Perceiving Structure in Mathematical Expressions

Anthony R. Jansen (tonyj@csse.monash.edu.au)
School of Computer Science and Software Engineering
Monash University, Clayton, Victoria, Australia

Kim Marriott (marriott@csse.monash.edu.au)
School of Computer Science and Software Engineering
Monash University, Clayton, Victoria, Australia

Greg W. Yelland (Greg.W.Yelland@sci.monash.edu.au)
Department of Psychology
Monash University, Clayton, Victoria, Australia

Abstract

Despite centuries of using mathematical notation, surprisingly little is known about how mathematicians perceive equations. The present experiment provides an initial step in understanding what sort of internal representation is used by experienced mathematicians. In particular, we examined if mathematical syntax plays a role in how mathematicians encode algebraic equations, or if just a simple memory strategy is used. Participants in the experiment performed a memory recognition task that required them to identify both well-formed (syntactically correct) and non-well-formed sub-expressions of equations. As hypothesised, performance was significantly better for well-formed sub-expressions, a result which suggests that mathematicians do indeed use an internal representation based on mathematical syntax to encode equations.

Introduction

Mathematical notation has evolved over hundreds of years. Like natural language, mathematical notation appears to have a well-defined syntax and semantics. It is clear that the expression

$$x(^2 \frac{+}{y8-}$$

is syntactically incorrect, while the equation

$$\frac{7^2 - 9}{5} = (3 - 2)^3$$

is syntactically well-formed but not true.

For decades now, phrase structure grammars have been used to understand how humans parse natural language sentences (for example, see Akmajian, Demers and Harnish, 1984). Such grammars have also been exploited in computer programming languages to process simple linearised mathematical expressions. However, unlike natural language (written or spoken), the syntax of mathematical notation is two-dimensional in nature. For example, the preceding equation relies on both vertical and horizontal adjacency relationships between the symbols to provide the meaning.

Given that mathematical notation has a well-defined syntax and a two-dimensional structure, it is natural to ask how humans comprehend mathematical equations and other notations with similar characteristics. We are especially interested in determining if humans parse mathematical expressions in a manner similar to the way in which they parse natural language. That is, do we assign grammatical structure to equations.

The notion that grammatical structure can be assigned to equations has some support from work on developing computer programs to understand mathematical notation. Since the pioneering work of Anderson (1977), almost all approaches have been grammar based, using some form of context-free attributed multiset grammar to specify the syntax. Attributes are used to capture geometric properties of the symbols, while the grammar works over multisets rather than sequences since there is no single way to sequence an equation. For further information, see the recent survey by Marriott, Meyer and Wittenburg (1998).

When processing an equation, the information needs to be stored in memory. Research has shown that humans have memory procedures for dealing with large amounts of information (Miller, 1956). They encode information into chunks that they can utilise within the limits of working memory. However, chunking is more efficient if it is guided by a structural principle. Research done by Johnson (1968, 1970) has shown that in the context of natural language, chunking is guided by syntax, with individual chunks conforming to grammatically defined units. Our aim is to determine if a similar process is true for mathematical expressions.

This experiment therefore has been designed to examine whether or not experienced users of mathematics use guided encoding when processing equations. That is, do mathematicians use some sort of internal representation that takes syntax into account, or do they use a less guided memory strategy to chunk a mathematical equation.

Surprisingly little work has been done that directly addresses this issue. The two-dimensional nature of mathematical notation seems to place it the same domain as diagrams. There has been much work done in the field of diagrammatic reasoning already (for example, see Glasgow, Narayanan and

Table 1: Example equations and sub-expressions.

Equation	Sub-Expression		
	Well-Formed	Non-Well-Formed	Incorrect
$\frac{9}{x(2y-5)} - 4x^3$	$(2y-5)$	$\overline{x(2}$	$x(4y+$
$x = 6yx - \frac{2x+2}{x}$	$2x+2$	$= 6yx-$	$\frac{6x+2}{y}$

Chandrasekaran, 1995), however this has concentrated on how diagrams are used in reasoning, rather than low-level perception of structure.

Our hypothesis is that experienced mathematicians do use some sort of internal representation to guide their encoding of equations, and that this representation is based on mathematical syntax. To test this hypothesis, we have set up a recognition task to determine if participants can more readily identify syntactically well-formed sub-expressions of an equation, as opposed to non-well-formed sub-expressions. If a simple memory strategy were being used, we would expect chunking in its most basic form, randomly splitting up the equations and providing no advantage for well-formed over non-well-formed sub-expressions. Such a result would mean that our hypothesis has no support. However, we expect that participants will respond significantly faster to the well-formed sub-expressions, indicating that mathematical syntax plays an important role in encoding equations.

