



**HAL**  
open science

## **AMBRE-add: An ITS to Teach Solving Arithmetic Word Problems**

Sandra Nogry, Nathalie Guin, Stéphanie Jean-Daubias

► **To cite this version:**

Sandra Nogry, Nathalie Guin, Stéphanie Jean-Daubias. AMBRE-add: An ITS to Teach Solving Arithmetic Word Problems. *Technology, Instruction, Cognition and Learning*, 2008, 1, 6, pp.53-61. hal-01500376

**HAL Id: hal-01500376**

**<https://hal.science/hal-01500376v1>**

Submitted on 4 Sep 2017

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# **AMBRE-add: an ITS to teach solving arithmetic word problems**

Sandra Nogry, Nathalie Guin-Duclosson, and Stéphanie Jean-Daubias

LIRIS Lab., Univ. Lyon 1

Nautibus, 43 Boulevard du 11 novembre 1918, 69622 Villeurbanne Cedex, France

**Abstract:** This paper describes the design and evaluation of AMBRE-add. This ITS was designed to teach problem classes and the solving techniques associated to those problem classes in the domain of arithmetic. AMBRE-add uses case-based reasoning as a learning strategy: it presents solved problems in each class to learners, and then guides them to use the case-based reasoning steps to solve new problems. In each step, the system diagnoses the learner's answers and provides help and explanation messages. We evaluated this system in the classroom on three classes of eight-year-old pupils who used AMBRE-add or a "control" system for six sessions, and on one class of nine-year-old pupils who used the system for four sessions.

**Keywords:** Intelligent Tutoring Systems, Case-Based Reasoning, problem solving, cognitive psychology, ITS evaluation.

## **INTRODUCTION**

This paper describes the design and the evaluation of AMBRE-add, an Intelligent Tutoring System (ITS) that teaches problem classes in arithmetic. This ITS was designed in the framework of the AMBRE project. This project aims at showing that Case-Based Reasoning (CBR) can be proposed as a learning strategy by an ITS. The AMBRE project proposes to design ITSs in several domains that will present examples to learners and then guide them in solving a new problem following the Case-Based Reasoning steps. These systems should facilitate the acquisition of problem classes and the solving techniques associated with each class. We applied this principle to the design of the

AMBRE-add ITS, intended to help seven-year-old to nine-year-old children learn how to solve arithmetic word problems. This ITS is based on a knowledge-based system which provides the pupil with help and explanations regarding his or her answers.

In this paper, we present the AMBRE-add ITS. First, we describe what we want to teach with AMBRE-add. Then, we present its principle, which is to use case-based reasoning to guide the learner. Next, we describe the AMBRE-add ITS, by describing its interface, the interactions it proposes, and the knowledge-based system it uses to diagnose pupils' answers and to provide help and explanations. Lastly, we outline how we evaluated AMBRE-add in a real-word setting.

## **1 AN ITS TO TEACH ARITHMETIC PROBLEM CLASSES**

### **1.1 The AMBRE Project: Designing an ITS to Help Students Learn Problem Classes**

Research on didactics of mathematics that has studied experienced problem solvers has shown that “by the time the mathematician has finished reading the problem, the problem will have been characterized and the standard procedure for solving such problem will have been invoked. [...] The response may be a pre-packaged solution to the entire problem, if the problem is recognized as being a standard type.” (Schoenfeld 1985, p. 20). However, in some domains, the terms defining problem classes and problem-solving techniques are not used by teachers and are not known by novices. So novices have difficulty learning these problem classes.

The objective of the AMBRE project is to design computer environments using case-based reasoning to teach problem classes and solution techniques to be applied to each class (Schoenfeld 1985, Rogalski 1994). Rogalski (1994) suggested that in order to design such an intelligent tutor, the knowledge included in the tutor should be similar to the knowledge the learner has to acquire (Rogalski 1994). So to design such an ITS, it is first necessary to know the problem classes in the domain and the associated techniques. For arithmetic, such a classification already exists.

## 1.2 The Domain of Arithmetic Word Problems

Arithmetic problems constitute one of the most widely studied domains in research on didactics of mathematics, cognitive psychology, and linguistics (see Fayol 1990, chap. 6 for a review).

Arithmetic problems describe a concrete situation such as a game of marbles: “Brad went to school with marbles. He gave thirteen of the marbles to Luke during the day. In the evening, he had fifty-six left. How many marbles did he have when he went to school?”

For this type of problem, a solution involving several stages is expected:

- Describe the problem using a “fill in the gap” operation:  $? - 13 = 56$
- Write out how you perform this calculation<sup>1</sup>:  $56 + 13 = ?$
- Do the calculation: 69
- Write out the answer to the question: Brad had 69 marbles.

Riley, Greeno, and Heller (1983) suggested a classification for addition and subtraction problems based on their underlying semantic relations that draw a distinction between three main categories of problems: combine, change, and compare (Riley *et al.* 1983). *Change problems* are about dynamic situations in which some event changes the value of an initial quantity. *Combine problems* describe a static situation involving two sets that are considered either separately or in combination. *Compare problems* involve two sets that are compared and the difference between them. With each of the three problem categories, further distinctions are made depending on which quantity is the unknown. For example, in change problems the unknown can be either the starting quantity, the change quantity or the result quantity.

Greeno and Riley (1987) showed that the difficulties experienced by young children when solving arithmetic problems are basically due to the fact that they do not manage to correctly represent the situation described in the wording of the problem. What enables older children to effectively solve

---

<sup>1</sup> Possibly, depending on the grade in school.

arithmetic problems is their ability to model these problems (Greeno & Riley 1987). The classification established by Vergnaud (Vergnaud 1982) further specifies Riley, Greeno and Heller's classification. By drawing a distinction between numerical calculation and relational calculation, he proposed an approach that is more effective for modelling arithmetic problems. However, an evaluation of the success rates on these problems showed that the difficulty is not related to the underlying mathematical operation. On the other hand, the problem category (combine, change, or compare) does play a part in making a problem difficult, as does the nature of the unknown element.

This body of research on didactics of mathematics leads to the idea that the domain of arithmetic problems is a good one for trying out an AMBRE ITS. They are problems in which modelling plays a major role, for which a classification has been established, and which are difficult for elementary school pupils. Figure 1 summarizes the problem classes (represented here by diagrams proposed by didactics studies) that we hope to help pupils acquire using the AMBRE-add ITS. This classification is not presented as such to learners. For each problem class, there is a "fill in the blank" operation, which has to be determined in relation to the problem to be solved. Even if we have an explicit classification of the problems and of the techniques at our disposal, and a computer system capable of applying it, we know that it is not necessarily desirable to explicitly present the classification to the learner. In the domain of arithmetic, the terms defining the problem classes and problem-solving techniques are not used in the schools and are not known to learners. Moreover, it seems preferable for the learner to play an active part and thus to build his or her own classification.

Combine			
Change-add			
Change-subtract			
Change-unknown			
Compare			

**Figure 1.** Classification of arithmetic problems treated in AMBRE-add

## 2 LEARNING PROBLEM CLASSES USING CASE-BASED REASONING

To assist the learner in building his or her own classification, we chose to show solved problems and next to encourage the learner to apply analogical reasoning to solve other problems. In this section, we first present analogical problem solving and how it can be facilitated. Then we present the learning strategy proposed by the AMBRE-add ITS: guided problem solving following case-based reasoning steps.

---

<sup>2</sup> The symmetrical diagram can also be used by the learner.

## 2.1 Analogical Problem Solving in Cognitive Psychology

When solving a new problem, people often remember a similar problem they saw earlier and draw an analogy between the two problems. A great deal of research had focused on analogical problem solving (Bassok 1990, Bassok & Holyoak 1989, Gick & Holyoak 1980, 1983, Falkenhainer, Forbus & Gentner 1989, Holyoak 1984, Holyoak & Thagard 1989, Hummel & Holyoak 1997, Keane, Ledgeway & Duff 1994, see Sander 2000 for a review). These studies have demonstrated the utility of analogical reasoning in learning and in acquiring both contextual knowledge and abstract knowledge (e.g. Cummins 1992, Gick & Holyoak 1983, see Didierjean 2001 for a review of the issue) for adults and for children (see Goswami 1992 for a review). Various processes can be implemented to acquire abstract knowledge (see Cauzinille-Marmèche & Didierjean 1999 for a review) such as a similarities detection process or an explanation-based generalization process. Moreover, generalization can also occur when the source problem seen earlier is being adapted to the new problem (the target). However, as Gick and Holyoak (1983) showed, analogical transfer is not always spontaneous and knowledge is not automatically acquired during analogical problem solving. So, although problem solving by analogy has significant advantages, its use as a learning methodology does not guarantee effective learning.

