Expert Systems with Applications 37 (2010) 140–148

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Automatic system for the detection and analysis of errors to support the personalized feedback

Carina S. Gonzalez *, Davinia Guerra, Hilda Sanabria, Lorenzo Moreno, Maria Aurelia Noda, Alicia Bruno

School of Computer Science, University of La Laguna, CP 38200 La Laguna, S/C de Tenerife, Spain

ARTICLE INFO

Keywords:
Automatic diagnosis
Datamining
Errors
Diversity
Adapted aids

ABSTRACT

The study of errors in learning and the search for patterns to explain their causes have always been of great interest to researchers and educators alike. Mistakes are a constant in students' solutions to mathematical problems and are inseparable from the learning process. It is essential, then, to diagnose and address the mistakes made by students so as to allow them to reflect on their errors and adjust their knowledge. To this end, we have created a system that tracks all the actions carried out by a student when solving a mathematical algorithm, not just the final results, and which is capable of diagnosing the faults and possible causes. It can also recommend the actions to be taken based on the individual difficulties encountered. In short, we have created a personalized teaching system whose features could be particularly useful for special-needs students, such as those with Down syndrome. This paper explains the error detection modules in the addition, subtraction and error-adapted assistance algorithms.

This work is part of a multidisciplinary research effort financed by R&D project called "Divermates", of the Ministry of Labor and Social Affairs, and involving personnel from the Computer Engineering and Mathematics and Fine Arts Education Departments of the University of La Laguna, as well as professionals from the Tenerife Trisomic 21 Association (ATT21).

1. Problem description

While arithmetic operations are present in a wide variety of everyday situations, most students, regardless of their personal characteristics, have problems properly understanding and applying them. Specifically, students with Down syndrome experience special difficulties when confronting abstract concepts. The comprehension and handling of algorithms requires a series of factors (intellectual level, graphomotor skills, attention, memory, and so on) in which a knowledge of the number concept and of the decimal numbering system plays a key role.

In a 1987 study by Buckley and Sacks (1987) on an adolescent population of 90 subjects with Down syndrome, it was noted that only 18% were able to recite over 20 numbers and 50% were able to perform a simple addition. Only a few, however, were able to multiply or divide.

Our research focuses on the simplest arithmetic operations: addition and subtraction. Taking these two elements as our main focus, and using previous research on students without disabilities and errors as source of learning in Mathematics (Baroody, 1988; Bruno et al., 2003; Dikson, Brown, & Gibson, 1991; Fernández, Llopis, & Pablo, 1991; Horacek & Wolska, 2006; Jiménez & Girando, 1993; Luseño, 1993; Mathan & Koedinger, 2003; Maza, 1989; Melis, 2004; Melis & Siekmann, 2004; Renkl, 2002; Zinn, 2006) as starting point of our proposal. Taking into account these previous works, we propose the following initial classification of possible mistakes students can make as shown in Table 1.

Starting with this initial classification, we have designed an automatic system for detecting the mistakes made when working with the algorithms in question. Our stated goal is to identify the potential causes which result in the mistakes detected, as well as to supply assistance depending on the specific needs.

To achieve this aim, we have designed a system that is capable of detecting arithmetic mistakes in addition and subtraction algorithms, of classifying them, of inferring the possible causes and of offering hints or helping depending on the mistake. The system is described in the sections that follow.

2. Error detection: analysis, treatment and presentation of results

Once a detailed list of all the possible mistakes that can be made, and which the system must therefore be able to detect, is compiled, the next step is the development of an application that is capable of analyzing the information gathered as a result of

* Corresponding author. Tel.: +34 922 318284; fax: +34 922 318288.
E-mail address: cjgonza@ull.es (C.S. Gonzalez)

0957-4174/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2009.05.027

Table 1
Potential mistakes in addition and subtraction algorithms.

Mistake categories and types
1. Graphomotor and perception problems
[1.1] *COMMON:* Confusing numbers: 3–5, 4–7, 6–9, 12–21...
[1.2] *COMMON:* Alternatively adding and subtracting
[1.3] *COMMON:* Starting operation from left
2. Number and unit alignment errors
[2.1] *COMMON:* Improper placement of ones and tens (difficulties aligning, changing unit order)
[2.2] *COMMON:* Adding/subtracting different order units
3. Carry/borrow errors
[3.1] *COMMON:* Forgetting to carry/borrow
[3.2] *ADDITION:* Writing the incomplete intermediate partial results
[3.3] *ADDITION:* Working as if dealing with individual numbers
[3.4] *ADDITION:* Failing to write the units of the last column
[3.5] *ADDITION:* Mistakes when regrouping
[3.6] *SUBTRACTION:* Always borrowing
[3.7] *SUBTRACTION:* Always subtracting smaller number from larger.
4. Confusing the role of the zero in algorithms which have a zero as one of the digits
[4.1] *COMMON:* Associating the zero with the partial result
[4.2] *COMMON:* $N + 0 = 0$ $N - 0 = 0$
[4.3] *COMMON:* Identifying 0 with 10
[4.2] *SUBTRACTION:* Subtracting zero from the number associated with the subtrahend.
[4.3] *SUBTRACTION:* Putting the value of the subtrahend in the result when the zero is in the minuend
5. Total ignorance of the algorithm
[5.1] *ADDITION:* Ignoring the positional value of the digits and adding all the numbers
6. Ignorance of the meaning of the operation
[6.1] *COMMON:* Subtracting when they have to add and vice versa
7. Made-up numerical facts
[7.1] *COMMON:* Committing errors with certain numerical facts
8. Other
[8.1]*COMMON:* Giving up

