressions involving multiple operations. Intuitively, such expressions could be evaluated by selecting and evaluating simple sub-expressions until a single value is reached. For example, faced with the complex expression "1+3×2," one might first evaluate "3×2" as "6," then "1+6" as "7." The processes underlying evaluation of simple arithmetic expressions (e.g. "3×2" and "1+6") are well-understood, and include counting-based strategies, calculation algorithms, and retrieval from memory (Brissiaud & Sander, 2010; Miller, Perlmutter, & Keating, 1984; Moore & Ashcraft, 2015; Shrager & Siegler, 1998; Siegler & Stern, 1998). Less well-understood are the mechanisms by which simple sub-expressions are selected and prioritized for evaluation in the first place. For instance, in the above example, how does one decide to begin by evaluating "3×2" rather than "1+3?" We first describe three mechanisms that could support such selections: syntactic parsing, perceptual grouping, and opportunistic selection. We then discuss the possible roles of these mechanisms over the course of learning and development.

1.1. Mechanisms

1.1.1. Syntactic parsing

In syntactic parsing, evaluation of complex expressions is preceded and guided by their syntactic structure, which is determined according to formal rules of syntax. For example, applying rules of operator precedence to the expression "2+7×5" would allow one to identify "7×5" (but not "2+7") as a syntactic phrase within the larger expression. This simpler sub-expression could then be evaluated directly via retrieval from memory. As another example, applying the rule for left-to-right evaluation among operators of equal precedence to the expression "25–13–3," one would identify "25–13" (but not "13–3") as a syntactic phrase, which could then be evaluated.

Consistent with such a mechanism, adults trained in arithmetic and algebra are sensitive to syntactic structure (Jansen, Marriott, & Yelland, 2003; Schneider, Maruyama, Dehaene, & Sigman, 2012). During scanning of complex arithmetic expressions, adults' gaze trajectories quickly move to the sub-expressions deepest in the syntactic hierarchy and thereafter proceed upwards along the syntactic tree (Schneider et al., 2012), suggesting that syntactic structure is extracted quickly and automatically. Further, after viewing complex algebraic expressions, sub-strings that constituted syntactic phrases within the expressions are recalled more easily than sub-strings that did not constitute syntactic phrases (Jansen et al., 2003). Apparently, syntactic structure influences encoding and subsequent recall of algebraic expressions. Several computational models assume that human processing of algebraic expressions begins with, and is subsequently guided by, syntactic parsing (Anderson, 2005, 2009; Jansen, Marriott, & Yelland, 2007).

1.1.2. Perceptual grouping

In perceptual grouping, as in syntax-based processing, evaluation of complex expressions begins with identification of simpler sub-expressions, but perceptual constraints rather than formal rules determine which symbols are grouped together to form sub-expressions (Landy, Allen, & Zednik, 2014). There is strong evidence that at least one such constraint – a tendency to group together symbols that are physically close to each other, consistent with the Gestalt principle of proximity (Wertheimer, 1938) – does indeed influence processing of symbolic expressions in arithmetic and algebra (Jiang, Cooper, & Alibali, 2014; Kirshner, 1989; Landy & Goldstone, 2007b, 2010). For example, violations of operator precedence rules are more common with expressions in which the operands surrounding a lower-precedence operator are more narrowly spaced than those surrounding a higher-precedence operator, as in "2+7 × 5" (Landy & Goldstone, 2010). Apparently, the perceptual constraint that closely-spaced symbols are more likely to be perceived as groups can sometimes override the formal rules that determine syntactic structure.

Importantly, while perceptual constraints may cause violations of formal rules, perceptual processing does not in general preclude formally correct performance. The reason is that perceptual constraints are flexible, and may evolve over time to come into closer alignment with such formal rules (Goldstone, Landy, & Brunel, 2011; Goldstone, Landy, & Son, 2010). For example, there is some evidence that adults experienced with arithmetic perceive higher-precedence operator symbols (e.g. ×, ÷) as more visually salient than lower precedence ones (e.g. +, –; Landy, Jones, & Goldstone, 2008). These differences in salience could lead to preferential grouping of operand symbols surrounding higher-precedence operators, resulting in formally correct order of evaluation. More generally, practice with symbol systems could lead to the development of automatic perceptual routines that effectively implement syntactic rules, without representing such rules explicitly. Consistent with this possibility, in a recent neuroimaging study, participants viewing arithmetic expressions of varying syntactic complexity showed effects of syntactic complexity on BOLD response in brain areas relating to early visual processing, while such effects were not found in areas associated with language (Maruyama, Pallier, Jobert, Sigman, & Dehaene, 2012; see also Friedrich & Friederici, 2009; Monti, Parsons, & Osherson, 2012; but see Scheepers et al., 2011).

1.1.3. Opportunistic selection

Opportunistic selection refers to prioritizing for evaluation sub-expressions which are relatively easy to evaluate. For example, in evaluating "25 + 13 – 3" one might begin by evaluating the subtraction ("13 – 3") because it is easier to evaluate than the addition ("25 + 13"). Opportunistic selection yields a formally correct answer in this case (13 – 3 = 10, 25 + 10 = 35), but not in all cases. In the similar problem "25 – 13 – 3," evaluating "13 – 3" first violates the rule of left-to-right order of operations and so yields an error.

There is some evidence that opportunistic selection does occur, and can even override formal rules of arithmetic. Linchevski and Livneh (1999; Herscovics & Linchevski, 1994) found that students frequently commit errors like that just mentioned, justifying their procedures by appeals to convenience (e.g. "when you do [operation] first, it becomes much easier"). However, these findings are not entirely conclusive for present purposes because the errors in question may have resulted from random slips, with convenience mentioned only as a post hoc rationalization. The present study addressed this possibility by comparing rates of order-of-operations errors between similar problems in which the (formally) low-priority sub-expressions either were or were not particularly easy to evaluate. Higher error rates for the former type of problem would provide strong evidence that opportunistic selection does occur and can override formal syntactic rules.

An important difference between opportunistic selection and syntactic parsing relates to the types of information to which they are sensitive. Because the ease of evaluating sub-expressions depends on the specific numbers involved, opportunistic selection is necessarily sensitive to the values of these numbers. Syntactic parsing, by contrast, depends only on syntactic structure, not on content, and should therefore be insensitive to the number values involved in an expression. This insensitivity to number values is