Task-oriented modelling of learner behaviour in exploratory learning for mathematical generalisation

Mihaela COCEA [a,1], and George D. MAGOULAS [a]
[a] *London Knowledge Lab, Birkbeck College, 23-29 Emerald Street, WC1N 3QS, London, UK*

Abstract. By their nature, Exploratory Learning Environments allow a high degree of freedom which leads to a diversity of learner trajectories and make the modelling of all possible behaviours difficult. To address this, we propose an approach for knowledge representation and identification of task-dependent strategies that learners follow during exploration. The knowledge representation combines heterogeneous sources of informations which are used for strategy identification by means of appropriate similarity metrics. Scenarios capturing educational aspects are presented and the outputs of the similarity metrics are discussed with examples from the domain of mathematical generalisation.

Keywords. Exploratory learning, strategy identification, similarity metrics, mathematical generalisation

Introduction

The freedom given to the learners in Exploratory Learning Environments (ELEs) [4] allows the learners to follow different paths when solving a task. For this reason ELEs are particularly suitable for domains that allow multiple solutions and where exploration is beneficial for understanding the domain's characteristics. Simulation-based exploratory learning environments allow learners to manipulate the parameters of different models and some may also allow the learners to construct their own models. The system employed in our research, *eXpresser* [2] [9], falls in the latter category and is developed for teaching mathematical generalisation in classrooms.

As in exploratory environments there are rarely unique approaches (and even solutions) to a task, it would be valuable from pedagogical perspective to know the strategies learners adopt when solving a task. This information would allow replacement of 'one-for-all' feedback with personalised guidance. Thus, instead of guiding the learner to a predefined solution that may have nothing in common with the learner's thinking, feedback would be formulated in terms of the learner's approach.

[1] Corresponding Author: Mihaela Cocea, London Knowledge Lab, Birkbeck College, 23-29 Emerald Street, WC1N 3QS, London, UK; E-mail: mihaela@dcs.bbk.ac.uk.

[2] Developed in the context of MiGen Project, funded by the ESRC/EPSRC Teaching and Learning Research Programme (RES-139-25-0381); http://www.migen.org.

This paper is an extended version of the work presented in [3]. We describe the knowledge representation used to delineate the possible strategies that learners could use when solving a task, and the identification mechanism used to distinguish which strategy is followed by the learner. Partial and complete solutions are represented as sequences of cases linked by temporal and dependency relations. These are mapped to the learner's behaviour in the system using the identification mechanism which is based on similarity metrics for each type of information used in the cases.

The paper is structured as follows. Section 1 briefly presents the problem of mathematical generalisation and the software tool employed. Section 2 presents the strategies representation and identification mechanisms. Several pedagogically-driven scenarios are illustrated in Section 3 as a means of validating the inference mechanism and Section 4 concludes the paper.

1. Exploratory Behaviour in Mathematical Generalisation

Mathematical generalisation is at the core of mathematical thinking. Usually some generalisation tasks are given in the context of algebra, as "algebra is, in one sense, the language of generalisation of quantity. It provides experience of, and a language for, expressing generality, manipulating generality, and reasoning about generality" [7]. However, algebra is perceived as separate from what it represents [5] and students do not associate it with generalisation.

To address this issue, eXpresser [9,8] aims to link the visual with the algebraic-like representation of rules. It enables constructions of patterns, creating dependences

