Challenges for Intelligent Support in Exploratory Learning: the case of ShapeBuilder*

Mihaela Cocea, Sergio Gutierrez-Santos and George D. Magoulas

London Knowledge Lab, Birkbeck College, University of London,
23-29 Emerald Street, London, WC1N 3QS, UK
{mihaela,sergut,gmagoulas}@dcs.bbk.ac.uk

Abstract. Exploratory learning environments give a lot of freedom to learners to explore tasks on their own. However, although this can have a positive effect on learning, the lack of structure makes it difficult to provide intelligent support in such systems. Furthermore, the open nature of these systems makes it harder to compare the support provided by different systems. This papers describes a series of scenarios that demonstrate these challenges in the context of an exploratory learning environment for mathematical generalisation and proposes a formulation that employs cases as a form of knowledge representation for modelling this domain.

1 Introduction

Exploratory Learning Environments (ELEs) provide students with a lot of freedom to explore and play, rather than being constrained or very directed. In ELEs, learners are encouraged to explore a broad set of possibilities, and construct models within them[1]. They arguably have a more positive impact on the user's learning than structured guided environments, due to their open nature in line with constructivist principles [2]. However, their open nature makes it difficult to design a system that supports the user, as the possible different actions are broad and mostly unstructured. On the other hand, an exploratory environment without any support or guidance can actually hinder learning [3].

This paper describes several scenarios that can take place in an exploratory learning environment for mathematical generalisation, called ShapeBuilder, and explores alternatives for providing intelligent support through scenarios. It also describes succintly a modelling strategy that is useful for the environment to identify when these scenarios take place.

The paper is structured as follows. The next section provides some background of the specific microworld (in the sense of [4]) that has been the main drive behind this work. Based on preliminary pilot studies in which it was deployed, several possibilities for support have been identified. The scenarios for

* Work funded by TLRP e-Learning Phase-II; RES-139-25-0381.

[1] The term ELE is sometimes used for referring to systems used in simulation-based learning [1], as they usually allow a limited degree of exploration.

support in such an environment are described in Section 3. A knowledge representation scheme that employs case-based reasoning is presented in Section 4 as a a first step toward developing an engine that would underpin the provision of intelligent support in our microword. Finally, Section 5 closes the paper, drawing the final conclusions.

2 Background

ShapeBuilder [5] is an exploratory environment that is built to support the development of algebraic thinking, aimed at learners between 11 and 14 years old. From the many mathematical skills that children learn in a typical curriculum, the ability to express general concepts through algebra is arguably the most difficult to grasp. However, it is also one of the most important because the ability to generalise is at the core of scientific enquiry, and the use of algebra to express general concepts is basic in most branches of mathematics. Therefore, there is a need for systems that support students in the acquisition of the basic concepts of generalisation thinking, scaffolding their learning towards algebra.

ShapeBuilder gives the learners the opportunity of creating shapes (usually rectangles, although other shapes are possible) on a square grid. The shapes are defined by several attributes (e.g. "length") that the learners can set and/or modify. The interesting features of ShapeBuilder allow users to select specific attributes from these shapes and use them to create other shapes.[2] The attributes of the shapes are represented by means of icon-variables, that are graphical representations of attributes of shapes. Their graphical nature makes them more intuitive and arguably easier to grasp for young students, that have been reported to have difficulties understanding the use of letters to represent general concepts [6]. Icon-variables can be combined to form algebraic expressions (this will be discussed in detail in the next section, see for example Scenario 5).

Several pilot studies have been conducted in classrooms to explore how students undertaking a common activity in mathematical curricula known as pond-tiling work with ShapeBuilder. In this activity, students are given a pond (usaully rectangular) and are asked to surround it with square tiles (see Figure 1a). Such a task helps learners develop the ability to predict the number of tiles that are necessary to surround *any* pond, given its dimensions: this is, in essence, an algebraic expression. There are several ways of surrounding a pond of any size or shape, and each one leads to a different (yet equivalent) algebraic expression. For example, the number of tiles needed to surround a rectangular pond can be generally expressed as "twice the length, plus twice two more than the breadth" (i.e. $2L + 2(B + 2)$) or "twice two more than the length, plus twice the breadth" (i.e. $2(L + 2) + 2B$); many others can be developed. Given these different arrangements, students can be prompted to have interesting discussions about the equivalence of the expressions derived from each viewpoint.

[2] The procedure, as well as a full description of system features, is described in [5].

3 Scenarios for Intelligent Support in ShapeBuilder

There are many open problems with regard to ELEs. This section looks at some of the challenges relevant for this workshop through user scenarios from ShapeBuilder.