It seems that specific conditions can be more or less favorable to knowledge acquisition. One way to facilitate learning consists in encouraging the implementation of learning processes. Some approaches using already solved problems or presentation formats for problem wording and solving seem to play a positive role in learning: For example, an **explanation-based generalization process** may be facilitated by asking pupils to provide their own explanations of solved problems (Atkinson, Derry, Renkl & Worthman 2000, Bielaczyc, Pirolli & Brown 1995, Chi *et al.* 1994, Neuman & Schwarz 1998, Wong *et al.* 2002). In this framework, some ITSs are designed to support learning by triggering self-explanations (Aleven & Koedinger 2002, Conati & Vanlhen, 2000). The use of **processes for detecting similarities** can be facilitated by asking subjects to compare solved

problems (Cummins 1992) or by manipulating the order and similarity of the problems presented. If the problems are isomorphic, surface features that differ are more likely to trigger a similarity-detection process (Bassok 1990, Bassok & Holyoak 1989, Gick & Holyoak 1983, Quillici & Mayer 1996). Conversely, if the problems presented are not isomorphic, surface features that are similar seem to facilitate the detection of similarities (Chen & Mo 2004, Gick & Paterson 1992, Pass & van Merriënboer 1994, Quillici & Mayer, 2002). The **generalization-by-adaptation process** is facilitated by presenting a solved problem followed by a problem to be solved that is very close to the example (Ross & Kennedy 1990, Didierjean 2003).

The AMBRE project guides learners through the case-based reasoning cycle in order to encourage them to apply analogical problem solving and to use generalization processes like detection of similarities and generalization by adaptation.

## **2.2 Using Case-Based Reasoning**

The case-based reasoning (CBR) paradigm (Kolodner 1993) is based on the reuse of solved problems. Initially used in artificial-intelligence research on problem solving, its origins are to be found in cognitive psychology studies on episodic memory (Schank 1982) and analogical reasoning (Gentner 1983). CBR is a computational description of intra-domain analogical reasoning. This problem-solving principle can be described using a set of sequential steps (elaborate, retrieve, adapt, test/revision, store), which are often seen as a cycle (Aamodt & Plaza 1994, Mille 1998). Working from a target problem, the case is elaborated by using general knowledge and by retaining only relevant information. Then a similar case (the source case) is retrieved by searching in the case base, after which the source case solution is adapted in order to obtain a solution for the target problem. The solution obtained is revised, and then the target case is stored in the case base for future reuse.



The CBR paradigm has already been used in various parts of ITSs (e.g. learner model, diagnosis). The closest application to our approach is case-based teaching (Aleven & Ashley 1997, Bélanger, Thibodeau & Aïmeur 1999, Burke & Kass 1996, Capus & Tourigny 2000, Joiron & Leclet 2003, Masterton 1997, Schank & Edelson 1990). Systems based on this teaching strategy present a “close” case to the learner when he or she is having trouble solving a problem, or when facing a problem he or she has never come across before (in a new domain or a new type of problem). The Foresti ITS (Bélanger, Thibodeau & Aïmeur 1999), for example, gives learners problems to solve and then presents problem solutions produced by experts and by a CBR system in order to encourage the learner to compare their solutions with others. In these systems, the cases can be given by an expert, or can contain the solutions proposed by other learners, or ones found by a CBR system. These systems provide several levels of interactivity between the learner and the computer environment (Tourigny & Capus 2000). The learner can ask the system to find a similar example and explain how that case was solved, or the system can suggest a case to help the learner when necessary, as in SPIEL (Burke & Kass 1996).

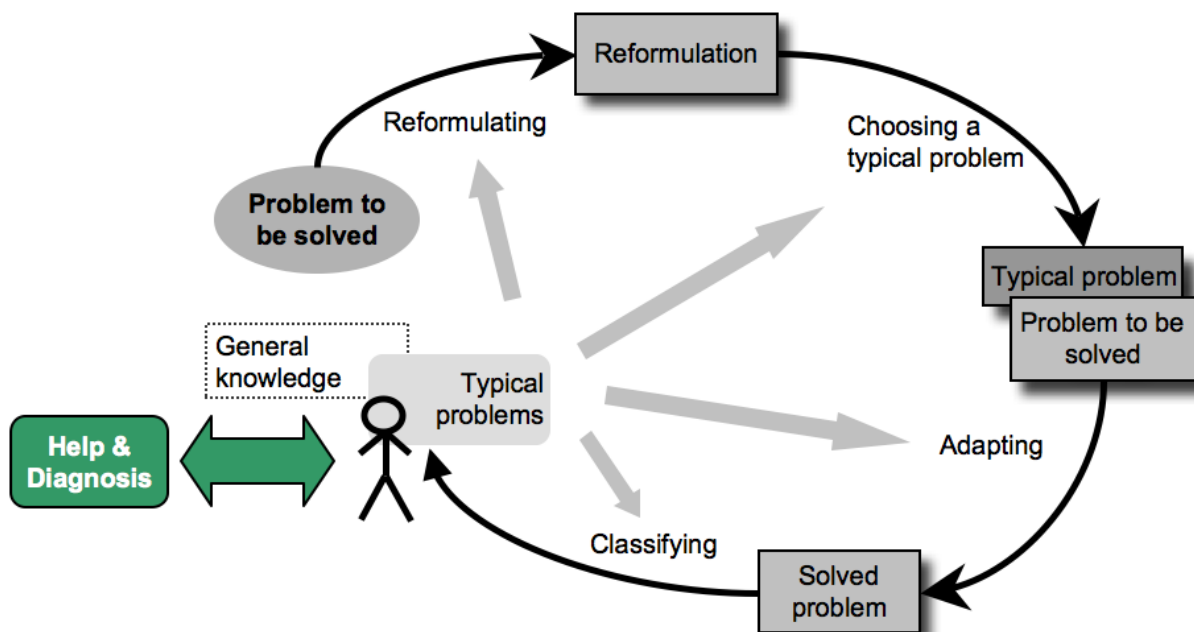
### **2.3 The AMBRE Cycle**

In the AMBRE project, CBR is not used by the system, but is proposed to the learner as a learning strategy. Thus, in order to help learners to acquire problem classes and the techniques associated with those classes, they are first presented with a few typical solved problems (serving as case-base initialization). Then they are assisted in solving new problems. The environment directs the learner toward the steps of the AMBRE cycle, inspired by the CBR cycle (see Figure 2), as follows:

- The learner reformulates the problem in order to identify problem-structure features (elaboration step of the CBR cycle).
- Then, he or she chooses a typical problem (retrieval step of the CBR cycle).
- Next, he or she adapts the typical problem solution to the problem to be solved (adaptation step of the CBR cycle).

- Finally, he or she classifies the new problem (storing step of the CBR cycle).

Although the system guides the learner by proposing the CBR steps, they are done by the learner him/herself. Contrary to the CBR cycle, the revision step is not included in the AMBRE cycle. Instead, a diagnosis of the learner's answers is realised on each step of the cycle: a knowledge-based system diagnoses the learner's answers and provides explanations, if necessary, to help the learner understand his or her mistakes and correct them.



**Figure 2.** The AMBRE cycle

We relied on theoretical studies of analogical problem solving in order to specify the processes that learners might carry out on each step of the AMBRE cycle. For instance, during the analysis of the typical problems, they might execute an explanation-based generalization process; in the choice of a typical problem and classification steps, they might carry out processes to detect similarities; during the adaptation step, they might carry out a generalization by adaptation process. Then, based on cognitive psychology studies, we made various recommendations aimed at facilitating the implementation of these processes in the AMBRE-add ITS (Nogry 2003). The recommendations are related to the content of the instructions, the way of presenting typical problems, the surface features of the typical problems and problems to be solved, and the type of diagnosis.

### **3 AMBRE-ADD ITS**

We applied the AMBRE principle and these recommendations to the domain of arithmetic. AMBRE-add was designed by a multidisciplinary team of researchers including computer scientists, one cognitive psychologist, and teachers. As outlined above, the objective of AMBRE-add is to teach arithmetic problem classes and techniques associated with each class. According to the AMBRE principle, the system first presents typical solved problems and then guides the learner through the solution of new problems following the steps of case-based reasoning (the AMBRE cycle).

This section provides an in-depth presentation of the AMBRE-add ITS. First we describe its interface and the interactions it proposes to the learner. Then we present the knowledge-based system which provides help and diagnosis.

#### **3.1 What does the Learner Have to Do Using AMBRE-add?**

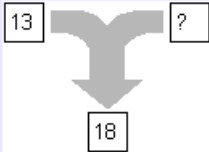
##### *3.1.1 Analysis of Typical Solved Problems*

The first phase of AMBRE-add is dedicated to the presentation of typical solved problems (see Figure 3), which will subsequently be used as reference problems. During each session, the learner has to read and analyze three typical problems from three different problem classes. A sequential presentation of the solution steps was chosen so as to encourage the student to anticipate the solution. In order to facilitate comparisons between the typical problems, they all have the same surface features (they are all marbles problems). Moreover, the system presents a report, on the same screen, of all the typical problems analyzed by that learner since he or she first used the system. After several sessions, the learner will have seen a typical problem that matches each problem class defined in the classification.

**Ambre - showing some examples**

**Wording of the example**  
 Brian and Harry have 18 marbles together. Brian has 13 marbles. How many marbles does Harry have?

**Rewriting of the example**



**Writing of the solution**

The problem is written:  $13 + ? = 18$   
 The calculation is written:  $18 - 13 = ?$   
 The solution is: 5  
**The answer is:** Harry has 5 marbles

The problem is solved by looking at how it was rewritten.