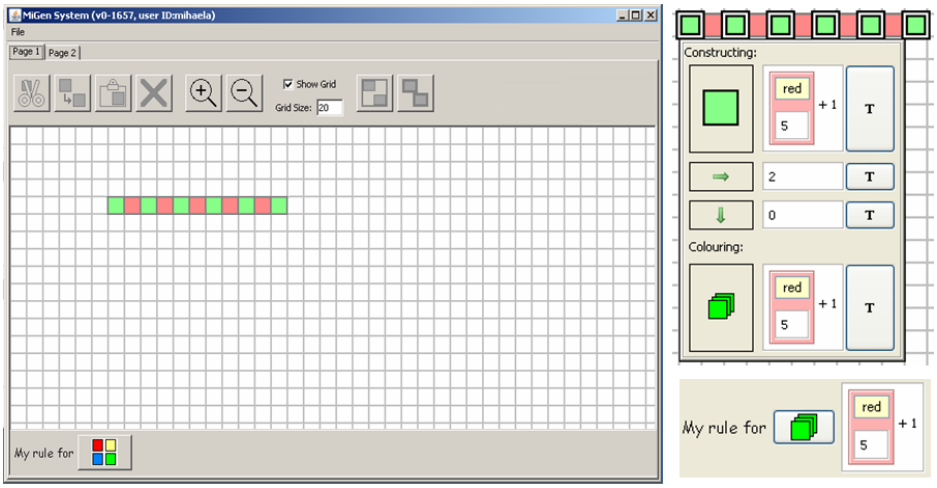

Figure 1. eXpresser screenshots. The screenshot on the left includes a toolbar, an area for pattern construction and an area for defining rules; the toolbar (at the top) allows the following actions: cut, copy, paste, delete, zoom in, zoom out, show grid, grid size (changeable from here or using the zoom tools), group and ungroup; the main area has two patterns - one composed of 5 red (darker colour) tiles and one composed of 6 green (lighter colour) tiles; the rule area (at the bottom) supports definition of rules based on the colours used in the patterns. The screenshot on the top right shows the *property list* of the green pattern. The bottom right screenshot gives an example of a rule for the number of green tiles.

between them, naming properties of patterns and creating algebraic-like rules with either names or numbers. Some screenshots are displayed in Figure 1, illustrating the system, the *properties list* of a pattern that is dependent on another one and an example of a rule. In the screenshot on the left, in the main area of the screen two patterns are displayed: a red (darker colour) pattern, having five tiles and a green (lighter colour) one, having six tiles. In the property list for the green pattern displayed in the top right corner, it can be seen that this pattern depends on the red one by the fact that the number of iterations of green tiles is set to 'the number of tiles of the red pattern plus one'. The second property in the list establishes the units for *moving left* on the horizontal axis - in the current case it is set to 2, which makes green tiles appear with gaps in between them; these are filled by red tiles, which also have *moving left* property set to 2. The following property sets the units for *moving down* on the vertical axis - in the current case is set to 0. The last property establishes the number needed to colour all the tiles in the pattern; in the current case it's the same as the number of iterations in the pattern. However, if a pattern is a group of several tiles, this would not be the case anymore; for example, if a pattern is a group of three tiles and is repeated/iterated five times, the number required to colour it would be three times five. The bottom right screenshot displays a simple example of a rule for the number of green tiles, which is the number of red tiles plus 1.

The construction in Figure 1 and the rule in the bottom-right corner constitutes one possible solution to the following generalisation problem: how many green tiles are required for any given number of red tiles in order to produce the construction? Although the rule is unique, there are several ways to build the construction. For example, another way of constructing it would be to define a pattern as a group of a green and a red tile and repeat it five times (i.e. equal to the number of red tiles) along the horizontal axis and than add an extra green tile. Therefore, there are several *strategies* that one could follow when building a particular construction and the *components* of these strategies are *patterns* with certain *attributes* or properties (i.e. the ones defined in the property list) and linked with certain *relations*, such as the dependency relation illustrated in Figure 1 (i.e. the number of green tiles depends on the number of red ones). In the following section, a formalisation for knowledge representation is presented that covers both the components of a strategy and the strategy as a whole.

2. Strategies Representation and Identification

In our approach, strategies in building a construction are represented as a series of cases [6] with certain relations between them. 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, corresponding to the *property list* of a pattern. RA_i is a set of relations between attributes and RC_i is a set of relations between C_i and other cases, respectively.

The set of *attributes* is defined as $F_i = \{\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_N}\}$ and it includes two types of attributes: (a) numeric and (b) variables. Variables refer to different string values (i.e. type) that an attribute can take; some numeric attributes are binary, indicating whether a case is a group of patterns, or if it can be considered in formulating a particular strategy or not. The later is represented as a "part-of-strategy" function: $PartOfS_u : C_i \rightarrow \{0, 1\}$, $PartOfS_u = 1$ if $C_i \in S_u$ and $PartOfS_u = 0$ if $C_i \notin S_u$, where S_u represents a strategy and is defined further on. The set of attributes of a generic case for eXpresser

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

Category	Name	Attribute	Possible Values
Patterns properties	Colour	α_{i_1}	Red/Green/Blue/Yellow
	Width type	α_{i_2}	Number (n)/Icon variable (iv)/numeric expression (n_exp)/icon variable(s) expression (iv_exp)
	Height type	α_{i_3}	n /iv /n_exp /iv_exp
	⋮	⋮	⋮
	Colour type	α_{i_v}	n /iv /n_exp /iv_exp
	Width value	$\alpha_{i_{v+1}}$	numeric value
	Height value	$\alpha_{i_{v+2}}$	numeric value
	⋮	⋮	⋮
	Colour value	α_{i_w}	n /iv /n_exp /iv_exp
Group flag	isGroup	$\alpha_{i_{w+1}}$	0/1
Part of Strategy	$PartOfS_1$	$\alpha_{i_{w+2}}$	0/1
	$PartOfS_2$	$\alpha_{i_{w+3}}$	0/1
	⋮	⋮	⋮
	$PartOfS_r$	α_{i_N}	0/1

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 ($\alpha_{i_j}, j = \overline{w+1, N}$).

