Using Alternative Multiplication Algorithms to 'Offload' Cognition

Dan Jazby
University of Melbourne
<dan.jazby@unimelb.edu.au>

Cath Pearn
University of Melbourne
<cpearn@unimelb.edu.au>

When viewed through a lens of embedded cognition, algorithms may enable aspects of the cognitive work of multi-digit multiplication to be 'offloaded' to the environmental structure created by an algorithm. This study analyses four multiplication algorithms by viewing different algorithms as enabling cognitive work to be distributed across environmental and mental resources to varying degrees. This produces a plausible framework which could allow further analysis designed to guide the pedagogical use of alternative algorithms.

Many students struggle to learn the traditional written algorithms introduced in primary school (Pearn, 2009). Heirdsfield (2004) states: "vertical algorithms dictate a rigid procedure, and do not lend themselves to encouraging students to manipulate numbers flexibly" (p.8). However, Westwood (2000) suggests that:

> children should have no problem mastering these procedures [algorithms] if they are linked as closely as possible with the more informal methods … that are typically used by children … difficulties arise if the processes are taught without reference to children's prior learning or way of recording (p. 47).

While many teacher publications (such as Randolph & Sherman, 2001) have advocated that alternative algorithms may be beneficial for students, there is less written regarding exactly how and why an alternative algorithm may help students (Carroll & Porter, 1998). This paper uses a theoretical framework based on embedded cognition to try to assess how different algorithms may impact on the cognitive demands of multi-digit multiplication. This analysis may provide some explanation as to why some students develop a preference for an alternative algorithm, enable such preferences to be used to diagnose student understanding and suggest a framework which could demonstrate when teaching an alternative algorithm may or may not be justified. This paper focuses on alternative multiplication algorithms that can be used instead of 'traditional' long multiplication for multi-digit problems.

Student use of Alternative Algorithms for Multiplication

'Alternative' algorithms are defined in contrast to the 'traditional' algorithm which is sometimes referred to as long multiplication (West, 2011). Many articles which explain alternatives to long multiplication are aimed at helping teachers learn how to use these alternatives so that:

> Students' individual needs and styles are the focus of lessons on alternative algorithms. Using these options, students develop their own understanding of, and skills in, arithmetic operations, enhancing their decision-making and critical thinking skills (Randolph & Sherman, 2001, p. 484).

Advocates of alternative algorithms tend to argue that they may be of benefit to students because knowing alternative methods improves general understanding. It has been noted that some students develop a preference for some algorithms. Lattice multiplication is one such alternative algorithm – an example is presented in Figure 1 and explained in detail in Results. When Carroll and Porter (1998) described the method they noted that, "although the reasons are not obvious to us, this method has proved to be very popular with students" (p. 111). They also note that low-achieving students tend to like this

2015. In M. Marshman, V. Geiger, & A. Bennison (Eds.). *Mathematics education in the margins* (Proceedings of the 38th annual conference of the Mathematics Education Research Group of Australasia), pp. 309–316. Sunshine Coast: MERGA.

method, perhaps because of its structure. If low achieving students prefer an algorithm such as the lattice method, then perhaps there is something about either the lattice method or student understanding which explains this preference. Also, if students' general understanding of mathematics is enhanced by use of alternative algorithms, some investigation of the mechanisms which underpin this seem warranted. This study uses a framework of embedded cognition to attempt to make sense of students' use of alternative algorithms. The analysis seeks to provide a lens which enables alternative algorithm use to be interrogated in more detail. If alternative algorithms can cater to students' individual needs, as Randolph and Sherman (2001) state, then one should ask which student needs an alternative algorithms addresses. Embedded cognition, described in the next section, provides a method for analysing how cognitive work can be distributed across mental and environmental resources. It is hypothesised that different distributions of this cognitive work by different algorithms may change the mental demands placed on students as they solve a problem.

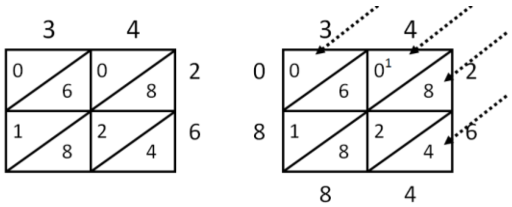

Figure 1. 34 × 26 using the lattice algorithm

An Embedded View of Cognition

Embedded cognition posits that cognition is embedded in an environment. This means that people use environmental structures to 'offload' cognition so that cognition is distributed across both environmental and mental resources (Kirlik, 2007). It is theorised that the environment provides a direct model of a problem and that a person can use and create environmental structures which offload cognition, using a combination of mental and environmental resources to perform tasks which, traditionally, have been seen to occur purely mentally.