Method

Participants Twenty-four participants successfully completed the experiment. All were staff members, graduate or undergraduate students from the Computer Science department, all competent mathematicians who were very familiar with algebra. All participants were volunteers between the ages of 18 and 35 years, with normal or corrected-to-normal vision. Data from an additional 13 participants were not included due to excessive error rates.[1]

Materials and Design One-hundred-and-twenty equations were constructed, all consisting of between twelve and fourteen characters. The equations contained at most one fraction and the variables were x and y, since these are most commonly used. For each equation, sub-expressions of three types were constructed.

a) A *well-formed sub-expression*, which is a component of its equation, and conveys the same meaning on its own that it conveys in the equation.

b) A *non-well-formed sub-expression*, which is also a component of its equation, but does not convey any coherent mathematical meaning on its own.

c) An *incorrect sub-expression*, which was not part of the original equation. It can be either well-formed or non-well-formed. These act as fillers.

Each of the sub-expressions contained between four and six characters (the average for well-formed sub-expressions was 4.89; for non-well-formed, 4.49; for incorrect, 4.72). See Table 1 for examples of equations and sub-expressions used. As the examples show, a variety of sub-expressions were used, some of which were bracketed, but most of which were not.

In order to present all three sub-expression types for each equation, but ensuring that participants were presented with each equation only once to avoid practice effects, three counterbalanced versions of the experiment were constructed. For each version, there were forty instances of each type of sub-expression. Two additional equations were constructed as practice items. The same practice items were used in each version. The items of each version were presented in a different pseudo-random order for each participant.

Procedure Participants were seated comfortably in an isolated booth. Items were displayed as black text on a white background on a 17" monitor at a resolution of 1024x768, controlled by an IBM compatible computer running a purpose designed computer program. The average width of the equations in pixels was 177 (range 99-220) with an average height of 47 (range 26-61). The average width of the sub-expressions in pixels was 73 (range 39-192) with an average height of 26 (range 16-54).

Participants were given a brief statement of instructions before the experiment began. Practice items preceded the experimental items, and the participants took approximately fifteen minutes to complete the task. Progress was self-paced, with participants pressing the space bar to initiate the presentation of each trial.

Each item was presented in the centre of the monitor in the following sequence. First, a simple algebra equation was shown to the participant for 2500ms. The equation then dis-

[1]For the results of a participant to be included in the final analysis, they were required to get at least 70% correct responses overall and at least 50% correct for any given sub-expression type. This resulted in there being twenty-four participants whose data was included, eight for each version of the experiment.

Table 2: Mean correct response times (ms) and error rates (%) as a function of sub-expression type.

Sub-Expression	RT(ms)		%Error	
Well-Formed	1147	(147)	13.1	(6.2)
Non-Well-Formed	1352	(228)	23.5	(8.8)
Incorrect	1429	(213)	32.8	(8.9)

appeared and the screen remained blank for 1000ms. Then the sub-expression was shown. The participant was required to decide whether the sub-expression was in that equation, responding via a timed selective button press. They pressed the green button, (the '/' key on the right side of the keyboard), to indicate that the sub-expression was part of the original equation, and the red button, (the 'Z' key on the left of the keyboard), to indicate that the sub-expression was not part of the original equation. Participants were instructed to respond as quickly as possible, while taking care not to make too many errors. The sub-expression remained on the screen until a response was made.

The response time recorded was the time between the sub-expression first appearing and the participant's response. After the response, the participant was given feedback. If the response was correct then the word "CORRECT" appeared on the screen. Otherwise, the word "INCORRECT" appeared on the screen. In both cases, the participant's response time in milliseconds also appeared on the screen.

Data Treatment Two measures were employed to reduce the unwanted effects of outlying data points. Absolute upper and lower cut-offs were applied to response latencies, such that any response longer than 2500ms or shorter than 500ms was excluded from the response time data analysis and designated as an error. Secondly, standard deviation cut-offs were applied, so that any response time lying more than two standard deviations above or below a participant's overall mean response time was truncated to the value of the cut-off point.

Three items, one from each condition of the experiment, were excluded from the analysis due to error rates in excess of 75%. As a result, the final analyses were over thirty-nine items per condition, not the original forty. Response time and error data were analysed by a series of analyses of variance (ANOVAs), over both participant and item data. Where both the subject-based and item-based analyses were significant they were combined in the *minF'* statistic to ensure the generalisability of results over both these domains (Clark, 1973). Results are reported only where effects are significant.