Navigation buttons: < >

**Figure 3.** Presentation of a typical solved problem<sup>3</sup>

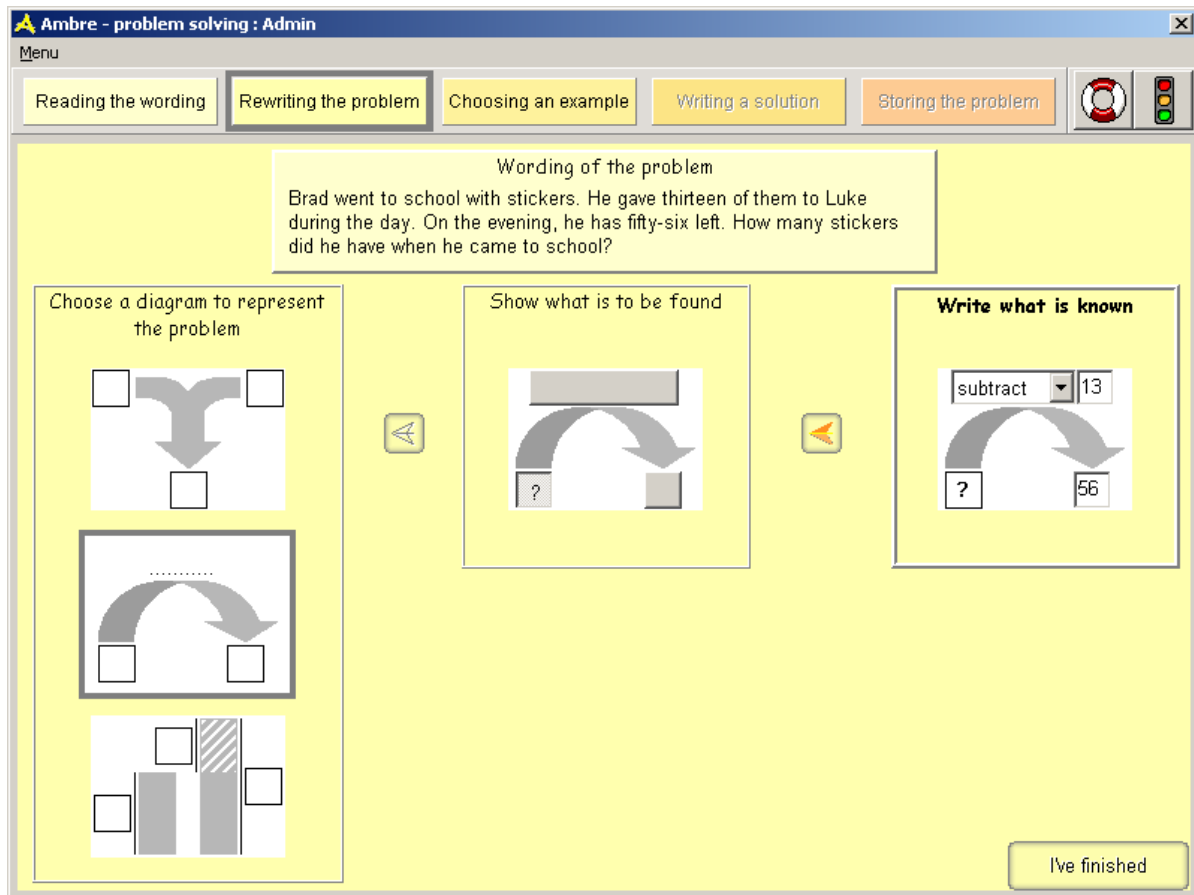
### 3.1.2 Solving New Problems

In the second phase, the learner has to solve several problems belonging to the various classes of typical problems seen earlier. In order to facilitate the adaptation, the surface features of the first problem to be solved in each class are close to the typical problem's surface features. The subsequent problems in that class have different surface features.

For each problem, the learner is guided by the system through the five steps of the AMBRE cycle (see Figure 2). In the first step, the problem statement is provided.

<sup>3</sup> English translation of the French interface.

**Reformulating the problem.** In the second step, the learner has to reformulate the problem by identifying the elements in the problem statement that are relevant to the solution.



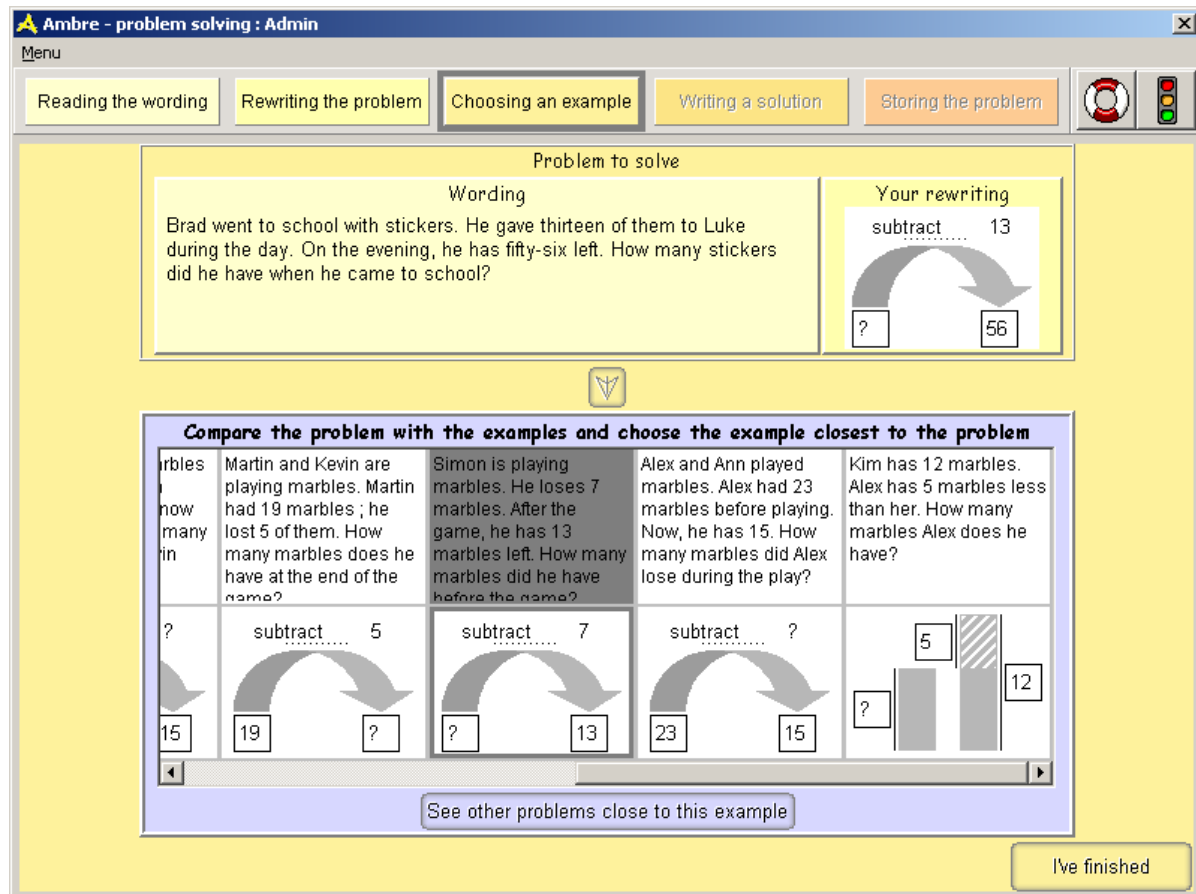
**Figure 4.** Reformulation step<sup>4</sup>

In order to facilitate the reformulation step, the problems are depicted by diagrams which have been proposed and tested in didactics of mathematics studies. These studies have shown that these diagrams can improve performance on problem solving and categorizing (Damm 1992, Fisher 1979, 1993, Vergnaud 1982, Willis & Fuson 1988). For the combine and change categories, we adapted existing diagrams. For the compare category, we could not find any diagram that fulfilled our needs, so we worked within a multi-disciplinary team in order to come up with several, which we tested then on pupils (Jean-Daubias 2004).

<sup>4</sup> The vocabulary used in the interface was adapted to the learner's level by the design team, with the agreement of the teachers.

So, during the reformulation step (see Figure 4), learners have to represent the problem statement using one of the three diagrams. This involves determining what they are looking for (the position of the unknown element) and specifying what they know (the numerical data in the problem). The reformulated problem becomes a reference for the rest of the solving process.