Kirlik's (2007) model of embedded cognition has been developed in the research field of Human Factors and Ergonomics. It draws on situated theories of learning, such as those developed by Greeno (1998), to describe the interaction between workers and their task environments – mainly in Kirlik's main field of interest, aviation. While situated learning theorists often focus on how learning is situated within complex social entities (Greeno, 1998), Kirlik's focus examines how cognition is situated within a physical task environment. This paper seeks to assess whether the view that cognition can be 'offloaded' to physical environmental structures could provide a productive lens to understand how children develop preferences for particular algorithms. Kirlik's view of embedded cognition would suggest that, when children use different algorithms, they may be able to manipulate or utilise environmental structures, in such a way as to 'offload' cognition and reduce demands made on mental resources. Information can be stored and updated in the environment. A calculation can be separated into simpler calculations. Each of the simple calculations can be performed mentally, then results can be stored in the environment and combined with the results of other calculations at a later time, so that running totals of simple calculations do not have to be maintained mentally. By writing or typing information into an algorithm's predetermined structure, not only can information be stored, but cognitive processes can be ordered and coordinated. Just as Kirlik (2007) has argued that professionals in aviation are able to improve their task performance by effectively distributing 'cognitive work' across both mental and environmental resources, algorithms may facilitate more effective computation via a similar process of offloading.

What 'Cognitive Work' do Algorithms Perform?

For the purposes of this study, multiplication algorithms are being viewed as cognitive aids which enable a multiplication problem to be broken up into a series of less cognitively demanding subroutines. The authors distinguish between two phases in multiplication algorithms – a multiplication phase and an addition phase. For example, when calculating 34 × 26, most algorithms enable 34 × 26 to be calculated by separating 34 × 26 into a series of simpler calculations (e.g. 6 × 4, 6 × 3, 2 × 4, 2 × 3). Kilian, Cahill, Ryan, Sutherland, and Taccetta (1980) found that 32% of student errors occurred using the traditional algorithm related to miscalculation in this multiplication stage.

During these calculations, the place value position of the numbers being multiplied may be suspended. When calculating 34 × 26, 30 must be multiplied by 6. Algorithms which suspend place value enable this calculation to be carried out as 3 × 6. If an algorithm suspends place value, then successful use of the algorithm requires some cognitive work which recognises that this 3 × 6 is in fact 3 tens × 6 (and is therefore 18 tens rather than 18). Kilian et al. (1980) found that 18% of student errors involving the traditional algorithm related to place value.

As algorithms break 34 × 26 into a series of simpler calculations another cognitive demand of using the algorithm entails ensuring that all of these simpler calculations are performed. The term 'completeness' is used to refer to the cognitive task of ensuring that all relevant simpler calculations have been performed. Kilian et al. (1980) found that 7% of students' errors using the traditional algorithm involved missing one of these simpler calculations.

In the addition phase, the products of the simpler calculations performed in the multiplication phase must be totalled correctly. This is often performed using a traditional addition algorithm which adds like place value parts. Kilian et al. (1980) found that, when the traditional algorithm was used, errors in addition calculations were low (9%), but 'carrying' mistakes accounted for approximately a quarter of all errors (24%). One of the algorithms analysed does not use a traditional addition algorithm. Instead, the products of the simpler multiplication steps are ordered without explicit reference to place value. See the line algorithm in Results for a detailed description of this kind of addition phase.

Method

West (2011) provided a description of nine alternative multiplication algorithms. In this paper we will discuss three algorithms. These three algorithms (Line, Lattice and Area multiplication algorithms) were selected because they make use of structures in their physical layout. These alternative algorithms are compared to the partial product algorithm commonly taught in Australian schools. Each algorithm will be compared to each other when used to solve the same problem. A 2-digit multiplied by 2-digit number problem was selected (34 × 26) with the numbers being chosen using a random number generator.

During the multiplication stage of each algorithm, initial analysis will involve determining which simpler calculations this multiplication problem – 34 × 26 – is broken into. The next analysis involves identifying whether any of these calculations need to be performed mentally or can be offloaded to the structure of the algorithm.