3.1 Balance freedom with control

Scenario 1. In this scenario a student is working on a model that the system identifies as being far way from a typical or "desirable" construction according to some knowledge base that it stores. This may happen as a result of the student: (a) coming up with an "unusual/uncommon" solution that is not stored in the knowledge base of the system, or (b) acting very diferently from the typical path of conducting the task (see Figure 1). Two posible strategies to follow would be: (i) letting the learner continue with the exploration of the task, or (ii) guiding the learner to a "desirable solution". Unfortunately, it is unclear what would be the best way to provide support here, as it depends on the context. In certain instances of case (a), if action (ii) is selected, the learner would be provided a support that is not needed. On the other hand, applying action (i) in case (b) would result in the student needing a support that is not provided. Neither situation can be expected to end in a better learning experience.

A possible solution for intelligent support here would be that the system informs the teacher when this situation occurs in order to let him/her choose a course of action, e.g. turn off feedback in situation (a) (at least for a while until the current context becomes clearer) and let the learner explore further, or let the system guide the student in situation (b).

Scenario 2. In this scenario, a learner initially started the construction of a model but then suddenly moved into a totally irrelevant activity; the system has detected this "abnormal behaviour". Should the support take the form of a system intervention that would prompt the learner to come back to the relevant task, and after how long should this intervention occur? Possible ways to provide support here are the system to: (a) intervene automatically when a time threshold/interval is reached; (b) inform the teacher about this particular behaviour and suggest an intervention, or leaves the teacher to decide the next action.

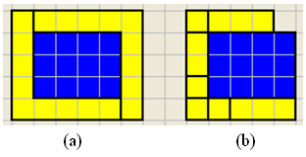

Fig. 1. Scenario 1: (a) correct innovative strategy; (b) "innefficent" and non-general and incomplete strategy.

3.2 The value of correct/incorrect actions

Making mistakes and observing the results of "wrong actions" is considered quite useful in exploratory learning. It encourages reflection and self-explanation [7]. In ELEs, the learner should have the freedom to explore without continuous distractions by prompts at each wrong action. However, learners do need guidance when they are stuck or their actions are not leading to a sensible outcome. One way to do this is to make use of the consequences of their actions "in the long run". What appears to be a wrong action could be caused by misconceptions, slips or migrating bugs [8] and some of them would require immediate feedback (e.g. slips, random errors) or delayed feedback (e.g. misconceptions). The following scenarios attempt to provide some examples for this type of situation.

Scenario 3. Our preliminary studies have shown that learners often start with non-general strategies like surrounding the pond using 1 by 1 tiles or in another non-systematic manner as illustrated in Figure 1b. Although this might be a correct way for surrounding a pond, it will not lead to generality, i.e. the solution provided cannot be generalised to other similar cases. Intervention at this point may be helpful especially if it points out the lack of generality of the approach. An effective strategy from the literature of Dynamic Geometry Environments is "messing up" [9]. This involves changing something in the figure that demonstrates that the student's construction, although it may look correct, does not follow the constraints that would make it general enough in this environment. For example, in our case the system or the teacher could change the size of the pond. "Messing-up" could also be used as a way to support the learner by checking the correctness of a solution that the learner believes is general. This process can be triggered by the system, or can be led by the user (e.g. in a collaborative scenario, in which a learner tests the generality of solutions produced by their peers).

Scenario 4. For this scenario, consider the tilings illustrated in Figure 2. They are both correct but different, which shows that the way to approach a task may vary depending on personal preferences. One may argue that the surrounding in Figure 2b is not exactly right, but it would be difficult to argue that it is wrong. Situations like this can be handled with constraints on the position of shapes and thus both ways of surrounding the pond could be identified as correct.

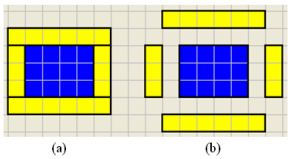

Fig. 2. Scenario 4 - "I strategy": (a) shapes tied together; (b) shapes at a distance.

Scenario 5. Some learners construct the general expression for surrounding the pond in the minimal form (Figure 3a) and other do not simplify their expressions (Figure 3b). Although the minimal form is "desirable", the non-simplified ones are still correct. Depending on the goals of the task, the "expanded" expressions could be acceptable or not.

3.3 The timing of feedback

Feedback on correctness of solution is not enough for ELEs, as support is often needed during an activity, i.e. in ShapeBuilder during model construction not just when construction is complete.