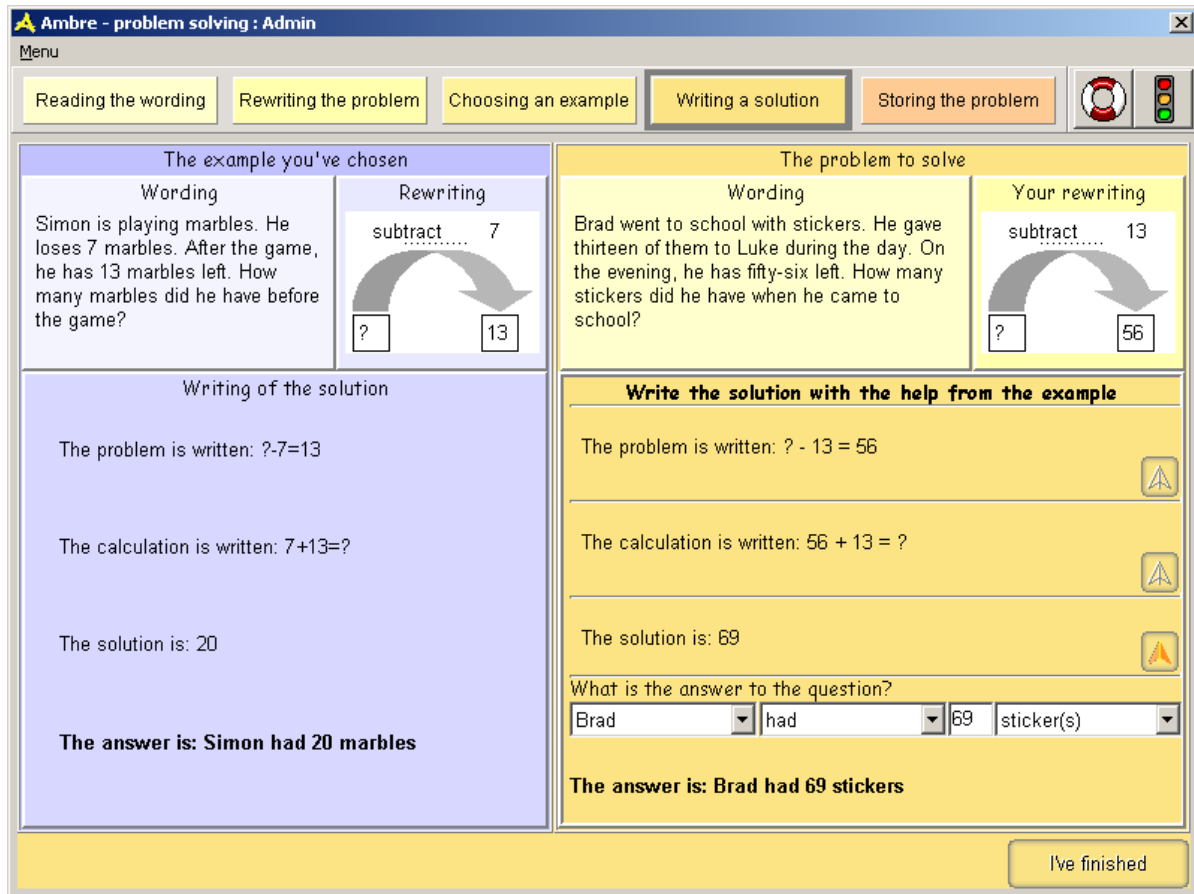
**Choosing a typical problem.** The learner then has to choose the typical problem that is closest to the problem to be solved (see Figure 5). Reminders of the typical problems the learner has already seen (at the beginning of the session and during previous sessions) are provided: the initial problem statements and their rewritten formats are given. In this step, the learner can compare the problem to be solved with typical problems, choose the closest typical problem, and thus implicitly identify the class of the problem to be solved.



**Figure 5.** Choosing a typical problem

**Adapting the typical problem.** In this step, the learner has to write the solution to the problem, drawing inspiration from the solution in the typical problem he or she chose in the previous step (see Figure 6: typical problem on the left-hand side, problem to be solved on the right-hand side). It

is assumed that, through generalization, this adaptation step will lead the learner to identify a problem-solving technique that is suitable for this class of problems. Depending on the school grade, the second part of the solution (writing down how the calculation was performed) may not be requested.



**Figure 6.** Adaptation step

**Classifying the problem.** In the last step, the learner is presented with a report of the problem's solving process: the problem statement, its reformulation, and the solution (see Figure 7). He or she is then asked to put the problem in the same category as one of the typical problems in order to establish groups of problems, each group being represented by a typical problem. As in the typical problem choosing step, the learner should compare the problem to be solved with the typical problems.

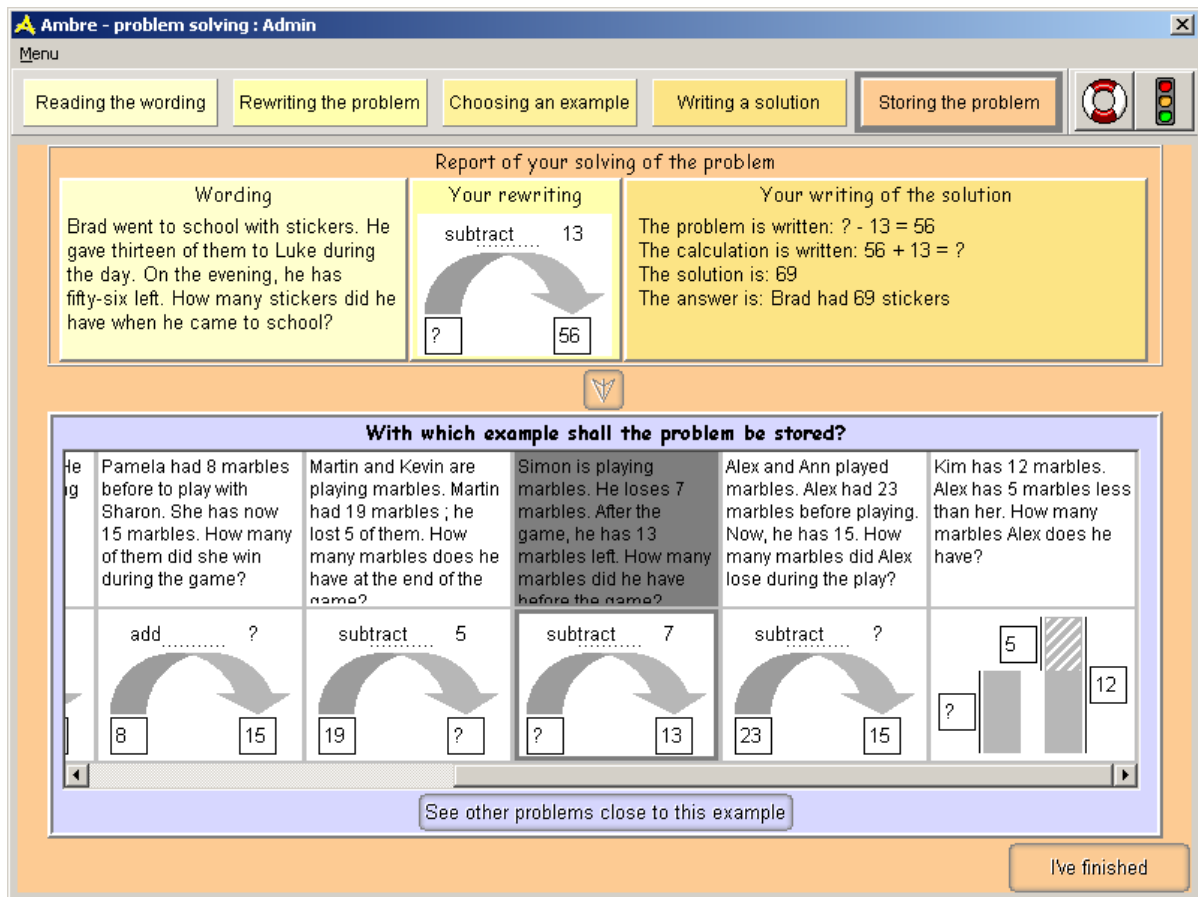


Figure 7. Classification step

### 3.2 Providing Help, Diagnosis and Explanations

At each step in the AMBRE cycle, the system can evaluate the pupil's answers and help him or her revise them by providing explanations of different types. Moreover, learners can ask for a diagnosis of their answers at any time using the "traffic light" button (see Figure 6 - upper right-hand corner), in order find out whether or their answers are correct, and to obtain an explanation if not. The learner can obtain help using the "life-buoy" button (next to the traffic light). Although student modelling is not provided yet by AMBRE-add, the system adapts its explanations to the learner's answers.

This section presents the help, diagnosis, and explanation capabilities of AMBRE-add, and then describes the knowledge-based system that implements these functions.



### 3.2.1 *Help*

When a learner asks for help, messages often provide information about what has to be done. Tools intended to make the learner's task easier are also supplied. For example, to help him or her write the solution to the problem, elements that have the same role in the problem statement, its reformulation and the solution, are displayed in the same color in the typical problem and in the problem to be solved.

### 3.2.2 *Diagnosis*

Learners can ask for a diagnosis of their answers; the system can also provide automatic diagnosis, that is parametrized by the teacher. The default setting triggers an automatic diagnosis at the end of each step. In this case, the learner has to correct the answer in order to move on to the following step. This prevents the learner from solving the problem based on an incorrect reformulation or after choosing a typical problem that is unlike the problem to be solved.

In order to carry out the diagnosis, the system compares the learner's answers to answers generated by a problem solver (described below). If the learner's answers are incorrect, the system generates explanations in order to help him or her correct them.

### 3.2.3 *Explanations*

The system can provide explanations of different types: responses to expected errors, messages generated according to the learner's answers, or graphic elements. Explanatory messages following a diagnosis often **provide a response to an expected error**. For example, when drawing up the sentence for the answer, pupils frequently write a sentence that is in the original problem statement (for example, "Brad has 56 stickers"). In this case, the following explanation is given: "What you wrote is true, but it does not answer the question".