Results and Discussion

The mean correct response time and error rate for the three sub-expression types are summarised in Table 2, along with the corresponding standard errors (in parentheses). Planned comparisons of the data were conducted using two-way ANOVAs (versions × sub-expression), carried out separately over subject and item data.

The sub-expressions whose content was drawn from their corresponding equations (i.e., both well-formed and non-well-formed sub-expressions), were responded to more rapidly than incorrect sub-expressions (well-formed: $minF'(1, 40) = 57.34$, $p < .01$; non-well-formed: $F_1(1, 21) = 4.33$, $p < .05$, $F_2(1, 114) = 15.73$, $p < .01$, $minF'(1, 34) = 3.40$, $p = .074$).[2] This outcome is reflected in the error rate data also. Fewer errors were made relative to incorrect sub-expressions, on responses to both well-formed ($minF'(1, 105) = 45.15$, $p < .01$) and non-well-formed sub-expressions ($minF'(1, 59) = 5.60$, $p < .05$). While not unexpected, this pattern of outcomes is comforting for it indicates that both the experimental task and the participants are sensitive to the contents of algebraic equations.

More importantly, there is also a 205ms recognition advantage for sub-expressions that are well-formed components of their corresponding equation, over their non-well-formed counterparts ($minF'(1, 46) = 30.70$, $p < .01$). This recognition advantage holds for error rates also, with participants making significantly fewer errors on well-formed than non-well-formed sub-expressions ($minF'(1, 66) = 12.32$, $p < .01$). Clearly, the participants perceive the original equations in a way that allows faster and more accurate recognition of well-formed sub-expressions than non-well-formed sub-expressions.

The results of the experiment give support for the hypothesis stated in the introduction. That is, experienced mathematicians use an internal representation based on mathematical syntax to encode equations. This support comes from the logic of the experiment. Any encoding of an equation that significantly favours recognition of well-formed sub-expressions must rely on an underlying knowledge of mathematics; i.e., on the existence of internal representations of the properties of equations.

An additional source of evidence supporting our hypothesis comes from the fact that performance on non-well-formed sub-expressions provides a recognition advantage over incorrect sub-expressions. This suggests that there is some form

[2]The *minF'* is an extremely conservative statistic. It is considered to be significant if (a) it has an alpha level less than or equal to 0.05, or (b) it has an alpha level less than or equal to 0.1, and both subject (F_1) and item (F_2) analyses are significant at an alpha level of less than or equal to 0.05 (Santa, Miller and Shaw, 1979)

of internal mechanism that can rapidly reconstruct the equation from well-formed components, and thus identify sub-expressions that lie across component borders.

General Discussion

The natural question to ask now is what structural principle underlies the encoding of mathematical equations. In the introduction, we considered the possibility that humans might parse mathematical expressions in a manner similar to the way in which they parse natural language. In an experiment conducted by Johnson (1968) it was shown that it was easier for participants to learn sequences of words that conform to grammatical units (phrases), than to learn sequences of words which are equally probable and acceptable, but do not conform to grammatical units. This result is analogous to our own result for equations, in which syntactically well-formed sub-expressions are more readily recognised than non-well-formed sub-expressions, suggesting that experienced mathematicians might encode equations according to the rules of a grammar.

In natural language, the use of a phrase structure grammar allows us to construct *parse trees* for sentences. It is natural to ask then, whether the internal representation used by mathematicians might also utilise a *parse tree* when encoding an equation. For example, consider the following algebraic expression.

$$(x - 3y)^2$$

Figure 1 shows a parse tree of this expression based on mathematical syntax. The dashed boxes in the diagram would be equivalent to phrases in natural language, with the top node of tree (which contains the entire expression) being equivalent to a sentence. Just as natural language consists of noun phrases and verb phrases and so on, the dashed box containing $x - 3y$ might, for example, be considered a subtraction phrase.

While the results of the experiment discussed in this paper do not contradict the possibility of a hierarchical structure such as a parse tree, it will require much further research before such a conjecture can be confirmed. Our research direction for the immediate future therefore is aimed at both reinforcing the results of the experiment conducted, and investigating further the nature of the internal representation used by mathematicians to encode equations. There are two experiments in particular which we plan to conduct in order to meet these aims.