Place value in the multiplicative stage of each algorithm has been categorised as either ‘suspended’ or ‘not suspended’. If an algorithm suspends place value, then some cognitive work must be performed in order to reinstate place value position after simplified calculations have occurred. Algorithms which suspend place value have then been categorised according to whether reinstating place value position needs to be performed mentally or can be offloaded to the structure of the algorithm.

Completeness has been analysed in terms of whether this needs to be maintained mentally or whether this cognitive task can be offloaded to the structure of the algorithm.

There are two classifications that have been used in relation to the addition phase of each algorithm. Addition phases either employ the traditional addition algorithm or they employ a non-traditional addition algorithm which does not add products in place value parts (described in the next section).

Results

In the following section, each alternative algorithm is analysed separately before these separate analyses are compared in Tables 1 and 2.

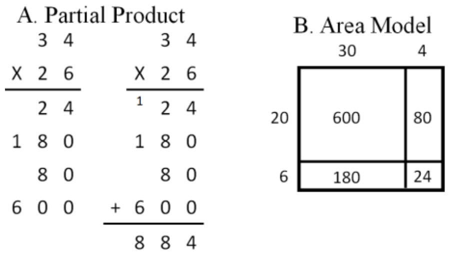

Figure 2. 34 × 26 using the partial product and area model algorithms

Partial Product Algorithm

Part A of Figure 2 shows the partial product algorithm broken down into two phases – the multiplicative phase and the additive phase. In the multiplicative phase, four single-digit numbers must be multiplied separately. These calculations must be performed mentally. Place value is suspended when these calculations are performed. When recording each partial product, place value must be tracked, mentally, by the student. In particular, when ‘3’ is multiplied by ‘2’, students must mentally track that this is really 3 tens multiplied by 2 tens, and thus, the product is 6 hundreds. A significant amount of mental cognition must be employed to track that the product of digits in the second column need to be recorded in the third column. Hence, this algorithm has been classified as requiring mental cognition to maintain place value. Completeness is also only partially helped by the structure of the algorithm. The common approach is to start with the right most digit of the bottom row number and multiply this by the digits of the number on the top row starting from left to right. However, there is no element of the structure of this algorithm which enables students to visually recognise if they have missed a step. Hence, this aspect of the cognitive work of using the algorithm has been coded as requiring mental cognition.

In the addition phase, the column structure of the algorithm allows each place value part to be totalled separately. When the second column results in a total greater than 9, ‘carrying’ is used. Then students must recognise the need to move into the next column, so that 18 tens is renamed as 1 hundred and 8 tens.

Area Model Algorithm

Part B of Figure 2 shows the area model algorithm. An area model is used to model the problem and the resulting rectangle is partitioned into four separate areas. Place value parts are used to partition the rectangle so that the side of the rectangle which is 34 long is separated into two segments which are 30 and 4 long respectively. The area of each of the four partitions is arrived at by multiplying 4 × 6, 6 × 30, 20 × 4 and 20 × 30. While this entails multiplication of more than a single-digit number, the 2-digit numbers do not contain a non-zero digit in the ones-digit position, which reduces the difficulty of the calculation. This enables calculations to be performed mentally. When this algorithm is used, place-value positions in the multiplicative phase are maintained throughout. Furthermore, the structure of algorithm provides a visual representation of the magnitude of the products of multiplication – the 30 × 20 partition looks considerably bigger than the 20 × 4 partition. As partial products are recorded in each of the four partitions of the original rectangle, completeness is offloaded to the structure of the algorithm. In the addition phase, a traditional addition algorithm is used which has the same processes as a partial product algorithm.

Line Algorithm

Figure 3 shows the line algorithm. The horizontal lines represent 34 as there are 3 lines grouped at the top and 4 lines grouped at the bottom. The vertical lines represent 26 (2 lines to the left and 6 lines to the right). Diagonal lines (dotted in Figure 3) are used to create three areas. The number of line intersections in each area are then counted and totalled. In the top left region, there are 6 line intersections, 26 intersections in the middle region and 24 in the lower right region.