When **a statement that matches the pupil's answer** is generated, it often enables the learner to understand how his or her answer is wrong. This statement-generation capability is used, for example, to explain why the learner's reformulation is incorrect (the reformulation step is presented

in Figure 4) if the error is on the numbers indicated (assuming that the diagram and the position of the unknown element are correct). The learner can then compare the problem statement that is generated to match his or her reformulation with the original problem statement and understand why his or her reformulation is not correct.

**Graphic elements** are used for explanatory messages concerning errors, either because we do not know how to interpret the user's answer, or because it is more useful than a lengthy explanation. In this case, the parts of the answer that are correct are displayed in green, while incorrect elements are shown in red. This red/green tool is especially useful for the step where the learner has to choose a typical problem by comparing it to the problem to solve. In this case, red is used to show what differs between the reformulation of the problem to solve and the reformulation of the chosen typical problem (e.g., the position of the unknown element).

#### *3.2.4 A Knowledge-Based System*

The interface of the AMBRE-add ITS interacts with a knowledge-based system in order to provide learners with help and diagnosis messages concerning their answers. This knowledge-based system (CHAMADE-add) was derived from the CHAMADE architecture. Implemented in Prolog, it models knowledge specific to a given domain as well as pedagogical knowledge. The interface, which is implemented in Delphi, makes use of CHAMADE-add whenever necessary. Text and XML files are used to facilitate communication between the interface and CHAMADE-add. CHAMADE-add can respond to three types of requests: a help request from the learner, a diagnosis request from the learner, and a direct request from the interface (for an automatic diagnosis or in order to display parts of problems and solutions). The CHAMADE knowledge-based system architecture is based on the SYRCLAD architecture (Guin-Duclosson 1999), with which one can design a problem solver based on problem classes and associated solving techniques. The SYRCLAD architecture enables us to make a problem classification explicit, along with reformulation and solving knowledge about those classes. This makes it possible to model the knowledge we are seeking to observe in learners,

and to obtain a problem solver for the domain. For arithmetic problems, we obtained the SYRCLAD-add solver using the classification presented in Figure 1.

To provide help to learners, diagnose their answers and explain their errors, we added the knowledge bases required for the CHAMADE architecture to SYRCLAD-add (Duclosson 2004). These knowledge bases pertain to two major groups of knowledge: knowledge for communicating with the learner, and knowledge for diagnosing the learner's answers and generating help messages and error messages. This is supported by a problem base which contains the typical problems already presented to the learner, as well as the problems he or she has already solved.

There are three types of **knowledge for communicating with the learner**: natural-language knowledge, knowledge for generating a problem statement (or a formulation in natural language of knowledge in the domain), and knowledge that links the interface to the elements it interprets.

There are three types of knowledge for **diagnosing the learner's answers and generating help messages**. The **help related knowledge** enables elements similar to the typical problem and the problem to solve to be highlighted in order to help the learner with adaptation. This knowledge is also used to provide the learner with a reminder of what he or she has to do if he or she asks for help. **Diagnosis-related knowledge** is used to determine how to compare the learner's answers with all correct answers provided by the SYRCLAD-add solver, in order to find out whether the learner's answers are correct or incorrect. **Error-explanation knowledge** is used to define a set of expected wrong answers, and to associate explanatory messages with them. It also serves to define ways of helping the pupil correct an error the system was unable to interpret.

#### 4 EVALUATION OF AMBRE-ADD

In order to evaluate AMBRE-add's usability (Senach 1993) and its impact on learning, we conducted a series of experiments combining traditional system-evaluation techniques developed in the fields

of human-computer interfaces, comparative methods (Shute & Regian 1993), and qualitative methods developed for the human sciences (e.g. observations, interviews, questionnaires) (see Mark & Greer 1993 for a review about ITS evaluation). In this section, we present evaluations of AMBRE-add made in the laboratory and in the classroom with eight-year-old and nine-year-old pupils.

#### **4.1 Evaluation in the Laboratory**

Arithmetic word problems like those presented by AMBRE-add are studied in French schools by pupils between the ages of seven and ten. We chose to conduct the evaluations with eight-year-old pupils because the problems they are given are quite difficult, they can read quite well and they are able to use adaptation process to solve problems.

In the first experiment, five children were observed individually in laboratory. The purpose of this experiment was to identify the difficulties encountered by the children and to examine its usability. The results prompted us to apply modifications to the software. Following observation of difficulties with familiarisation, we replaced the initial presentation display (which was hardly read) by an oral presentation of the software's general principle, in combination with a demonstration showing how the various elements in the interface work. A few vocabulary words that seemed to pose problems for the children were changed. We also changed the appearance of the typical solved problems in order to reduce the amount of information to be processed.

#### **4.2 Evaluation of the Impact of AMBRE-add on Learning in the Classroom**

The second experiment, aimed at evaluating the impact of the AMBRE cycle on learning, was conducted in a real situation of use, at school with three classes of eight-year-old pupils. It consisted in comparing pupils that used AMBRE-add with two control groups that used other systems. During the experiment, the characteristics of this situation of use were taken into account using various qualitative approaches (observation, interviews and a questionnaire).

#### 4.2.1 *Experimental Design*

Three pupil classes from the same school participated in the experiment (N=76). Two of the three teachers of the classes worked in the same way, and pupils came from the same socio-cultural background, so the classes appeared to be basically equivalent.

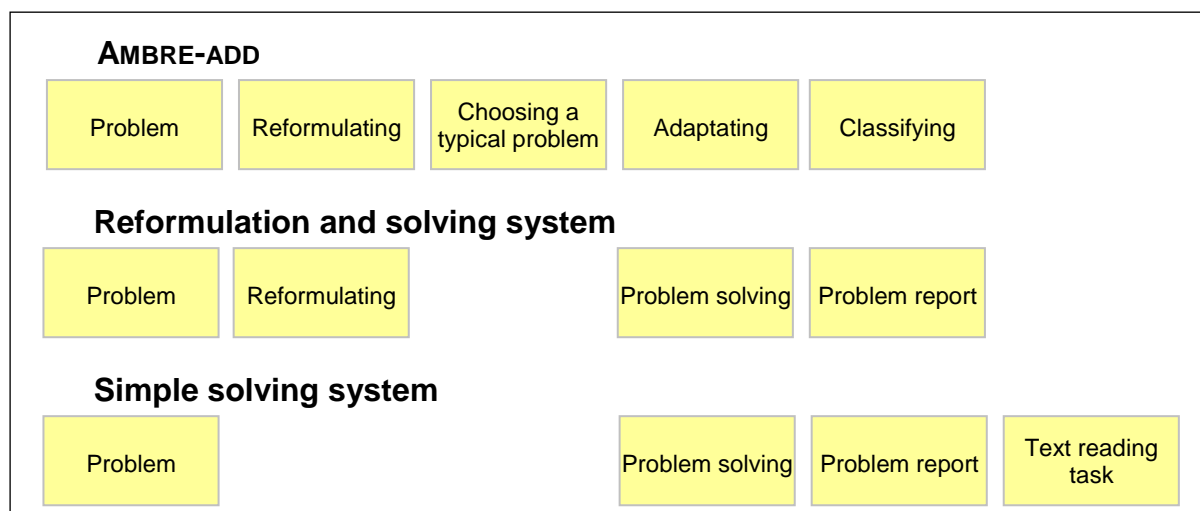
The sessions involving the use of the software took place once a week over a period of six weeks in the school's computer room, each one lasting 45 minutes. In each class, half of the pupils used AMBRE-add (n=39) while the other used one of the two "control" systems. The same problems were presented by the three systems. Three typical solved problems belonging to three problem classes were first presented and then two problems from each of these problem classes had to be solved. No two problems presented in succession belonged to the same class. During the six sessions, eleven problem classes were presented (all classes except the change-unknown class).

In two of the three pupil classes, half of the pupils used the "simple solving" system (n=25) (see Figure 8). This system presented solved problems and then asked for a "simple" solving process of a new problem. In this system, the learners were not guided through the AMBRE cycle. They did not have to reformulate the problem or choose a typical problem. They only had to find the solution and then read the problem report. Because this system had fewer steps than AMBRE-add, learners had to perform another task after problem solving. This task consisted in reading a problem statement and finding relevant information in the text (a number) to answer a question.

In the third pupil class, half of the pupils used the "reformulation and solving" system (n=14) (see Figure 8). This system presented solved problems and guided the learner in solving the new problem. The learners reformulated the problem and then wrote the solution. Then they read the problem report. In contrast to AMBRE-add, this system does not allow the child to select and use a typical problem. We chose this control condition in order to verify the impact on learning of

choosing a typical problem and adapting it. The two “control” interfaces were very similar to the AMBRE-add interface.

Learners were assigned by the teacher to groups (AMBRE-add versus control) according to their mathematical level in such a way that groups were equivalent in terms of mathematical ability.



**Figure 8.** The three systems used in the experiment

Learning was evaluated using several tests. The ability to recognise problems belonging to the same class was tested on a computer at the end of each session using a structural-similarity detection task (Thibodeau, Dufresne & Mestre 1989). This task consisted in reading a first problem, and then in choosing which of two problems could be solved like the first problem. In this task, we manipulated the place of the unknown, the problem type, and its surface features. If pupils had acquired the problem classes using AMBRE-add, they would choose problems based on structural features (problem class) without being influenced by surface features.