Scenario 6. Here the learner is working on the task and has constructed a model that is partially correct. She makes several changes but still does not get it right. Should an intervention occur at this point or maybe earlier/later? Should the learner be left to explore the task further? This scenario is similar to *Scenario 1*, except that the focus now is on the exact timing of an intervention should it be needed, as opposed to the decision of whether or not to intervene as discussed in *Scenario 1*. Possible strategies for support in this case could be: (a) a help button that the learner could use (it has been reported that this might lead to other problems such as help abuse [10] or gaming the system [11]); (b) a "what others did next" button that shows what other learners did in a similar situation (this might require tools to allow teachers to validate learners choices in advance so that the next steps displayed are beneficial for learning). As certain studies show that some learners tend to avoid system's help [12], it would be interesting to investigate how the "what others did next button" is used. Help on request is a typical example of this scenario as it allows learners to request support when they feel stuck.

Scenario 7. In this scenario support during model construction is needed depending on the stage within the task. The support can take the form of examples, which could be partial or complete solution depending on the context. They could also be "standard examples", i.e. pre-encoded in the system, or models developed by peers which have been previously validated by the teacher or by peer learners.

Fig. 3. Scenario 5, "I strategy": (a) simplified expression; (b) "expanded" expression.

3.4 Teacher support

Assisting teachers in developing teaching strategies, informing them of students who are in difficulty, as well as the appearance of common misconceptions or problematic strategies is another form of support needed in ELEs like Shape-Builder.

As it is impossible for a teacher to attend to all students in a class, general statistics of the whole class are particularly beneficial in identifying outliers: pupils that are doing very well or very poorly. Knowing the overall status of the class with respect to the task/learning goals makes it possible for the teacher to give tasks to groups of students that are appropriate to their level or abilities, or even challenge them.

At the same time, knowing what a student has been doing helps the teacher to support the student. To facilitate this, the following information could be provided to the teacher: (a) fast replays of the most recent interactions of the student with the system; (b) screenshots taken at certain times during the task that create a "visual diary" of some important steps of the construction process followed by a particular student or a "scrapbook" for a student that experiments with alternative solution/strategies of approaching a task ; (c) summary statistics like current stage within the task, number of help requests, etc.

3.5 Support for collaboration

While the focus so far was on individual learning, it is acknowledged that effective learning benefits from recognising the importance of participation in communities of practice. There is however a particular challenge when designing for collaborative learning in mathematics. This stems from the difficulty of identifying engaging things for students to talk about as abstract concepts like generalisation are not necessarily intrinsically motivating. The challenge, therefore, is to support collaboration between students around models produced by their community, help them understand what other students in the community are doing and identify others working on the same or similar models.

Group formation is particularly important in this scenario as collaboration is more beneficial when the learners are paired up according to educational criteria like: (a) if the goal is to help a learner that has difficulties, the other member of the pair should be more advanced on the same task and follow the same or a similar strategy; (b) if the goal is to point out that the same task can be solved with different approaches, the pairing should be done with learners that have different models/expressions. Models' similarity can be measured by comparing their characteristics and the strategies used for their construction (this will be discussed in the next section).

4 Towards Personalised Feedback and Support

Research into personalised feedback and support has primarily focused on the learner's knowledge "level". While this might work well in structured learning

environments that adopt the role of an expert tutor presenting concepts gradually and use questions and quizzes to assess students knowledge, it will not suffice for our purposes with its more conceptual focus. In ShapeBuilder, the problem is to find ways to support the process of the construction of models by exploiting information collected during individual model construction, identify similarities among students as expressed by the characteristics of their models and the strategies used for their construction. Furthermore, we need to recognise patterns in student behaviour during the various stages of the learning process. This process is exploratory and investigative and leads to progressive construction of structural knowledge. This information can then be used to provide support and facilitate collaboration with peers.

To this end we present an approach to personalised feedback that is based on recent student actions rather than simply on their knowledge levels, and provide different types of feedback during this progressive provision; for example, advice on the quality of learners' constructions, possible next actions, use of appropriate construction components, and possible alternative construction strategies. The ultimate aim is to create an intelligent support system that will foster the learner's articulation of general patterns and relationships, suggest exploration of special cases, critique model solutions and seek justification for students' models.

Typically, within a task or activity there are several sub-tasks, and the activity is sequenced within the system so as to know at any time the current context. Attributes and relations are stored in cases, which represent characteristic components of a learner's construction. They carry different relevance depending on the context, which in ShapeBuilder corresponds to different stages of the constructivist learning process that learners go through as they explore the various sub-tasks within a learning activity. Through comparison of strategies the system can identify how far a learner is from desired or alternative construction strategies (this could be used to deal with situations like the ones described in Scenarios 1, 2, 3 and 4), whether the construction includes the right components, or the similarity between the models of different learners (this could be used when informing decisions about collaboration. Comparisons are based on defining similarity measures as described below.