The first of these is essentially the experiment described in this paper, except that rather than using experienced mathematicians as participants, people with very little experience with mathematics will participate. Since the participants have a very weak background in working with mathematical syntax, we would expect no significant advantage in recognising well-formed sub-expressions over non-well-formed sub-expressions. This experiment would be aimed at providing further evidence to support the hypothesis that knowledge of

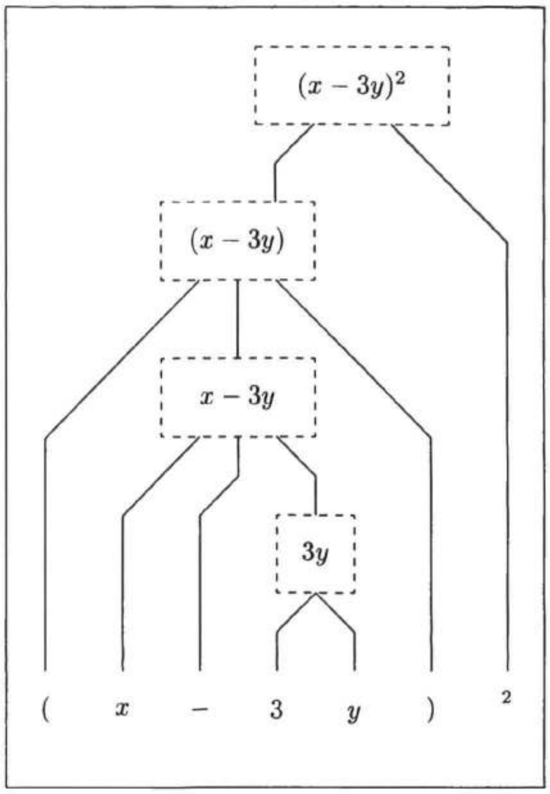

Figure 1: Parse Tree for $(x - 3y)^2$

mathematical syntax is used by experienced mathematicians when encoding equations.

The other experiment is designed to examine the nature of the internal representation used to encode mathematical expressions. The structure of the experiment is similar to the one described in this paper. However, the sub-expressions presented will be of slightly different types. While there will be an incorrect type to again act as fillers, the focus will be on the two types whose content is drawn from their corresponding equations. Consider the following algebraic expression.

$$\frac{8y - 3x^2}{(x - 2y)^3}$$

The sub-expression $x - 2y$ has a valid syntax, and it conveys the same meaning on its own as it conveyed in the equation. It is well-formed, and in a parse tree it would form a phrasal node. However, while the sub-expression $y - 3x$ also has a valid mathematical syntax, it conveys a different meaning on its own than it does in the equation. It would not form a phrasal node on a parse tree.

If the conjecture that mathematicians use a parse tree when encoding an equation is correct, then it would be expected that the first well-formed sub-expression would be recognised significantly faster than the sub-expression which does not form a phrasal node. The result of such an experiment would indicate if the examination of parse trees as an internal representation for encoding equations, is a worthwhile line of investigation.

Finally, it is hoped that this research will not be limited to just mathematical expressions. There are other visual languages with a two-dimensional structure and a well-defined syntax, such as finite state automatas, sheet music and chemical structural formulae. Similar questions need to be asked about these visual languages. A long term aim of this research is to examine the perception of visual languages in general.

References

Akmajian, A., Demers, R.A., & Harnish, R.M. (1984). *Linguistics: An Introduction to Language and Communication* (2nd ed.). Massachusetts: MIT Press.

Anderson, R.H. (1977). Two-dimensional mathematical notation. In K.S. Fu (Ed.), *Syntactic Pattern Recognition Applications*. New York: Springer-Verlag.

Clark, H.H. (1973). The language-as-fixed-effect fallacy: A critique of language statistics in psychological research. *Journal of Verbal Learning and Verbal Behavior, 12*, 335–359.

Glasgow, J., Narayanan, N.H., & Chandrasekaran, B. (Eds.). (1995). *Diagrammatic Reasoning: Cognitive and Computational Perspectives.* AAAI Press.

Johnson, N.F. (1968). The influence of grammatical units on learning. *Journal of Verbal Learning and Verbal Behavior, 7*, 236–240.

Johnson, N.F. (1970). Chunking and organization in the process of recall. In G.H. Bower (Ed.), *The Psychology of Learning and Motivation*, (Vol. 4). New York: Academic Press.

Marriott, K., Meyer, B., & Wittenburg, K. (1998). A survey of visual language specification and recognition. In K. Marriott & B. Meyer (Eds.), *Theory of Visual Languages*. New York: Springer-Verlag.

Miller, G.A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review, 63*, 81–97.

Santa, J.L., Miller, J.J., & Shaw, M.L. (1979). Using Quasi *F* to prevent inflation due to stimulus variation. *Psychological Bulletin, 86*, 37–46.