The ability to match the equation to the class was tested on a computer after the fifth session and after the last session. In each session, three diagrams representing problem classes were presented to the learner, whose task was to type the equation corresponding to the diagram (by filling in boxes with numbers and an operation).

The ability to solve problems was evaluated via a paper-and-pencil task on which each participant solved six problems. This task was used four times: as a pre-test, after the fourth session, as a post-test, and as a delayed post-test one month after the last session.

We completed our analyses using qualitative methods such as observations, questionnaires, and interviews. We adopted a participatory observation technique. Although observation without intervention would have been more objective, it was not possible to leave the pupils in case they got stuck, because the test was in a school setting. While pupils were using the software, we supervised the session by eliminating technical problems and answering the learners' questions. We wrote down which questions were asked and what help we provided. During the system use by the learners, we observed the strategies they employed, as well as their difficulties. After the last computer session, we administered a questionnaire and held a collective interview with the pupils, which allowed us to gain feedback from them about their use of the software.

Interaction traces were recorded during the use of the system. We counted the number of problems solved, and the time taken per problem, per step, and per substep.

#### 4.2.2 Results

**Quantitative results.** We analyzed the problem-solving performance (see Table 1) by conducting a repeated-measures analysis of variance on problem-solving performance on the four tests, with group (AMBRE-add, simple solving system, reformulation and solving system) as a between-subject variable. Performance on the pre-test was significantly lower than performance on the other tests ( $F(3,192)=18.1$ ,  $p<0.001$ ,  $MSe=1.25$ ) but there was no difference between the groups ( $F(2,64)<1$ , ns,  $MSe=10.23$ ) and no interaction between the group and test variables ( $F(6,192)=1.15$ ,  $p=0.33$ ,  $MSe=1.26$ ). In order to control for the effect of pre-test score heterogeneity, these data were analyzed using a multiple analysis of covariance with pre-test score as the concomitant variable, group as between-subject factor, and the second, third, and fourth tests as dependent variables. The results indicated no significant difference between groups ( $F(2,64)<1$ , ns,  $MSe=5.48$ , observed

power =.19). Because of the low power of the statistical test, the lack of a significant difference between groups does not mean that no effect exists. But if one does exist, the effect size was too small for it to be detected with this number of participants.

Thus, these results show that problem-solving performance improved after the use of the three systems, but they do not indicate that pupils who used AMBRE-add progressed significantly more than the control groups. Analyses of the other two tasks did not yield significant differences between groups either. AMBRE-add appears to have the same impact on learning as the “control” systems.

	Test after the						Delayed	
	Pre-test		4th session		Post-test		post-test	
	M	SD	M	SD	M	SD	M	SD
AMBRE-add	2.38	1.94	4.04	1.95	3.81	2.14	4.16	2.04
Simple solving system	2.76	1.61	3.75	2.13	4.08	2.04	3.59	1.79
Reformulation & solving system	2.42	1.38	3.33	2.06	3.75	1.96	4.42	1.16

**Table 1.** Mean (and standard deviation) of number of problems solved on each test by group

We also analyzed the interaction traces recorded for the three systems in order to determine the number of problems solved with the systems. The interaction traces showed that pupils in the three groups spent the same amount of time working with the systems ( $F(2,60) < 1$ , ns,  $MSe = 5521517$  (the unit is seconds)). The test group used AMBRE-add on average 2 hours and 32 minutes, while the control groups averaged 2 hours and 30 minutes on their systems. However, pupils using AMBRE-add or the “reformulation and solving system” solved significantly fewer problems than pupils using the simple solving system (on average, nine problems vs. fourteen problems over the six sessions) ( $F(2,60) = 9.56$ ,  $p < .01$ ,  $MSe = 4.19$ ).

**Qualitative results.** The observations made during system used showed that the pupils had a lot of trouble using the systems and asked many questions. First, they had difficulty reading the



instructions, the help, and the explanation messages. These messages were not understood and the pupils were unable to identify their mistakes. Moreover, during the first three sessions, pupils using AMBRE-add or the “reformulation and solving system” experienced difficulties both in navigation and in understanding the interface elements. As a result, they asked for help to understand the diagnosis messages and asked questions about how to navigate between the steps and substeps.

The pupils’ actual use of the software differed from its recommended use: in the step involving choosing the typical problem, learners spent little time comparing the problems before choosing. Moreover, we were expecting to see an adaptation process during the adaptation step. Yet pupils rarely used the typical problem or its reformulation to build the equation associated with the problem. Lastly, we noted that some of pupils adopted trial-and-error behavior.

Pupils using the simple solving system had trouble with the fill-in-the-blank operation that corresponded to the problem statement. Even if they had learned such an operation in class, they had difficulty writing this operation in the ITS without depicting the situation on paper. So, in the first sessions, we helped some of them look for salient features in the problem statement.

#### *4.2.3 Discussion*

The results indicated an improvement in problem-solving performance but no significant difference between the three systems on any task. The impact of the three systems on learning seems to be equivalent.

However, the comparison between AMBRE-add and the “simple solving system” is debatable. On one hand, the power of the statistical test was low, so it is possible that the group difference was not detected by the statistical test. In the other hand, the assistance provided to learners during system use differed in nature, depending on the group. The explanations provided to the AMBRE-add pupils were reformulation of the explanation messages, whereas those provided to the “simple solving system” pupils referred to concepts used in the AMBRE cycle. In addition, the AMBRE-add pupils solved fewer problems than the group that used the simplest control software (on average, nine

versus fourteen problems, respectively). We think that AMBRE-add learners' difficulties (concerning reading, understanding messages, and navigating) limited the number of problems solved with this system and thereby reduced the impact of the AMBRE cycle on learning.

In this context, then, we cannot conclude that the AMBRE cycle facilitates the acquisition of problem classes and the association between solving techniques and problem classes more than another problem-solving strategy. Moreover, considering the difficulties observed, AMBRE-add appears to be too complicated for eight-year-old pupils. However, an improvement in problem-solving performance was observed, and the questionnaires showed that AMBRE-add was appreciated both by the pupils and by the teachers, who considered it capable of supplementing the activities proposed in class.

As a result, we modified the messages provided by the system and conducted a new study with nine-year-old pupils in order to find out whether AMBRE-add would be more suitable for these learners and would have a true impact on learning.

### **4.3 Experiment with Nine-Year-Old Pupils**

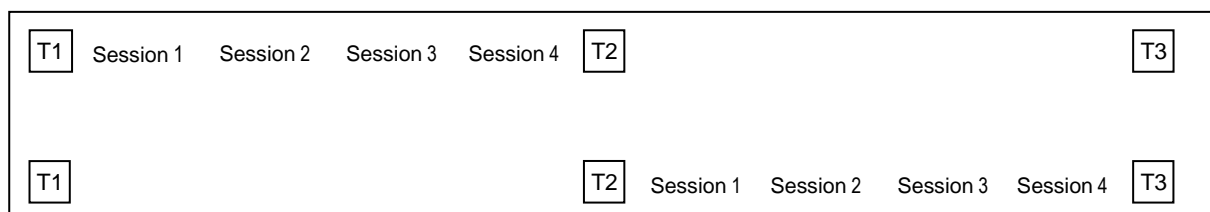
At the age of nine, pupils are better readers and have a greater mastery of arithmetic operations than eight-year-olds, but they still have difficulties with modelling arithmetic problems. AMBRE-add thus seemed to be more suited to these slightly older pupils. So we conducted our next experiment with a class of nine-year-olds. The purpose of this experiment was twofold: determine whether the way the pupils actually used the software more closely matched the recommended way, and evaluate the effect of AMBRE-add on problem-solving performance. In the previous experiment, AMBRE-add was compared to control systems in order to evaluate the impact of the learning strategy (the AMBRE cycle). Because the software turned out to be difficult to use, we chose here to test the impact of the software itself rather than that of the learning strategy. So in this experiment, a simpler control

condition was proposed. Pupils using the AMBRE-add ITS were compared with pupils who performed the same problem-solving task but did not use a computer between tests; they worked on another activity with the teacher.

#### 4.3.1 Experimental Design

A class of 23 nine-year-old pupils participated in the experiment, which had two parts (see Figure 9). In the first part consisting of four sessions, twelve pupils used AMBRE-add in the school's computer room (Group 1), while the other eleven pupils stayed in the classroom with their teacher and worked on French composition (Group 2). During the second part, the two groups were switched: pupils who had not used the system first worked on AMBRE-add for four sessions, while the others stayed in the classroom with the teacher. The problems presented in AMBRE-add were adapted to the pupils' level: the numbers were higher than in the previous experiment, the easiest two problem classes were not presented, and the change-unknown problem classes were included.

Learning was measured with a problem-solving task that consisted in solving six problems belonging to three problem classes. This task was performed before the use of the system (T1), at the end of the first part (T2), and after the last session as a post-test (T3). If AMBRE-add has an impact on learning, the learning gain in the first half of the experiment (T2-T1) should be larger for Group 1 than for Group 2, and the gain in the second part (T3-T2) should be larger for Group 2 than for Group 1.



