A Learning Environment for Promoting Structured Algebraic Thinking in Children

Sergio Gutiérrez[1], Manolis Mavrikis[2] and Darren Pearce[1]
London Knowledge Lab
[1]Birkbeck College [2]Institute of Education
23-29 Emerald St, WC1N 3QS London (UK)
{sergut,darrenp}@dcs.bbk.ac.uk m.mavrikis@ioe.ac.uk

Abstract

Although the notion of generality is central in mathematics and science, being able to identify and express general patterns and/or articulating structures is one of the main difficulties for children when they learn mathematics. This paper presents a step towards a set of tools that addresses this problem problem by encouraging students to connect between actual activities within the task and expressions of generality. The paper describes a novel tool — ShapeBuilder — and how it can be used in the context of a well-known generalisation activity: tiling a pond. Insights gained from various small-scale studies with children are discussed followed by a description of how we expect this tool to develop in future work.

1. Introduction

The need to recognise, express and justify generality is at the core of mathematical thinking and scientific enquiry. However, a voluminous body of mathematics education research (see [14]) suggests that expressing generality, recognising and analysing patterns and articulating structure is complex and problematic for students. As suggested in [14], the difficulties students face should not be interpreted simply as their own failure but have to be investigated in the context of the curriculum, the nature of the tasks posed and the tools available for their solution.

In the traditional mathematical curriculum, algebra is a means of expressing generality. However, generalisation is so implicit in algebra that experts no longer notice the strategies they have integrated into their thinking [12]. This causes problems for students who perceive algebra as an *endpoint* rather than a tool for problem solving [14]. One of the reasons for this is that the questions asked and the actions to be taken have no real need for mathematical expression. The latter — even when achieved correctly — is often disconnected from the activity which precedes it which 'neither illuminates the problem nor provides a means for validating its solution' [8].

This paper presents a step towards a set of tools that addresses this problem by encouraging students to connect their actions within activities with the need to express generality. These tools are being developed as part of a learning environment which will promote the learning of mathematical generalisation. The environment will also contain tools that are (a) able to provide personalised support adapted to students construction processes and (b) foster and sustain an effective online learning community by providing assistance to learners based on analyses of their own learning and the activities of the group and advising learners and teachers as to which constructions and reports of others to view, compare, critique and build upon.

Throughout the development of this system, we are following an iterative user-centred design methodology. Former studies have shown that the use of educational tools in the education of mathematics must be carefully integrated with the normal development of the class [8]. In addition, studies about the adoption of educational software highlight that teachers would like the opportunity to be more involved in the whole design process of computer-based environments for their students [16, 15]. Similarly, many researchers report that when developers of innovations are able to match learning environments to teachers goals and expectations for students learning, they are more likely to implement those innovations [3]. Based on the above, teachers in our project are given the active role of being critical users and co-designers of the activities and the learning environment rather than just the role of receivers. Guided by the same principles we attempt to involve students in appropriate parts of the design process.

The paper is structured as follows. Section 2 discusses related work from both the learning sciences and computer science fields. Section 3 presents the learning environment and its application to a specific example generalisation task followed by Section 4 which details the insights obtained from testing this tool with various children. Section 5 closes the paper, drawing general conclusions and presenting the lines of research that remain open.

2. Related Work

The difficulty that algebraic thinking poses to children has been thouroughly studied in the field of mathematics education [11, 7]. There have been numerous attempts to foster a shift towards an appreciation of generality, most notably through finding ways for students to construct their own mathematical models [14]. This modelling approach seeks not only to foster seeing the general in the particular by construction and exploration, but also a sense of ownership of the abstraction process. However, despite some successes, difficulties remain, and these tend to coalesce around the need for appropriate pedagogic support from the teacher.

Two intelligent tutoring system for the domain of algebra are presented in [18, 13], but they deal with actual formulae with variables and numbers, and thus are only suitable for children that already have a clear notion of variable and some skill in algebraic notation; they cannot give information about structural misconceptions in the learners' heads [10]. T-Algebra [9] is a learning environment for the domain of algebra but it also focuses on the language of algebra. Our system works at a cognitively preliminary stage as it tries to bridge the gap between the language and the current objects in the mind of the learner

3. The ShapeBuilder

3.1. Using Expressions to Define Shapes

ShapeBuilder is an environment for early secondary education children (10–12 years) that aims to encourage structured algebraic reasoning. Figure 1 shows the layout of the ShapeBuilder. The Expression Toolbar allows the user to populate the Expression Palette with constants, variables and arbitrarily complex compound expressions involving the operators addition, subtraction, multiplication and division. The Shape List enumerates the shapes that are currently in existence and also allows the creation of new rectangles.[1] The width and height of shapes are specified through dragging expressions from the palette to the appropriate position within the list.

