A Framework for Abductive Computational Modeling

Osvaldo Luiz de Oliveira
Computer Science Department
UNIFACCAMP
Campo Limpo Paulista (SP), Brazil
osvaldo@faccamp.br

Ricardo Jose Martins
Muzambinho Campus
IFSULDEMINAS
Muzambinho (MG), Brazil
ricardo.martins@muz.ifsuldeminas.edu.br

Marcio Yuji Matsumoto
São Paulo Campus
IFSP & UNIFACCAMP
São Paulo (SP), Brazil
matsumoto@ifsp.edu.br

***Abstract*—This Research to Practice full paper presents a framework for abductive computational modeling. Abduction is a type of reasoning which starts with observed facts then seeks to find hypotheses to explain the facts. This type of reasoning is fundamental for engineers to solve many problems, including design, planning, and fault diagnosis. Computational modeling is the process of programming models of machines, circuits, buildings, and phenomena in general. Abductive Computational Modeling refers to development of programs to make abductive reasoning about devices, techniques, and phenomena that are commonly studied, for example, in the curricula of engineering courses. As an educational activity, modeling belongs to the class known as constructivism, which students learn through activities of developing models and critically analyzing the developed model. Although computational modeling has been extensively investigated—since Papert's seminal studies in the 1960s about the use of the LOGO language, until the present day, demarcated by modeling with Arduino, Raspberry Pi, and languages like C, Scratch, and Python—there is little study on abductive computational modeling. This work proposes and experimentally investigates the use of a framework for abductive computational modeling named AbCM. The results suggest the feasibility of the framework and indicate important challenges to be overcome.**

Keywords—computational modeling, abductive reasoning, constructivism.

I. Introduction

Modeling is the activity of creating models of something (such as concrete things, ideas, and phenomena) [2]. One model could simply be the drawing of a landscape, or a painting, or a clay sculpture, or as sophisticated as computer simulations based on complex mathematical models.

Computational modeling is an activity that uses computational systems to construct a model of something. This activity of construction is extraordinarily rich for educational purposes, because it asks the student to get in touch with a lot of concepts and logical connections about these concepts. In the 1960s, the work of Papert [14] about constructivism using computers with geometrical models written in the LOGO language can be considered the starting point of what is, nowadays, known as MbL—Modeling-based Learning [3]. Nowadays MbL is studied and practiced in various contexts: animations programmed using the Scratch language [11]; development of robots via Arduino [10] or Raspberry Pi [18] electronic components, controlled by software programmed using C or Python languages; simulations of biological processes with CellCollective [6]; and dynamic systems modeled by the STELLA visual programming language [15].

Abduction is a type of reasoning usually described as "backwards thinking", in the sense that it is done in an inverted order compared to deductive reasoning. For example, in algebra, if a, b and c are positive integer numbers, $a = 2$, $b = 3$ and $a + b = c$, then one can deduce that $c = 5$. In the opposite way, abductive reasoning tries to establish possible hypotheses for the values of a and b knowing that $c = 5$. In this example, abductive reasoning leads to four possible hypotheses: (i) $a = 1$, $b = 4$; (ii) $a = 2$, $b = 3$; (iii) $a = 3$, $b = 2$; and (iv) $a = 4$, $b = 1$.

In formal terms, abduction could be defined as the reasoning that produces a set H of hypotheses to explain a set F of facts, taking into account a theory T. In the example on algebra, having a theory

$T = \{$ a, b, c are positive integers, $a + b = c$ $\}$ and

a set of facts (in this case, only one fact)

$F = \{ c = 5 \}$,

abductive reasoning produces a set of hypotheses

$H = \{$

$a = 1$ and $b = 4$, $a = 2$ and $b = 3$, $a = 3$ and $b = 2$, $a = 4$ and $b = 1$

$\}$.

Theories can be understood as models of something and the activity of modeling as an activity of developing theories.