The set of *relations between attributes* of the current case and attributes of other cases 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 (F_i) of the current case. 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 $DEP : \alpha_{i_k} \rightarrow \alpha_{j_l}$ is defined 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 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); (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 that could take 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.). A case is considered *specific* when it does not have dependency relations and is considered *general* when it has at least one dependency relation.

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 ($p \leq 2$).

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

To illustrate how the knowledge representation and the identification mechanism operates, a task called "footpath", typical in the UK curriculum, is used, which requires

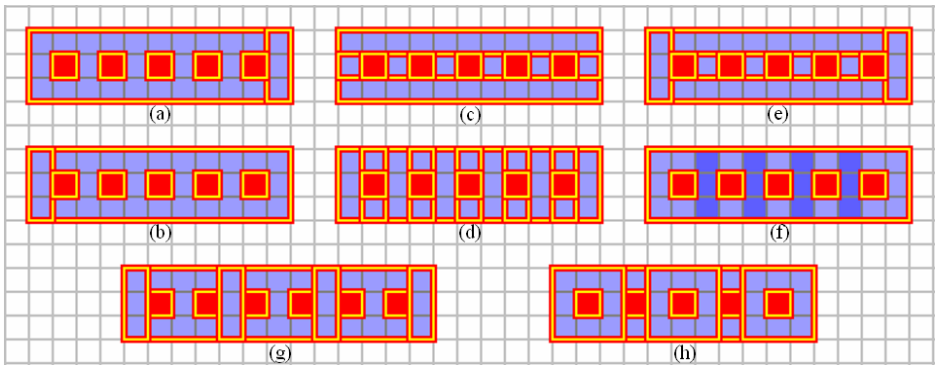

Figure 2. Strategies for footpath task: (a) forward C; (b) backward C; (c) HParallel (horizontally parallel); (d) VParallel (vertically parallel); (e) H&VParallel (horizontally and vertically parallel); (f) Squares; (g) ISeries (works only for even numbers for the red, darker-coloured, tiles) and (h) SquareSeries (works only for odd numbers for red, darker-coloured, tiles).

to find the number of tiles that surround a pattern like the red one displayed in Figure 1. There are severals strategies for constructing the surrounding for that pattern; among them some are more desirable (displayed in Figure 2) than others, in the sense that they facilitate generalisation.

Table 2. S_u definition for each step of the 'Forward C' strategy.

S_u	$N_u\,(C)$	$N_u\,(RA)$	$N_u\,(RC)$
Step 1	C_1	-	-
Step 2	$C_1, C_2, C_3,$	-	$Prev(C_{i+1}) = C_i$ for $i = \overline{1,5}$
	C_4, C_5, C_6		$Next(C_i) = C_{i+1}$ for $i = \overline{1,5}$
Step 3	C_1, C_2	-	$Next(C_1) = C_2$
			$Prev(C_2) = C_1$
Step 4	C_1, C_2	$\alpha_{2_3} = \alpha_{1_3}$	$Next(C_1) = C_2$
		$\alpha_{2_3} = DEP(\alpha_{1_3})$	$Prev(C_2) = C_1$
Step 5	C_1, C_2, C_3	$\alpha_{2_3} = \alpha_{1_3}$	$Next(C_i) = C_{i+1}$ for $i = \overline{1,2}$
		$\alpha_{2_3} = DEP(\alpha_{1_3})$	$Prev(C_{i+1}) = C_i$ for $i = \overline{1,2}$
Step 6	C_1, C_2, C_3	$\alpha_{2_3} = \alpha_{1_3}$	$Next(C_i) = C_{i+1}$ for $i = \overline{1,2}$
		$\alpha_{2_3} = DEP(\alpha_{1_3})$	$Prev(C_{i+1}) = C_i$ for $i = \overline{1,2}$

Besides multiple possible constructions, there are several ways of reaching the same construction. A possible trajectory for the 'forward C' strategy is illustrated in Figure 3.