4.1 Representation using case-based reasoning

In this subsection we present a formulation to represent attributes, relations and strategies based on case-based reasoning (CBR).

Definition 1. *A case is defined as* $C_i = \{F_i, RA_i, RC_i\}$*, where* C_i *represents the case and* F_i *is a set of attributes.* RA_i *is a set of relations between attributes and* RC_i *is a set of relations between* C_i *and other cases respectively.*

Definition 2. *The set of attributes is defined as* $F_i = \{\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_N}\}$.

The set F_i includes three types of attributes: (a) numeric, (b) variables and (c) binary. Variables refer to different string values that an attribute can take,

Table 1. The set of attributes (F_i) of a case.

Category	Name	Label	Possible Values
Shape	Shape type	α_{i_1}	Rectangle(/L-Shape/T-Shape)
Dimensions of shape	Width type	α_{i_2}	constant (c)/variable (v)/icon variable (iv)/numeric expression (n_exp)/expression with iv(s) (iv_exp)
	⋮	⋮	⋮
	Thickness type	α_{i_v}	c /v /iv /n_exp /iv_exp
	Width value	$\alpha_{i_{v+1}}$	numeric value
	⋮	⋮	⋮
	Thickness value	α_{i_w}	c /v /iv /n_exp /iv_exp
Part of Strategy	$PartOfS_1$	$\alpha_{i_{w+1}}$	1
	$PartOfS_2$	$\alpha_{i_{w+2}}$	0
	$PartOfS_r$	α_{i_N}	0

and binary attributes indicate whether a case can be considered in formulating a particular strategy (1) or not (0). The set of attributes of a generic case for ShapeBuilder is presented in Table 1. The first v attributes ($\alpha_{i_j}, j = \overline{1, v}$) are variables, the ones from $v+1$ to w are numeric ($\alpha_{i_j}, j = \overline{v+1, w}$) and the rest are binary ($\alpha_{i_j}, j = \overline{w+1, N}$).

Definition 3. *The set of relations between attributes of the current case and attributes of other cases (as well as attributes of the same case) is represented as* $RA_i = \{RA_{i_1}, RA_{i_2}, \ldots, RA_{i_M}\}$*, where at least one of the attributes in each relation* $RA_{i_m}, \forall m = \overline{1, M}$*, is from the set of attributes of the current case* F_i*.*

Two types of binary relations are used: (a) a *dependency relation* (D_{i_s}) is defined as $(\alpha_{i_k}, \alpha_{j_l}) \in D_{i_s} \Leftrightarrow \alpha_{i_k} = DEP(\alpha_{j_l})$, where $DEP : \alpha_{i_k} \rightarrow \alpha_{j_l}$ for attributes $\alpha_{i_k}$and α_{j_l} that are variables of cases i and j (where $i = j$ or $i \neq j$), and means that α_{i_k} depends on (is built upon) α_{j_l} (if $i = j$, $k \neq l$ is a condition as to avoid circular dependencies) (e.g. the width type of a case is built upon the height type of the same case; the width type of a case is built upon the width type of another case, an so on); (b) a *value relation* (V_{i_s}) is defined as $(\alpha_{i_k}, \alpha_{j_l}) \in V_{i_s} \Leftrightarrow \alpha_{i_k} = f(\alpha_{j_l})$, where $\alpha_{i_k}$and α_{j_l} are numeric attributes and f is a function and could have different forms depending on context (e.g. the height of a shape is two times its width; the width of a shape is three times the height of another shape, etc.).

Definition 4. *The set of relations between cases is represented as* $RC_i = \{RC_{i_1}, RC_{i_2}, \ldots, RC_{i_P}\}$*, where one of the cases in each relation* $RC_{i_j}, \forall j = \overline{1, P}$ *is the current case* (C_i)*.*

Two time-relations are used: (a) *Prev* relation indicates the previous case with respect to the current case: $(C_i, C_j) \in Prev$ if $t(C_j) < t(C_i)$ and (b) *Next*

relation indicates the next case with respect to the current case: $(C_i, C_k) \in Next$ if $t(C_i) < t(C_k)$. Each case includes at most one of each of these two relations ($p \leq 2$).