For example, consider the electrical circuit depicted by the diagram shown in Fig. 1. This circuit is composed of two LEDs (light-emitting diode), a battery and two switches. In electronics, a closed switch gives passage to an electrical current; an open switch interrupts the flow of the electrical current. For a LED to emit light, it requires that an electrical current pass through it. So that:

- If LED 1 emits light, the battery is charged and switches 1 and 2 are closed.

- If LED 2 emits light, the battery is charged and switch 1 is closed.

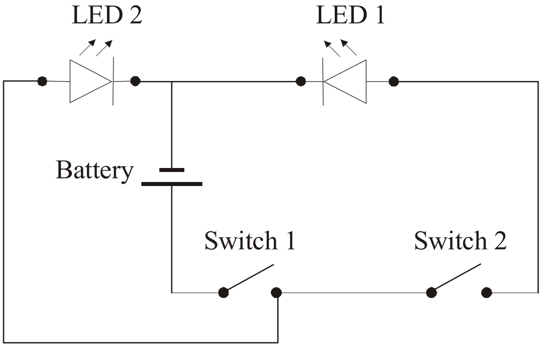


Fig. 1. Diagram of an electrical circuit containing two LEDs (light-emitting diode), one battery and two switches.

It is possible to create a model for the functioning of the circuit in Fig. 1 by using a theory that employs propositional logic [7] sentences. For example, the theory

$$T_c = \{ \; \mathit{chargedBattery} \land \mathit{switch1Closed} \land \mathit{switch2Closed} \rightarrow \mathit{led1On}, \;\; \mathit{chargedBattery} \land \mathit{switch1Closed} \rightarrow \mathit{led2On} \; \},$$

establishes a model for the functioning of the circuit of Fig. 1, which could be written as:

- If the Battery is charged (*chargedBattery*) and switch 1 is closed (*switch1Closed*) and Switch 2 is closed (*switch2Closed*), then LED 1 is on (it emits light).
- If the Battery is charged (*chargedBattery*) and Switch 1 is closed (*switch1Closed*), then LED 1 is on (it emits light).

Models defined by theories such as T_c, which can be written with propositional logic [7] sentences, can be executed by computational systems to calculate, computationally, the set of hypotheses resulting from an abductive reasoning. This process of creating models for computational execution of abductive reasoning is named Abductive Computational Modeling.

This article presents a framework, named AbCM (Abductive Computational Modeling), developed to organize an educational environment for abductive computational modeling. In the educational environment, the student is encouraged to become immersed in a cycle of developing a model, executing the model in the computer and, based on the results, improving the model. This process of improving the model is also the process of improving the student's knowledge. Therefore, the student learns by the activity of constructing and reconstructing models in a constructivist [14] style.

The remainder of this article is organized as follows. Section II describes related works. Section III details three computational systems that can be used for abductive computational modeling via theories denoted by sentences in propositional logic. The framework proposed, AbCM, is described in detail in Section IV. Section V is fully dedicated to one example that explains how the AbCM Framework works. An experimental study about the AbCM Framework usage is presented at Section VI. Finally, Section VII presents conclusions and future work.

Although the experiment reported in this work was carried out with students from a technical school (Electrotechnics, Computing and others) integrated to regular middle education, the AbCM Framework can be used in a diversity of educational contexts such as engineering undergraduate courses, middle schools, and fundamental schools. So this work is general and does not detail one or other educational environment.

II. Related Work

Papert's studies regarding the LOGO language [14] in the late 1960s can be considered as a forerunner of what today is known as MbL—Modeling-based Learning [3].

Recent studies about MbL include investigations regarding the use of systems such as Scratch [11], MATLAB [4], CellCollective [6], Raspberry Pi [18], Arduino [10], and STELLA [15].

Scratch has been used mainly to establish models via animation, MATLAB for mathematical modeling in physics and engineering, and CellCollective for modeling biological processes. STELLA is designed for dynamic phenomenon modeling. The models in STELLA are graphs in which the vertices represent dynamic variables and the edges represent the relationship between the dynamic variables.