Once the shape has been created and defined, it can be manipulated on the interaction canvas. All shapes can be moved through dragging the mouse and attached to other shapes. Rectangles whose width and/or height are defined by variables can also be resized using the mouse. Shapes can be copied through dragging the thumbnail to the interaction canvas. As the expressions defining a shape group are changed, all copies are updated appropriately.[2]

At any time, the current value of a variable can be edited directly. This can potentially lead to an unlimited number of changes in the dimensions of the shapes on the interaction canvas.

3.2. Using Shapes to Define Expressions

As described so far, ShapeBuilder provides facilities for the learner to build shapes using expressions. However, a critical feature of the software allows users to *define expressions using shapes*. Specifically, by double-clicking on an edge of a rectangle, the user is able to create an expression which, at all times, evaluates to the current value of that dimension of the rectangle. This is represented iconically.[3]

3.3. Using ShapeBuilder for Pond Tiling

Pond tiling is a very typical generalisation task for children in our target age group. Given a rectangular pond with a particular integer width and height, the task entails challenging the child to determine how many 1×1 tiles are required to surround it. Although simple, this task is surprisingly rich in that it lends itself to a variety of different solutions as shown in Figure 2. These tilings correspond to the general algebraic equations $2a+2b+4$, $2(a+1)+2(b+1)$, $2a+2(b+2)$ and $2(a+2)+2b$. Such scope for alternatives allows the learner to realize how multiple valid solutions to a problem lead to different — but equivalent — expressions. There is then an incentive to develop some of the basic rules of algebra intuitively such as commutativity and associativity as well as provocation to collaborate with other children who have found correct but different solutions.

Figure 3 details the gradual construction of one of these possible solutions using the interface features of the ShapeBuilder as described in the previous sections. Having constructed this solution, the user could then change the values of the variables and observe their resulting tiling thus validating the generality of their solution (across these specific cases).

[1] Creation of other shapes will be explored in future work.

[2] Note that shapes are grouped only when copied through thumbnail dragging. Creating a shape explicitly using the same expressions will lead to a separate entry in the shape list.

[3] We envisage that this functionality will be extended in the future so that, for example, a user could create a variable based on the interior angle of a shape.

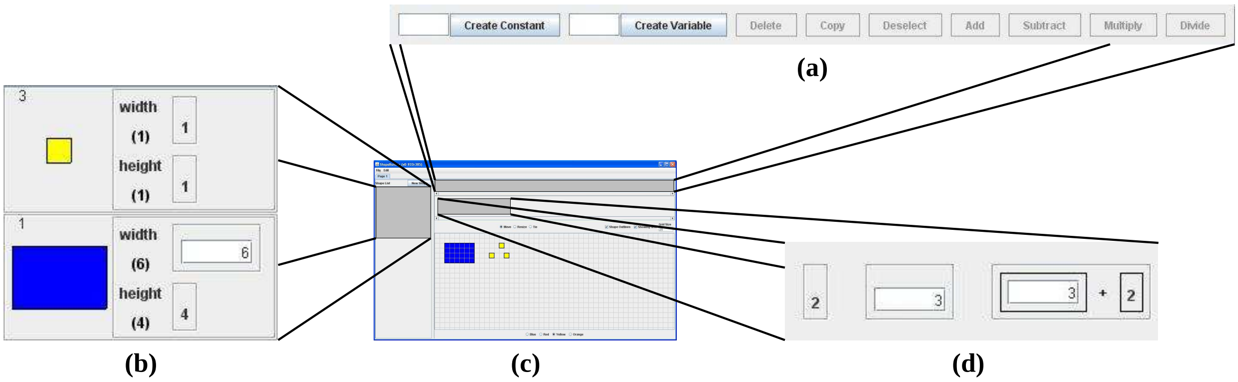


Figure 1. The layout of the ShapeBuilder. (a) the Expression Toolbar; (b) the Shape List; (c) the overall ShapeBuilder interface (the gridded area is the interaction canvas); (d) the Expression Palette.

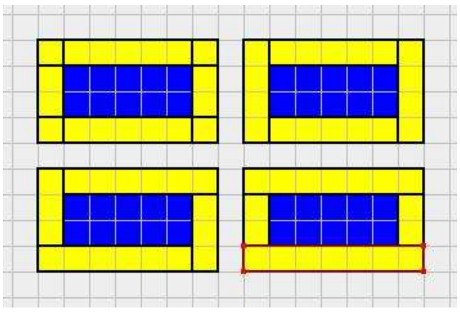


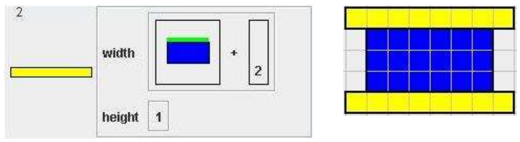
Figure 2. Four examples of general solutions