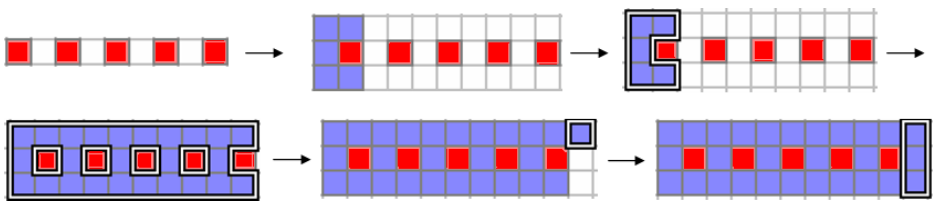

Figure 3. Possible steps for 'forward C' strategy.

The learner may start with the footpath (the red tiles) and then build a group of five blue tiles around the leftmost red tile having the form of a 'C'; this group is iterated five times (the number of red tiles). Finally, a vertical pattern of three tiles is added at the right of the footpath. The details for most steps of this particular strategy are displayed in Table 2.

The first step includes only one case: the red tiles pattern. After some intermediate steps, not illustrated here, the second step includes 6 cases, i.e. the red pattern and five single blue tiles, which are in a given order as expressed by the set of $Prev$ and $Next$ relations. In the third step, the 5 blue tiles are grouped in one pattern which now becomes C_2; consequently, at this point there are 2 successive cases. In the forth step, the second case, i.e the group of 5 blue tiles, is repeated 5 times (the number of red tiles), so now there is also a value and a dependency relation. In the fifth step a new blue tile is added, becoming C_3 and in the sixth step this tile is iterated 3 times; in the last two steps, the relations between attributes and between cases are the same as in step 4.

Strategy identification is based on scoring elements of the strategy followed by the learner according to the similarity of their attributes and their relations to strategies previously adopted and stored. Thus, to identify components of a strategy, four similarity measures are defined: (a) Numeric attributes - Euclidean distance: $D_{IR} = \sqrt{\sum_{j=v+1}^{w} \times (\alpha_{I_j} - \alpha_{R_j})^2}$ (I stands for the pattern the learner is constructing and R stands for patterns compared or recalled from the ones stored); (b) Variables: $V_{IR} = \sum_{j=1}^{v} g(\alpha_{I_j}, \alpha_{R_j})/v$, where g is defined as: $g(\alpha_{I_j}, \alpha_{R_j}) = 1$ if $\alpha_{I_j} = \alpha_{R_j}$ and $g(\alpha_{I_j}, \alpha_{R_j}) = 0$ if $\alpha_{I_j} \neq \alpha_{R_j}$. (c) Relations between attributes - Jaccard's coefficient: $A_{IR} = \frac{|RA_I \cap RA_R|}{|RA_I \cup RA_R|}$. A_{IR} is the number of relations between attributes that patterns I and R have in common divided by the total number of relations between attributes of the two cases; (d) Relations between cases - Jaccard's coefficient: $B_{IR} = \frac{|RC_I \cap RC_R|}{|RC_I \cup RC_R|}$ (same as relations between attributes).

To identify the closest strategy to the one followed by a learner during construction, cumulative similarity measures are used for each of the four similarity types: (a) Numeric attributes - as this metric has a reversed meaning compared to the other ones, i.e. a smaller number means a greater similarity, the following function is used to bring it to the same meaning as the other three similarity measures, i.e. a greater number means greater similarity: $F_1 = z/\sum_{i=1}^{z} D_{I_iR_i}$ if $\sum_{i=1}^{z} D_{I_iR_i} \neq 0$ and $F_1 = z$ if $\sum_{i=1}^{z} D_{I_iR_i} = 0$, where z represents the minimum number of cases among the two compared strategies; (b) Variables: $F_2 = (\sum_{i=1}^{z} V_{I_iR_i})/z$; (c) Relations between attributes: $F_3 = (\sum_{i=1}^{z} A_{I_iR_i})/z$; (d) Relations between cases: $F_4 = (\sum_{i=1}^{z} B_{I_iR_i})/z$. The similarity between the current strategy and a stored strategy is defined as the sum of these four measures after they are normalised as explained below.