**Figure 9.** Design of the experiment with nine-year-old pupils

In order to understand find out how the system was actually used, we observed several pupils solving one problem during each session. We described how they responded in each step, what their

reactions to the help messages and explanatory messages were, and what difficulties they experienced. The questions they asked were also noted. The trial-and-error phenomenon was assessed by observing pupils using the methods adapted by Baker, Corbett, Koedinger and Wagner (2004) based on previous quantitative observational studies of student behavior in traditional classrooms (Lloyd & Loper 1986). Each learner's behavior was observed three times per session in a specific order determined before the session began. A pupil's behavior was coded as being in one of the following categories: working in the system, talking on-task, talking off-task, inactive, and trial-and error behavior. At the end of the final session, we asked the pupils what kind of information they had used to find the solution, in order to determine the number of pupils who adapted the typical problem to the solving of a new problem. Finally, the pupils filled out a questionnaire that gave us feedback about their use of the software and their satisfaction.

#### 4.3.2 Results

**Problem-solving task.** The correctness of the problem answers given by each of the 23 pupils was scored. The protocols of two pupils who correctly solved all six problems on the pre-test were discarded. The correctness of the equation written by pupils to solve each problem (e.g. “ $56 + 13 = ?$ ” or “ $? - 13 = 56$ ”) was also scored: this measure is more representative of problem modelling and less sensitive to calculation difficulties. These problem scores are presented in the Table 2.

	Test 1		Test 2		Test 3	
	M	SD	M	SD	M	SD
Group 1	3.00	0.86	4.27	2.08	4.64	1.93
Group 2	4.00	2.05	4.20	1.81	5.10	1.20

**Table 2.** Mean number of correct equations produced on each test, by group

A multiple analysis of covariance was performed on these data, with group as a between-subject variable, Test 2 and Test 3 as dependent variables, and the pre-test as a covariate. This analysis yielded an effect of the test ( $F(1,19)=4.10$ ,  $p<0.05$ ;  $MSe=1.04$ ) but no interaction between group and test ( $F(1,19)<1$ , ns,  $MSe=1.04$ ). The comparison of the learning gain of each group in each part

of the experiment indicated a marginally significant difference between groups in the first part of the experiment ( $t(19)=1.79$ ,  $p=.089$ ), but no difference between groups in the second part ( $t(19)=.86$ ,  $p=.19$ ). The standard deviations were high, especially for the pre-test. Because of the high variance of the data and the small number of pupils who participated in the experiment, we supplemented these analyses by a comparison of the number of pupils in each group according to the learning gain (see Table 3) obtained in each part of the experiment.

	Learning gain 1 (T2-T1)		Learning gain 2 (T3-T2)	
	>0	≤0	>0	≤0
Group 1	8	3	5	6
Group 2	2	8	5	5

**Table 3.** Group size of the two groups distributed according to the learning gain obtained in each part of the experiment

The analysis of the number of pupils distributed according the learning gain obtained between Tests 1 and 2 showed that more pupils progressed in the AMBRE-add condition than in the control condition ( $\chi^2(1)= 5.84$ ,  $p<0.05$ ). However, there was no group difference on the second part of the experiment ( $\chi^2(1)= 0.04$ ,  $p=.835$ ). When the groups on the two parts of the experiment were pooled, the difference between the AMBRE-add condition and the control condition tended to be significant ( $\chi^2(1)=3.44$ ,  $p=.064$ ).

Thus, most of the children progressed after using AMBRE-add. Pupils who made no progress (particularly in the second part of the experiment) were good solvers: on the pre-test, they solved at least five of the six problems.

**Analysis of AMBRE-add use.** The observations showed that the nine-year-old pupils were much more autonomous (after the first session, they asked very few questions), understood quickly how to use the system, and did not experience difficulty in reading the help and error messages. Moreover, a substantial number of pupils seemed to use the software as we intended (particularly

during the adaptation step). Sixteen of the twenty-one pupils said they used the typical problem to find the solution to the problem to be solved; some of them said they used it when they had trouble solving the problem. Only three of the twenty-one pupils occasionally adopted trial-and-error behavior.

The analysis of the interaction traces showed that the nine-year-old pupils using AMBRE-add solved more problems (on average, eleven in four sessions,  $SD=5.3$ ) than the eight-year-old pupils did. However, the variance was high: five pupils solved only one or two problems per session, while others solved all problems proposed by the system in each session. The number of problems solved with AMBRE-add was not correlated with problem-solving performance; four pupils who were good solvers before using the system adopted an inactive behavior during the session and solved only a small number of problems.

To summarize, our quantitative results showed that pupils who had difficulty solving problems on the pre-test improved their performance after using the AMBRE-add ITS. Analysis of system use showed that nine-year-old pupils had less trouble using the system, and that their actual use was close to the intended use. These results are encouraging and allow us to consider AMBRE-add as suited to nine-year-olds, since pupils at this age still study arithmetic word problems that are difficult for them. The impact of the AMBRE cycle on learning still has to be confirmed. We plan to make a comparative evaluation between AMBRE-add and the simple solving system with other nine-old-pupils.

## **CONCLUSION AND PROSPECTS**

In this paper, we presented how our multi-disciplinary team designed the AMBRE-add ITS, based on didactics of mathematics studies concerning the domain of arithmetic problems, and based on cognitive psychology studies on analogical problem solving. For the purposes of teaching problem classes and associated techniques, this ITS was designed to promote analogical reasoning by

proposing an approach to the learner inspired by case-based reasoning. We also explained how, supported by a knowledge-based system, this ITS enables learners to obtain help, diagnosis regarding their answers, and related explanations. Lastly, we described how we conducted several experiments in order to evaluate this environment.

Given that our evaluation of AMBRE-add via a preliminary experiment with nine-year-old pupils proved to be encouraging, we are preparing to repeat the comparative experiment we conducted with eight-year-old pupils, on these older children.

In the longer term, the AMBRE project has two research orientations. The objective of the first is to create a set of activities based on AMBRE-add in view of developing a broader environment for use throughout elementary school. We are working with teachers to determine and design these activities in order to enable better integration of the software in the classroom. These activities are not directly related to CBR, but focus on skills related to arithmetic problem-solving. In some cases, they may prepare pupils for using AMBRE-add, and in others they may enable them to go further into arithmetic word problem-solving.

The second research orientation is aimed at looking into how to transform AMBRE-add so that it can be adapted to the learning context. Our objective is to have this environment be usable in a variety of educational situations by groups of different ages with different profiles. To this end, we are designing tools to enable teachers to customize the environment, create learning sequences, and generate the problems they want their pupils to learn to solve. At the same time, we are looking at how to define a learner's profile, by analyzing that learner's activity when he or she uses the AMBRE-add ITS. A learner's profile should bring together information about the learner's knowledge, skills, and/or conceptions. This will allow the system to adapt its interventions to the learner, and in particular, to generate problems or activities better suited to the learner's difficulties.

It will present the teacher with a report of his or her pupils' use of the software, including a cognitive profile.

**Acknowledgments:** We would like to thank the three reviewers for their comments on earlier versions of the manuscript. This research was supported by the CNRS interdisciplinary program STIC-SHS "Société de l'Information".

## References

- Aadmodt, A., Plaza, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AICOM*, 7, 39-59.
- Aleven, V., Ashley, K.D. (1997). Evaluating a Learning Environment for Case-Based Argumentation Skills. *ICAIL 1997*: 170-179.
- Aleven, V., Koedinger, K.R. (2002). An Effective Meta-cognitive Strategy: Learning by Doing and Explaining with a Computer-Based Cognitive Tutor. *Cognitive Science*, 26 (2), 147-179.
- Atkinson, R.K., Derry, S.J., Renkl, A., Wortham, D.W. (2000). Learning From Examples: Instructional Principles from the Worked Examples Research. *Review of Educational Research*, 70, 181-214.
- Baker, R.S., Corbett, A.T., Koedinger, K.R., Wagner, A.Z. (2004) Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game the System". *Proceedings of ACM CHI 2004: Computer-Human Interaction*, 383-390.
- Bassok, M. (1990). Transfer of Domain-Specific Problem-Solving Procedures. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 16, 522-533.
- Bassok, M., Holyoak, K.J. (1989). Interdomain Transfer Between Isomorphic Topics in Algebra and Physics. *Journal of Experimental Psychology: Learning, Memory & Cognition*. 15, 153-166.
- Bélanger, S., Thibodeau, M.-A. Aimeur, E: (1999). Training of the Learner in Criminal Law by Case-Based Reasoning. In *Twelfth International Conference on Industrial & Engineering Applications of Artificial Intelligence & Expert Systems (IEA/AIE-99)*, Cairo, Egypt, 398-408.
- Bielaczyc, K., Pirolli, P., Brown, A.L. (1995). Training in Self-Explanation and Self-Regulation Strategies: Investigating the Effect of Knowledge Acquisition Activities on Problem Solving. *Journal of Educational Psychology*, 86, 122-133.
- Burke, R., Kass, A. (1996). Retrieving Stories for Cased-Based Teaching. In Leake, D. (ed.), *Case-Based Reasoning: Experiences, Lessons, and future directions*, Menlo Park: AAAI Press/MIT Press, 93-110.
- Capus, L., Tourigny, L. (2000). Le raisonnement à partir de cas : une aide à la formation en analyse de sécurité routière. In *Proceedings of international symposium TICE 2000, Troyes, France*, 227-235.
- Cauzinille-Marmèche, E., Didierjean, A. (1999) Raisonnement par analogie et généralisation des connaissances, in G. Netchine-Grynberg (ed.), *Développement et fonctionnement cognitif : vers une intégration*, Paris, Presses Universitaires de France, 125-152.
- Chen, Z., Mo, L. (2004). Schema Induction in Problem Solving: A Multidimensional Analysis. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 30, 583-600.
- Chi, M.T.H., De Leeuw, N., Chiu, M.H., La Vancher, C. (1994). Eliciting Self-Explanations Improves Understanding. *Cognitive Science*, 18, 439-477.
- Conati, C., VanLehn, K. (2000). 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.
- Cummins, D. (1992). Role of Analogical Reasoning in the Induction of Problem Categories. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 5, 1103-1124.
- Damm, R. (1992). Apprentissage des problèmes additifs et compréhension de texte. PhD thesis, Université Louis Pasteur, Strasbourg.