4. Insights from User Testing

At this stage of the development process, various iterations of the ShapeBuilder have been tested with students selected by teachers who collaborate with our project. The objectives of these small-scale studies are (a) to refine aspects of the user interface (in line with our user-centred design methodology) and (b) to investigate how students approach the task in this innovative situation. The studies involved two or three students working with the system in the presence of their teacher and/or researchers from our team. Students were first presented with simple tasks which were designed to familiarise them with the user interface and introduce the possible ways of constructing shapes and expressions. After the familiarisation session, students are presented with the pond tiling activity which asks them to develop an expression that represents a rule that calculates the number of tiles needed to surround a pond. The 10 students who have used the system so far seemed very interested and engaged with the tasks posed. While they quickly perceive the affordances of the interface, some early ses-

Step 1 The user creates two independent variables, clicks on ‘New Shape’ and then drags these expressions to define the width and height of their pond (rectangle).

Step 2 The user creates an iconic expression for the pond width and adds 2 to it. They then use this to define the width of the horizontal tile. They copy this through dragging to create two instances.



Step 3 They then create an iconic expression for the height of the pond and use this to define the height of the vertical tile. Once again, they copy this through dragging.

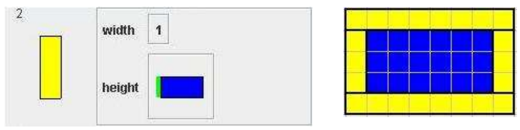


Figure 3. Using ShapeBuilder for the pond tiling task. Each step shows the latest entry in the shape list and the current state of the canvas.

sions helped us recognise the importance of finding motivating familiarisation tasks that would introduce students to the system in an open-ended and natural manner. Their constructions during the pond-tiling task show that they understand the notion of iconic variables. This is demonstrated particularly well when they are asked to explain their rules to other students. Not only they use references from the user interface of the system but they also create new constructions to justify their arguments. As explained in Section 3.3, one of the interesting aspects of the pond tiling task is the different ways for constructing a representation.

This variety lends itself to interesting pedagogical approaches. In terms of their constructions, an approach which has been established as potentially powerful in Dynamic Geometry Environments [1], is often referred to as 'messing up' [6]. This strategy challenges students to produce constructions which cannot be 'messed-up' when changed. One of the main advantages of ShapeBuilder is that it allows students to put their creations to exactly this kind of test. When they are asked to create shapes in relation to others (like when surrounding a pond with tiles) they can construct this for the particular case given (using constants and/or variables) or in a more general way (with icon variables related to the shapes given). However, only solutions constructed in a general way will pass this test [6]. Figure 4 shows a non-general solution that becomes invalid as soon as the pond changes its size. At the moment, the teacher (or indeed another student) can easily mess-up such constructions by changing, for example, the width of the pond. This raises discussions between teachers and students. This 'provocation' to generalise [12] provides students with the opportunity to realise that there is an advantage to using icon variables. In this way, it helps them start thinking in terms of abstract characteristics of the task rather than actual measurements.

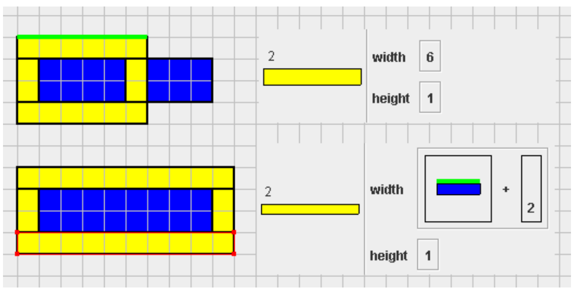


Figure 4. Messing up highlights the difference between a solution which is general and one which is not. The first one is not valid anymore if the pond size is changed, while the second one is.

In terms of the solutions that students produced during the small-scale studies, they varied from specific to general. With the help of the teacher/researcher, and usually by referring to a general construction that cannot be messed up, students were able to derive more general rules and compare them with each other. At the moment, this requires an intensive one-to-one interaction that would not be possible in a classroom. Analysing these interactions further will inform the design of the intelligent components of the system that will be able to provide support to students and teachers in a classroom context. This is discussed as future work in the next section.

5. Conclusions and Future Work

Algebra in general, and the development of mathematical patterns in particular, is one of the more important difficulties in the learning of mathematics at an early age. We have presented a tool that encourages the user to connect coherently between some concrete activity involving shapes (such as pond tiling) and the actual algebraic expressions that describe the general solution to the problem.

Some small-scale studies have been performed with young learners achieving encouraging results, as they where able to construct *general* solutions (and describe them verbally) that they could not *mess up*. However, the actual level of learning of algebra by them is a matter that demands further and more formal investigation, with an assessment strategy and much larger sample.