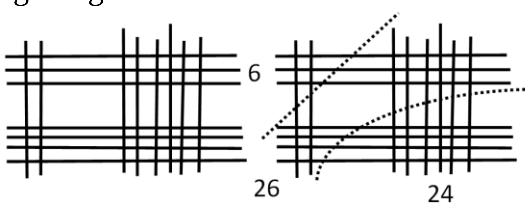

Figure 3. 34 × 26 using the line algorithm

This algorithm allows the problem to be solved without using mental multiplication calculations. A student could use a 'count all' strategy to total the number of line intersections. Hence, the structure of the algorithm enables 34 to be multiplied by 26 without having to mentally perform any multiplicative calculations. Place value is suspended and does not need to be tracked mentally. Completeness is supported by the structure as long as all line intersections are counted.

In the addition phase, the three products (6, 26 and 24) need to be combined. Starting with the right-most total (24), digits not in the right-most position are added to the number on the left. Thus, the '2' digit in 24 is added to 26 to get 28, the '2' digit in 28 is added to 6 to get 8. This produces the number 884 as the final product. In this phase as well, the student does need to keep track of place value parts because of the structure of the algorithm. If place value is maintained in this algorithm, then the three products derived from this algorithm are 600, 260 and 24, and the addition phase of the algorithm combines these. However, the algorithm enables these products to be combined without having to mentally reinstate place value.

The Lattice Algorithm

Figure 1 shows the multiplication and addition phases of the lattice algorithm. Like the partition product algorithm, four pairs of single-digit numbers are multiplied. The products of these calculations are recorded differently. When 2 is multiplied by 6 it is recorded in the grid where the tens-digit is recorded above the diagonal line and the ones-digit is recorded below (e.g. 2/4 for the total twenty-four). If there a product is less than 10 (e.g. in the case of 2 × 4), a zero is recorded above the diagonal line (e.g. 0/8).

Calculations during the multiplication phase are performed mentally and place value is suspended. The lattice structure of this algorithm maintains the place value position of these digits so that they do not have to be maintained mentally. This structure ensures completeness without requiring any mental effort on the part of the student as all parts of the grid need to be filled.

In the addition phase, the dotted arrows on Figure 1 indicate which numbers need to be added together. Following the diagonal lines of the lattice, there are four ‘diagonals’ that need to be totalled. When the second diagonal results in a total over 9 (8 + 2 + 8 = 18), the tens-digit is ‘carried’ to the next diagonal. Each diagonal maintains place value in a similar fashion to columns in standard addition algorithms, although zero digits are not needed to communicate place value position.

Table 1:
Calculations and place value in the multiplication phase of each algorithm

	Calculation		Place Value (PV)		
			Suspend	Maintain PV position of digits	
Algorithm	Mental	Structure	PV	Mental	Structure
Partial Product	Y	N	Y	Y	N
Area Model	Y	N	N	N/A	N/A
Line	N	Y	Y	N	Y
Lattice	Y	N	Y	N	Y

Comparing Algorithms

Tables 1 and 2 compare the four algorithms analysed. Each of the four multiplication algorithms analysed enable different aspects of the cognitive work of solving a multi-digit multiplication problem to be distributed differently between the environment and an individual’s mental resources. Multiplication calculations must be performed mentally with all algorithms except the line algorithm. Only the partial product algorithm requires students to maintain place value mentally – the structure of both the line and lattice algorithms allows this cognitive task to be offloaded while the area model does not suspend place value.

Table 2 shows that all three of the alternative algorithms enabled completeness to be offloaded to the environmental structure of each algorithm and the line algorithm used a non-traditional approach to addition which enabled successful addition to take place without mentally maintaining place value position of digits.

Table 2:
Ensuring all multiplication calculations are made and summary of the addition phase

	Completeness		Addition phase	
Algorithm	Mental	Structure	Traditional PV part addition algorithm	Addition algorithm without PV
Partial Product	Y	N	Y	N
Area Model	N	Y	Y	N
Line	N	Y	N	Y
Lattice	N	Y	Y	N

Discussion

Rather than catering to "students' individual needs and styles" (Randolph & Sherman, 2001) alternative algorithms – in this analysis – are posited to have a specific impact on students' mental workload. The analysis presented suggests that this view is theoretically plausible and could be used to guide further investigation. The lattice algorithm, for example, may enable successful calculation of multi-digit multiplication without the need to mentally attend to the place value position of component calculations and completeness. While Carroll and Porter (1998) could not identify any obvious reason why 'low achieving' students would develop a preference for the lattice algorithm, viewing students' use of algorithms as embedded cognition provides a theoretical explanation: if the lattice algorithm offloads aspects of the cognitive work of solving the problem into an environmental structure, students can successfully perform multi-digit multiplication with decreased cognitive load.