As the four similarity metrics have different ranges, normalisation is applied to have a common measurement scale, like $[0, 1]$. This is done using linear scaling to unit range [1] by applying the following function: $\overline{x} = \frac{x-l}{u-l}$, where x is the value to be normalised, l is the lower bound and u is the upper bound for that particular value: (a) Numeric attributes: the range of F_1 is $[0, z]$; therefore the normalisation function is: $\overline{F_1} = F_1/z$; (b) Variables - the range of F_2 is $[0, 1]$, so no normalisation is needed; (c) Relations between attributes: the range of F_3 is $[0, 0.5]$; therefore the normalisation function is: $\overline{F_3} = 2F_3$; (d) Relations between cases - same as the previous: $\overline{F_4} = 2F_4$.

Summarising, we propose a mechanism that identifies what strategy the learner is following by comparing what the learner is doing with each of the stored strategies.

The strategy that comes up as most similar is the one followed by the learner (totally or partly). The following section gives some scenarios that illustrate this process.

3. Examples of Strategies' Identification

To illustrate how strategy identification is performed using the similarity measures presented in Section 2, scenarios are used. The scenarios presented here cover some of the most complex situations encountered in the trials with pupils. They have been designed based on pedagogical principles and they deal with partial constructions (frequently observed in classroom), as well as with complete ones using a particular strategy or a combination of strategies. A summary of the scenarios, including their pedagogical rational is presented in Table 3.

Scenario 1. Detecting partial constructions. The pedagogical rational is to guide learners when they are stuck based on their partial construction. For most strategies the comparison with the complete sequences that specify particular strategies, if available, is sufficient to return the most similar strategy. For example, when the partial strategy displayed in Figure 4a is compared with all the strategies from Figure 2, the maximum similarity corresponds to the 'HParallel' strategy with a value of 2.83; the second best calculated similarity (1.99) corresponds to the 'H&VParallel' strategy. Some strategies require intermediate steps that are not reflected in the final construction; one such strategy is 'forward C', as illustrated in Figure 3 and Table 2. To recognize a partial intermediate construction like the one illustrated in Figure 4b, intermediate steps need to be stored in a knowledge base. Consequently, when comparing the strategy displayed in Figure 4b with the partial and complete strategies in the knowledge base, the maximum similarity will correspond to the intermediate *Step 3* from Table 2 (except that the number of red tiles is 3 instead of 5); the similarity value after normalisation is 3.

Scenario 2. Detecting whether the learners are working with the specific or the general. The pedagogical rational is to identify what approach suits the learner best: specific-to-general or general-to-specific. Some learners start from the general, while others (the majority) start from a specific situation. Identifying the one they are working with is necessary to be able to guide them further. Thus, if they are working with the specific the transition to the general is not always straightforward and requires a 'mental jump' from the learners. This may be the key point where personalised feedback is required. Sometimes this 'jump' is made, but the learners are not sure about it and are reluctant to proceed without feedback. The construction at such a point will probably have one element that is general whilst the other ones would be still specific; for example in the 'HParallel' strategy, the middle row of red tiles is general and the top and bottom ones are specific. This is reflected in the similarity measures by the fact that there is a bigger similarity with the specific strategy (3.88) rather than with the general one (3.78).

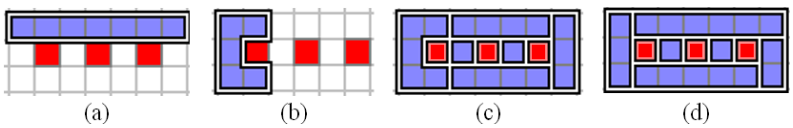

Figure 4. Constructions for: (a) Scenario 1: detecting partial construction; (b) Scenario 1: detecting intermediate partial construction; (c) Scenario 3: combination of strategies; (d) Scenario 3: lack of symmetry.

Table 3. Scenarios summary.

Scenario	Pedagogical rational	Construction	Top matching strategies
Detecting partial constructions	Guiding before end of construction	Figure 4a	HParallel: 2.83
			H&VParallel: 1.99
		Figure 4b	Partial forward C: 3
Working with the specific/general	Identify best approach	Figure 2c	HParallel specific: 3.88
			HParallel general: 3.78
Mixed strategies	Guide the learner towards one strategy	Figure 4c	H&VParallel: 2.12
			Forward C: 1.98
	Symmetry as a generalisation principle	Figure 4d	HParallel: 2.22
			H&VParallel: 1.96

Scenario 3. Mixed strategies. The pedagogical rational is twofold: (a) to guide the learner towards a strategy that is reflected in their construction and (b) to point out that symmetry is desirable. The image in Figure 4c combines two strategies: 'H&VParallel' and 'forward C'. The similarity metrics adequately identify them as the best available matches: 2.12 and 1.98, respectively. If the learner has difficulties in generalising from their construction, they could be guided to the best matching strategy or could be given a choice between the two. Some learners are able to generalise even if their construction is not symmetrical, while other are not. For the former, the usefulness of symmetry can be shown to them by displaying the algebraic-like form of the most similar symmetrical situations as in Table 4; for the later, they should be guided to a symmetrical solution.