Computational systems for abduction have been proposed and investigated outside the educational context [8] [9]. Peirce Online and Abd1 language were the first systems designed specifically for educational abductive computational modeling. Recent studies [16] [12] investigate the feasibility of these systems.

The feasibility of non-specifically designed systems for educational abductive computational modeling has also been investigated. For example, [13] describes a study in which the CHR language (Constraint Handling Rules), running on the mature SWI-Prolog system, was used to create models for abductive reasoning.

Peirce Online, Abd1 language and CHR language will be discussed in the next section.

III. Three Computational Systems for Abductive Modeling

All systems designed for logic programming theoretically have the potential to be used for abductive modeling. However, some of these systems may not be appropriate for use in an educational context. This section presents three systems that can be used for abductive computational modeling.

Peirce Online[1] [16] is a Web system that allows free access (Fig. 2). The main page of the system has a field for description of a theory set and a field for description of a facts set. If a user clicks the "Abductive Reasoning" button, the system calculates a set of hypotheses resulting from abductive reasoning. There is in the Peirce Online system an additional field, named "Conditions Set", which is not relevant to what is discussed in this article. For testing purposes, all the theories and facts described in this article can be copied and pasted into the respective theory and facts fields of the Peirce Online system, without the need for further editing.

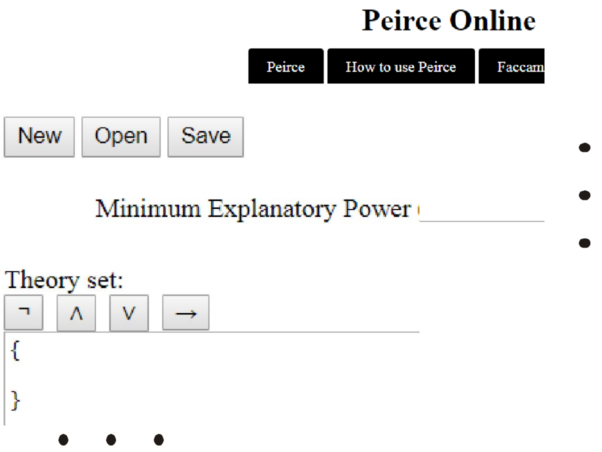


Fig. 2. General appearance of part of the main page of the Peirce Online Web system.

Abd1[2] [12] is free software, available for Windows, and is an integrated environment for editing, compiling and executing abductive models (Fig. 3). The Abd1 integrated environment has a language, also named Abd1, that allows a user to describe models constituted of one or more theories. The environment has recourses to facilitate the editing of big models, either by modularization, allowed by the description of many theories, or by displaying, in a tree view, the names of theories, facts and propositions.

SWI-Prolog[3] is sophisticated and robust software for logic programming [17], available for Linux, MacOSX and Windows, distributed under the Simplified BSD license, also known as the BSD-2 license. SWI-Prolog allows programs to be developed using the Prolog language and contains several extensions. One of these extensions, the CHR (Constraint Handling Rules) language, is particularly useful for describing models for abductive reasoning. The CHR language was designed to describe constraints in logic programming [5]. However, [1] was pioneer in proposing the use of the CHR language in the programming of abductive reasoning. The theories and facts, as written in this article, need to be "translated" into the CHR language to be used. Programs in the CHR language should be written in a generic text editor, usually available on the operating systems (Fig. 4-a) and then loaded and executed on the SWI-Prolog system (Fig. 4-b).

---

[1] Available for use at http://peirceonline.gear.host/.
[2] Available for download at http://abd1.gear.host/.
[3] Available for download at http://www.swi-prolog.org/.