- Didierjean, A. (2001). Apprendre à partir d'exemples: Abstraction de règles et/ou mémoire d'exemplaires ? *L'Année Psychologique*, 101, 325-348.
- Didierjean, A. (2003). Is Case-Based Reasoning a Source of Knowledge Generalization? *European Journal of Cognitive Psychology*, 15, 435-453.
- Duclosson, N. (2004). Représentation des connaissances dans l'EIAH AMBRE-add, *Technologies de l'Information et de la Connaissance dans l'Enseignement supérieur et l'industrie*, TICE'2004, 164-171.
- Falkenhainer, B., Forbus, K.D., & Gentner, D. (1989). The Structure-Mapping Engine: Algorithm and Examples, *Artificial Intelligence*, 41, 1-63.
- Fayol, M. (1990). *L'enfant et le nombre*. Delachaux & Niestlé.
- Fisher, J.-P. (1979). La perception des problèmes soustractifs aux débuts de l'apprentissage de la soustraction. Doctoral Dissertation in Didactics of Mathematics, Université Nancy 1.
- Fisher, J.-P. (1993). La résolution des problèmes arithmétiques verbaux : propositions pour un enseignement pro-actif. *Annales de Didactique et de Sciences Cognitives* 5, 177-210, IREM de Strasbourg.
- Gentner, D. (1983). Structure Mapping: A Theoretical Framework for Analogy. *Cognitive Science*, 7, 155-170.
- Gick, M.L., Holyoak, K.J.(1983). Schema Induction and Analogical Transfer, *Cognitive Psychology*, 15, 1-38.
- Gick, M.L., Holyoak, K.J. (1980). Analogical Problem Solving. *Cognitive Psychology*, 12, 306-355.
- Gick, M.L., Patterson, K. (1992). Do Contrasting Examples Facilitate Schema Induction and Analogical Transfer? *Canadian Journal of Psychology*, 46, 539-550.
- Goswami, U. (1992). *Analogical Reasoning In Children*. Hillsdale, New Jersey: Erlbaum.
- Greeno, J.G., Riley, M.S. (1987). Processes and Development of Understanding. In *Metacognition, Motivation and understanding*, F.E. Weinert, R.H. Kluwe, Eds, Chap 10, 289-313.
- Guin-Duclosson, N. (1999). SYRCLAD : une architecture de solveurs de problèmes permettant d'explicitier des connaissances de classification, reformulation et résolution. *Revue d'Intelligence Artificielle*, 13 (2), 67-94, Paris, Hermès.
- Holyoak, K.J. (1984). Analogical Thinking and Human Intelligence. In R. J. Sternberg (ed.), *Advances in the Psychology of Human Intelligence*, (2), Hillsdale, New Jersey: Lawrence Erlbaum Associates.
- Holyoak, K.J., Thagard, P. (1989). Analogical Mapping by Constraint Satisfaction, *Cognitive Science*, 13, 295-355.
- Hummel, J. K., Holyoak, K.J. (1997). Distributed Representations of Structure: A Theory of Analogical Access and Mapping, *Psychological Review*, 104, 427-466.
- Jean-Daubias S. (2004). De l'intégration de chercheurs, d'experts, d'enseignants et d'apprenants à la conception d'EIAH *Technologies de l'Information et de la Connaissance dans l'Enseignement supérieur et l'industrie*, TICE'2004, 290-297.
- Joiron, C., Lecllet, D.(2003). Inciting Discussions Between Physicans about Clinical Cases: The Diacom Forum and its Pairing Module. In *Proceedings of the Conference on Open and Online Learning, ICOOL 2003*, Ile Maurice.
- Keane, M.T., Ledgeway, T., & Duff, S. (1994). Constraints on Analogical Mapping: a Comparison of Three Models. *Cognitive Science*, 18, 387-438.
- Kolodner, J. (1993) *Case-Based Reasoning*, San Mateo, Morgan Kaufman Publishers.
- Lloyd, J.W., Loper, A.B. (1986). Measurement and Evaluation of Task-Related Learning Behavior: Attention to Task and Metacognition. *School Psychology Review* 15(3), 336-345.
- Mark, M.A. & Greer., J.E. (1993). Evaluation Methodologies for Intelligent Tutoring Systems. *Journal of Artificial Intelligence in Education*, 4(2-3), 129-153.
- Masterton, S. (1997). The Virtual Participant: Lessons to be Learned from a Case-Based Tutor's Assistant. In *Proceedings of the International conference on Computer Support for Collaborative Learning*, Toronto, 179-186.
- Mille, A. (1998). Associer expertise et expérience pour assister les tâches de l'utilisateur, Habilitation à diriger des recherches, Université Claude Bernard, Lyon1.
- Neuman, Y., Schwarz, B.B. (1998). Is Self-Explanation while Solving Problems Helpful? The Case of Analogical Problem Solving. *British Journal of Educational Psychology*, 68, 15-24.
- Nogry, S. (2003). Comment faire apprendre des connaissances abstraites à partir d'exemples : application au projet AMBRE. EIAH'2003 (session jeunes chercheurs) annexes aux actes de la conférence, 67-70.

- Paas, F., van Merriënboer, J. (1994). Variability of Worked Examples and Transfer of Geometrical Problem-Solving Skills: A Cognitive-Load Approach. *Journal of Educational Psychology*, 86, 122-133.
- Quilici, J.L., Mayer, R.E. (1996). Role of Examples in how Students Learn to Categorize Statistics word problems. *Journal of Educational Psychology*, 88, 144-161.
- Quilici, J.L., Mayer, R.E. (2002). Teaching Students to Recognize Structural Similarities between Statistics Word Problems. *Applied Cognitive Psychology*, 16, 325-342.
- Riley, M.S., Greeno, J.G., Heller, J.I. (1983). Development of Children's Problem-Solving Ability in Arithmetic. Ginsburg H.P. (Ed.) *The development of mathematical thinking*. New-York: Academic Press.
- Rogalski, M. (1994). Les concepts de l'EIAO sont-ils indépendants du domaine? L'exemple d'enseignement de méthodes en analyse. *Recherches en Didactiques des Mathématiques*, 14 (1-2), 43-66.
- Ross, B.H., Kennedy, P.T. (1990). Generalizing from the Use of Earlier Examples in Problem Solving. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 16, 42-55.
- Sander, E. (2000). *L'analogie, du naïf au créatif*, Paris : l'Harmattan.
- Senach, B. (1993). L'évaluation ergonomique des interfaces homme-machine. In *L'ergonomie dans la conception des projets informatiques*. J.-C. Sperandio eds. Octares editions: 69-122.
- Schank, R., Edelson, D. (1990). A Role for AI in Education: Using Technology to Reshape Education. *Journal of Artificial Intelligence in Education*, 1.2, 3-20.
- Schank, R. (1982). *Dynamic Memory: A Theory of Reminding and Learning In Computer and People*. Cambridge University Press.
- Schoenfeld, A. (1985). *Mathematical Problem Solving*. New York: Academic Press.
- Shute, V.J., Regian, J.W. (1993). Principles for evaluating Intelligent Tutoring Systems. *Journal of Artificial Intelligence and Education*, 4 (2/3), 245-271.
- Thibodeau Hardiman, P., Dufresne, R., Mestre, J.P. (1989). The Relation between Problem Categorization and Problem Solving among Experts and Novices. *Memory and Cognition*, 17(5), 627-638.
- Vergnaud, G. (1982). A Classification of Cognitive Tasks and Operations of the Thought involved in Addition and Subtraction Problems. In *Addition and Subtraction: A Cognitive Perspective*, Hillsdale: Erlbaum, 39-58.
- Willis, G.B., Fuson, K.C. (1988). Teaching Children to Use Schematic Drawings to Solve Addition and Subtraction Word Problems. *Journal of Educational Psychology*, 80, 190-201.
- Wong, R.M.F., Lawson, M.J., Keeves, J. (2002). The Effects of Self-Explanation Training on Students' Problem Solving in High-School Mathematics. *Learning and Instruction*, 12(2), 233-262.