If the use of algorithms – like the work of professionals in working environments (Kirlik, 2007) – is embedded, and students can use different algorithms to distribute cognitive work across both mental and environmental resources differently, then one may ask whether all algorithms are created equal; should all algorithms be taught to students; and what would warrant the use of a particular algorithm? Viewing algorithms through the lens of embedded cognition generates hypotheses which can be tested. If 'low achieving' students develop a preference for the lattice algorithm (Carroll & Porter, 1998), an embedded analysis of the algorithm would suggest that these students may also have less understanding of place value than students who use partial product algorithms. Students who prefer the line algorithm may also lack effective mental strategies for multiplication. Future research may be able to test whether such preferences for alternative algorithms correlate with specific mathematical difficulties of students.

Through the lens of embedded cognition, the traditional partial product algorithm enables the least amount of offloading of cognitive tasks to the environment. Completeness and place value must be maintained mentally with limited structural support. Kilian et al. (1980) found that 56% of students' errors using a traditional algorithm related to these procedural aspects (rather than calculation errors). While the partial product algorithm enables a multi-digit problem to be broken down into component calculations, there are many mental demands made when using the algorithm beyond calculations. Further research may be able to ascertain whether alternative algorithms could be used in a sequenced fashion, to enable the mental demands of the partial product algorithm to be approached gradually – in that the line algorithm could be used to introduce multi-digit multiplication algorithms (with a relatively small mental workload) before transition to the area model algorithm followed by the partial product algorithm. Theoretically, the analysis presented in this paper would suggest that this sequence may represent a viable scaffolding of the mental demands required to use the partial product algorithm. While further research is required to test such a sequence, theoretically it may help students avoid the procedural errors Kilian et al. documented with the partial product algorithm.

Conclusion

An embedded view of cognition has been applied to four multiplication algorithms to assess how the cognitive demands of solving 34 × 26 can be distributed across both environmental and mental resources. Results suggest that algorithms differ mainly in relation to how place value and ensuring all calculations are made ('completeness') are supported by algorithms' structures. This provides a plausible explanation as to why some students may develop a preference for particular algorithms as each algorithm requires different aspect of the cognitive work of solving 34 × 26 to be completed mentally. Thus, alternative algorithms may not be just a matter of individual style but may be used in specific ways to enable successful task performance. A traditional partial product algorithm offloaded the least amount of cognitive work to the environment and made the highest demands on mental resources of the algorithms analysed. Hence, further research – guided by a model of embedded cognition – may be able to identify how using alternative algorithms may address specific student errors relating to using traditional algorithms.

References

Carroll, W. M., & Porter, D. (1998). Alternative algorithms for whole-number operations. In L. J. Morrow & M. J. Kenney (Eds.), *The teaching and learning of algorithms in school mathematics* (pp. 106-114). Reston, VA: NCTM.

Greeno, J. G. (1998). The situativity of knowing, learning, and research. *American psychologist, 53*(1), 5-26.

Heirdsfield, A.M. (2004). Enhancing mental computation teaching and learning in Year 3. In *Proceedings Australian Association for Research in Education*, Melbourne.

Kilian, L., Cahill, E., Ryan, C., Sutherland, D., & Taccetta, D. (1980). Errors that are common in multiplication. *The Arithmetic Teacher*, 22-25.

Kirlik, A. (2007). Ecological resources for modeling interactive behavior and embedded cognition. In W. Gray (Ed.), *Integrated models of cognitive systems*. New York: Oxford University Press.

Pearn, C. (2009). Highlighting the similarities and differences of the mathematical knowledge and strategies of Year 4 students. In R. Hunter, B. Bicknell, & T. Burgess (Eds) *Crossing divides* (Proceedings of the 32nd annual conference of the Mathematics Education Research Group of Australasia, pp. 443-451)Wellington: MERGA.

Randolph, T. D., & Sherman, H. J. (2001). Alternative algorithms: Increasing options, reducing errors. *Teaching Children Mathematics, 7*(8), 480-484.

West, L. (2011). *An introduction to various multiplication strategies.* (Masters Thesis), University of Nebraska-Lincoln, Bellevue, NE.

Westwood, P. (2000). *Numeracy and learning difficulties: Approaches to teaching and assessment.* Camberwell: ACER.