the interaction between the users and the platform or, put another way, the results of the mathematical algorithms carried out by the students.

2.1. Algorithm design

One of the requirements of our algorithm is that the response times match those of a real-time system as much as possible, since the error detection algorithm must analyze an unknown, but potentially very high, quantity of data in a period of time that does not represent a bottleneck for the system as a whole.

With this requirement in mind, we designed an algorithm that was capable of differentiating among all potential errors, excluding an analysis of those made impossible by the nature of the algorithm at hand (Abbott, 2006; Duda & Hart, 1973; El-Kechaï & Després, 2006; Hensler & Beck, 2006; Mico, Oncina, & Vidal, 1994; Nava, Livne, Livne, & Wight, 2007; Ramasubramanian & Paliwal, 1990). For example, in an operation involving single digits, errors of the "starts addition from the left" or "adds units of a given order with units of a different order in the other summand" variety would never be detected. By considering these mutually exclusive cases, the algorithm, although still bound by the theoretical time complexity of its task, does show improved efficiency in practical examples.

As a result, we designed the following addition and subtraction error detection algorithm as shown in Fig. 1.

2.2. Handling the errors detected

An analysis of the data obtained from the user's interaction with the platform via the algorithm shown in Fig. 1 returns the set of mistakes made by the student in the corrected exercises. Contrary to what one may imagine, these values do not yield the expected result for a variety of reasons:

(a) A markedly low incidence of a mistake may reflect an occasional slip, and should not be considered essential when educating the student on his mistakes, as this would result in teaching an already known fact. While the repetition of a known fact may seem acceptable at first, the recurrence of such an event, especially considering the population that is the subject of our study, could discourage the student.
(b) The actual number of times a certain mistake is made is meaningless in and of itself. It becomes more meaningful, for example, as the total number of exercises increases. This is why a more useful piece of data is that which reflects the percentage of each error's significant incidence rate.
(c) In addition, it is a well-known fact that error types within a group of errors tend to cluster; that is, taking the table of potential errors as an example (Table 1), if a student makes a mistake of the "confusing the role of the 0" type, it is easy and likely for him to make one of the other mistakes in that grouping. That is why it is useful to know each student's percentage not just for individual errors, but also within an error classification.

If we are to solve the first problem, we must rule out all those mistakes which may arise from occasional absentmindedness and not from a lack of knowledge of the algorithm. To do this, we have to define a threshold that restricts what is and is not within the limits of what should be corrected. Despite the ample bibliography on errors that exists in the world of mathematics education, including the papers in Del Puerto, Minnaard, and Seminara (2004) and Blando, Kelly, Schneider, and Sleeman (1989), very few address those errors arising from sporadic carelessness. Given this situation, we obtained the desired value by using empirical methods which analyzed the information provided up to that point by the very subject being tested. That is why a study was conducted which sought to identify the incidence of mistakes in individual students who were known not to commit said mistakes on a regular basis.

Once the information was analyzed, the following conclusions were reached:

- The desired threshold must be generic and applicable to any instance in which the correct values are obtained. That is why the value should result from the average of the thresholds detected for each individual student.
- The applicable threshold cannot be a constant, but must instead be determined by the number of exercises solved by the student. The threshold, then, should be obtained as a percentage. An explanation of this situation is shown in Table 2.
- The threshold should also be determined by the maximum number of error types the system is able to differentiate. The information provided by an X% incidence of a certain error type is different depending on whether there are just two error types or 200.
- Given the above consideration, there is an obvious need to calculate a different threshold for each type of operation, since the number of error types can change for each algorithm.
- The above considerations will be repeated so as to calculate threshold values for error clusters, taking into account the number of recognizable clusters.

Considering all of the above, along with the knowledge obtained from an analysis of the data, we have defined the formula that will allow the threshold values to be calculated as shown in Fig. 2:

where:

op = type of operation (addition or subtraction)
num_op = number of "op" type exercises solved by the student