Immediate progress of our research tends towards provision of intelligent feedback on the activity by the tool itself. This would help students in self-learning and distance learning scenarios as well as act as an effective supplement to the support that a teacher provides in the classroom. Since the actions of the students will be logged by the system as they happen, . This allows for the provision of feedback based both in real time analysis of current actions as well as offline analysis.

However, there are two important challenges. First, the exploratory nature of the task we are dealing with makes it difficult to establish routes that the students may follow through the state-space of the task, and there is a broad grey area between "clearly correct" and "clearly incorrect" that cannot be easily assessed. The system cannot be an ITS in the classical sense where there are correct and incorrect answers. Some techniques have been used in the past for dealing with exploratory environments like Bayesian networks [4, 5], combinations of heuristics and formal methods [17] or multi-agent systems [19] based in Balacheff's formalization of mathematical knowledge [2]. Additionally, the domain of mathematical generalisation is vague and difficult to formalise. Our intention therefore is to develop ontologies with which we can categorise specific task domains and possible strategies within them. In this way, we aim to develop a generic system that enables effective and meaningful interaction with a wide range of generalisation tasks.

References

[1] N. Balacheff and J. Kaput. Computer-based learning environments in mathematics. In A. J. Bishop, K. Clements, C. Keitel, J. Kilpatrick, and C. Laborde, editors, *International Handbook on Mathematics Education*. Dordrecht: Kluwer, 1996.

[2] Nicolas Balacheff. A modelling challenge: untangling learner' knowing. In *L'apprentissage, un approche transdisciplinaire (JIOSC 2000)*, pages 7–16, 2000.

[3] P. Blumenfeld, B. J. Fishman, J. Krajcik, R. W. Marx, and E. Soloway. Creating usable innovations in systemic reform: Scaling up technology-embedded project-based science in urban schools. *Educational Psychologist*, 35:149–164, 2000.

[4] Andrea Bunt and Cristina Conati. Assessing effective exploration in open learning environments using bayesian networks. In *Intelligent Tutoring Systems*, pages 698–707. Springer, 2002.

[5] Andrea Bunt and Cristina Conati. Probabilistic student modelling to improve exploratory behaviour. *User Modeling and User-Adapted Interaction*, 13:269–309, August 2003.

[6] Lulu Healy, Reinhard Heltz, Celia Hoyles, and Richard Noss. Messing up. *Micromath*, 10:14–17, 1994.

[7] Lulu Healy and Celia Hoyles. A study of proof conceptions in algebra. *Journal for Research in Mathematics Education*, 31(4):396–428, July 2000.

[8] Celia Hoyles and Lulu Healy. Visual and symbolic reasoning in mathematics: Making connections with computers? *Mathematical Thinking and Learning*, 1(1):59–84, 1999.

[9] M. Issakova, D. Lepp, and R. Prank. Input design in interactive learning environment t-algebra. In *IEEE Int. Conf. on Advanced Learning Technologies (ICALT)*, pages 489 – 491, 2005.

[10] Dietmar Küchemann. *Algebra*, pages 102–119. Antony Rowe Publishing Services, 1981.

[11] Dietmar Küchemann and Celia Hoyles. Investigating factor that influence students' mathematical reasoning. In *PME XXV*, volume 3, pages 257–264, 2001.

[12] John Mason, Alan Graham, and Sue Johnston-Wilder. *Developing Thinking in Algebra*. Paul Chapman Publishing, 2005.

[13] Maria Moundridou and Maria Virvou. Authoring and delivering adpative web-based textbooks using WEAR. In *ICALT*, pages 185–188, 2001.

[14] Richard Noss, Lulu Healy, and Celia Hoyles. The construction of mathematical meanings: Connecting the visual with the symbolic. *Educational Studies in Mathematics*, 33(2):203–233, July 1997.

[15] W Pelgrum. Obstacles to the integration of ict in education: results from a world-wide educational assessment. *Computers and Education*, 37:163–178, 2001.

[16] J. Underwood, S. Cavendish, S. Dowling, K. Fogelman, and T. Lawson. Are integrated learning systems effective learning support tools? *Computers and Education*, 26:33–40, 1996.

[17] Koen Veermans and Wouter R. van Joolingen. Combining heuristics and formal methods in a tool for supporting simulation-based discovery learning. In *Intelligent Tutoring Systems*, volume 3220, pages 217–226. Springer, 2004.

[18] Maria Virvou and Maria Moundridou. An authoring tool for algebra-related domains. In *HCI (2)*, pages 647–651, 1999.

[19] Carine Webber, Silvie Pesty, and Nicolas Balacheff. A multi-agent and emergent approach to learner modelling. In *Proceedings of the 8th Iberoamerican Conference on Artificial Intelligence*, 2002.