Definition 5. *A strategy is defined as* $S_u = \{N_u(C), N_u(RA), N_u(RC)\}$, $u = \overline{1,r}$, *where* $N_u(C_i)$ *is a set of cases,* $N_u(RA_i)$ *is a set of relations between attributes of cases and* $N_u(RC_i)$ *is a set of relations between cases.*

Four different similarity measures are used for comparing cases: Eucledian distance for numeric attributes, Hamming distance for the set of variable attributes, and a Jaccard index for the relation between attributes and the relation between cases. In order to identify the closest strategy to the one employed by a learner, the four distances are combined for each of the compared strategies.

4.2 Identifying similarities in learner's strategies and constructions

Identifying similar strategies can provide insight on the behaviour of learners using the system and initiate different types of feedback and support depending on the situation. This can be eminently suitable when dealing with scenarios such as Scenario 1, 2, 3 and 4. This idea can be extended to identify how similar the models of different learners are in order to form collaborations among peers.

When the learner's construction is equally similar to two strategies, the following options could be adopted: (a) present the learner with the two options and let him/her choose one of the two (an approach that appears more suitable for advanced learners than for novices); (b) automatically suggest one of the two in a systematic way, e.g. present the one that occurs more/less often with other learners; (c) inform the teacher about the learner's trajectory and the frequency of strategies and let him/her decide between the two.

5 Conclusions

Given the open nature of exploratory learning environments, it is difficult to make fair comparisons between different systems, and between the approaches taken for providing intelligent support in them. The goal of this paper is to provide a range of possible scenarios that can take place in a particular exploratory learning environment for mathematical generalisation, in order to provide some common ground on which different techniques for intelligent support can be assessed.

Although most of the examples have been focused on this system, called ShapeBuilder, an effort has been made to extrapolate the situations, so they can be applied to other exploratory environments, like MoPiX [13] or SketchPad [14]. The scenarios cover different aspects, such as the difficulty of asserting what is right or wrong in an exploratory environment, the analysis of the actions of the students and the possibilities for collaboration. For each scenario, the different ways in which a learner can be supported are depicted.

The paper also described a case-based reasoning formulation for the representation of learner behaviour. This approach defines metrics that allow different types of comparisons to be made. These can be further used to inform decision-making for personalising the feedback and the support provided.

References

1. Swaak, J., van Joolingen, W.R. and de Jong, T.: Supporting simulation-based learning: the effects of model progression and assignments on definitional and intuitive knowledge. Learning and Instruction, 8, 235–253 (1998).
2. Piaget, Jean.: The Psychology of Intelligence. New York: Routledge (1950).
3. Kirschner, P., Sweller, J. and Clark, R.E.: Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential and inquiry-based teaching. Educational Psychologist, 41(2), 75–86 (2006).
4. Balacheff, N. and Kaput, J.: Computer-based Learning Environments in Mathematics. In A. J. Bishop, K. Clements, C. Keitel, J. Kilpatrick and C. Laborde (Eds.) International Handbook on Mathematics Education, chapter 13. Kluwer (1996).
5. Gutierrez, S., Mavrikis, M., Pearce, D.: A Learning Environment for Promoting Structured Algebraic Thinking in Children. IEEE Int. Conf. on Advanced Learning Technologies, 74-76, (2008).
6. Kuchemann, D.: Algebra. In Hart, K. M. (Ed.) Children's Understanding of Mathematics, Antony Rowe Publishing Services, 102–119 (1981).
7. Conati C., VanLehn K.: Toward Computer-Based Support of Meta-Cognitive Skills: a Computational Framework to Coach Self-Explanation. International Journal of Artificial Intelligence in Education, 11, 389–415 (2000).
8. VanLehn, K.: Mind Bugs: The Origins of Procedural Misconceptions. MIT Press (1990).
9. Healy, L., Hoelzl, R., Hoyles, C., Noss, R.: Messing Up. Micromath, 10(1), 14–16 (1994).
10. Wood, H., Wood, D.: Help Seeking, Learning, and Contingent Tutoring. Computers and Education, 33, 153-159 (1999)
11. Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z.: Off-Task Behavior in the Cognitive Tutor Classroom: When Students Game the System. Proceedings of ACM CHI 2004: Computer-Human Interaction, 383–390 (2004).
12. Aleven, V., McLaren,B.M., Roll, I., Koedinger, K.R.: Toward tutoring help seeking: Applying cognitive modeling to meta-cognitive skills. Proceedings of the 7th International Conference on Intelligent Tutoring Systems, 227–239 (2004).
13. Part of the ReMath project. Available at *http://www.lkl.ac.uk/mopix/* (last accessed: July 2008).
14. Jackiw, N. and Finzer, W.: The Geometer's Sketchpad: Programming by Geometry. In A. Cypher (Ed.), Watch What I Do: Programming by Demonstration, 293–308 (1993).