Table 4. Algebraic-like rules for symmetric and non-symmetric constructions.

Construction	Rule
Figure 4d	$3 + 2 + 2 * red + (2 * red - 1) + (red - 1)$
Figure 2c	$2 * 3 + 2 * (2 * red - 1) + (red - 1)$
Figure 2e	$2 * (2 * red - 1) + (red + 1)$

The construction in Figure 4d has the most complicated rule, as it needs a different expression for each part of the construction, while the most similar symmetric constructions, i.e. Figure 2c (2.22) and Figure 2e (1.96) are simpler owing to their symmetry.

4. Discussion and Conclusion

In this paper an approach for strategy identification in the domain of mathematical generalisation was presented. Details of knowledge representation and strategy identification were provided and some examples of pedagogically-driven scenarios were given.

The approach presented in this paper was chosen because of the nature of learning in ELEs and because of the characteristics of the domain. Classic approaches to learner modelling monitor the level of mastery of certain concepts of the domain mainly by assessing how well the learners are doing in solving problems. In ELEs, the focus is on exploration of the task and construction of knowledge. Therefore, the focus is on the interactions of the learners with the system rather than on the outcomes of their problem-

solving. Our approach allows identification of strategies followed by the learners *while* solving a task, enabling the possibility of personalised feedback or other courses of actions, like informing the teachers or identifying a peer that is following the same strategy and would be able to help.

Although we illustrated our approach using only one task, the approach is general and works for other tasks, e.g. [2], provided that knowledge of the possible strategies in solving the task is available.

Our future work investigates possibilities for intelligent support [9] upon recognition of strategies: personalised feedback to learners, inform teachers of the learners' strategies, pairing learners for collaboration to discuss similarities and differences between their approaches and the equivalence on the derived rules.

Acknowledgements

This work is partially funded by ESRC, UK, as part of the MiGen Project (TLRP e-Learning Phase-II, RES-139-25-0381).

References

[1] S. Aksoy, R.M. Haralick: Feature normalisation and likelihood-based similarity measures for image retrieval, *Pattern Recognition Letters* **22** (2001), 563-582.

[2] M. Cocea, G. Magoulas: Combining Intelligent Methods for Learner Modelling in Exploratory Learning Environments. *Proceedings of the 1st International Workshop on Combinations of Intelligent Methods and Applications (CIMA 2008)*, in conjunction with the 18th European Conference on Artificial Intelligence (ECAI-08) (2008), 13-18.

[3] M. Cocea, G. Magoulas: Identifying strategies in user's exploratory learning behaviour for mathematical generalisation, *Proceedings of The 14th International Conference on Artificial Intelligence in Education* (2009).

[4] T. de Jong and W.R. van Joolingen: Scientific discovery learning with computer simulations of conceptual domains, *Review of Educational Research*, **68** (1998), 179-202.

[5] J. Kaput: Technology and Mathematics education. In D. Grouws (ed.) *Handbook of Research on Mathematics Teaching and Learning*, New York: Macmillan (1992) 515-556.

[6] J.L. Kolodner: *Case-Based Reasoning*, Morgan Kaufmann Publishers, Inc., 2nd edn. (1993).

[7] J. Mason: Generalisation and algebra: Exploiting children's powers. In L.Haggarty, *Aspects of Teaching Secondary Mathematics: Perspectives on Practice*, Routledge Falmer and the Open University (2002), 105-120.

[8] R. Noss, C. Hoyles, E. Geraniou, S. Gutierrez-Santos, M. Mavrikis, D. Pearce: Broadening the sense of 'dynamic': an intelligent system to support students' mathematical generalisation. Submitted to *The International Journal on Mathematics Education* (2008).

[9] D. Pearce, E. Geraniou, M. Mavrikis, S. Gutierrez-Santos, K. Kahn: Using Pattern Construction and Analysis in an Exploratory Learning Environment for Understanding Mathematical Generalisation: The Potential for Intelligent Support. In S. Gutierrez-Santos, M. Mavrikis (eds.), *Proceedings of the 1st International Workshop on Intelligent Support for Exploratory Environments*, EC-TEL'08 (2008).