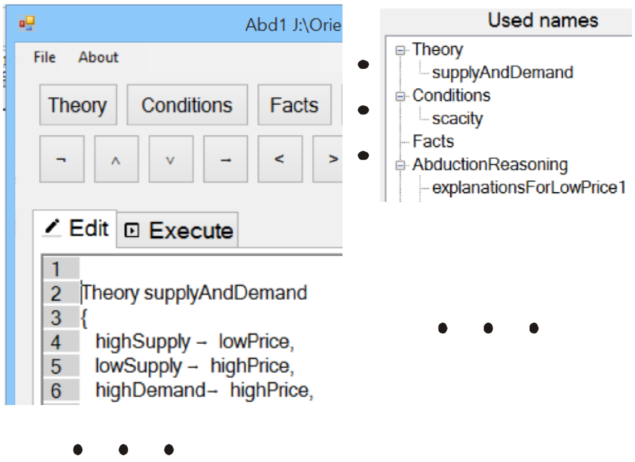
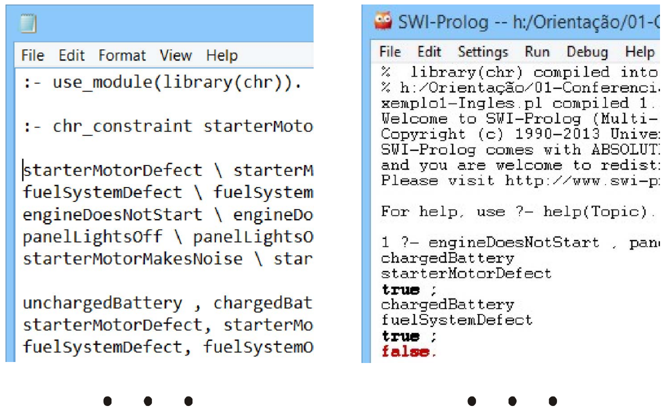


Fig. 3. Overview of some parts of the integrated abductive modeling environment Abd1. One can see, within the "Edit" field, part of a model written in Abd1 language.



(a) (b)

Fig. 4. (a) Part of a model programmed in CHR language, written in a generic text editor (Notepad), available in Windows. (b) Overview of the SWI-Prolog system running an abductive model.

IV. The Proposed Framework: AbCM - Abductive Computational Modeling

Environments for educational computational modeling have in common the cycle in which the student writes, executes, evaluates, and possibly rewrites a model. However, these environments are different in the programming concepts and paradigms they employ. For example, the classical modeling environment for geometry, designed by Papert [14], employs geometry concepts and uses a declarative language (LOGO), which employs deductive logic. In contrast, environments for abductive computational modeling use concepts of theory and facts that employ non-deductive logic to produce sets of possible hypotheses.

In this article the term "framework" is being used to describe how the educational modeling environment will work, what the student should do, and how he should interpret the results to decide whether the model should be rewritten.

The proposed AbCM Framework is distinguished by:

- Using the concepts of theory, fact and hypothesis, as conceptualized in Section I.
- Establishing that a model should be described by a theory.
- Stating that the student must propose facts to elaborate abductive reasonings that, executed by the computer, result in hypotheses to be evaluated.
- Discriminating, as will be seen, four distinct situations that may arise from the evaluation of the hypotheses resulting from model execution. For each of the four situations, the AbCM Framework provides guidance to the student to decide whether the model should be improved or whether the modeling work should be terminated.

The AbCM Framework comprises a modeling activity in which the student enters a cycle in which:

1) The student formulates a logical theory T_1 about the phenomenon being modeled.

2) The student investigates and proposes facts set ($i \geq 1$) about the phenomenon being modeled.

3) The student runs the abductive reasoning described by $\langle T_j, F_i \rangle$ ($j \geq 1$, $i \geq 1$) to obtain a set of hypotheses H_i.

4) The student evaluates the set of hypotheses H_i ($i \geq 1$) in relation to a modeling intention. Typical modeling intentions include the development of models that (i) imitate reality, (ii) are consistent with some existing scientific theory or consolidated knowledge, (iii) are alternatives to existing theories, (iv) are surreal, etc. Four distinct situations may arise from the evaluation of a set of hypotheses H_i.

a) All hypotheses $h \in H_i$ are coherent with the modeling intention.

b) $H_i = F_i$. In this situation, the hypotheses in H_i are not explanatory, i.e., the computed hypotheses do not explain the facts.

c) There is at least one hypothesis $h \in H_i$ not coherent with the modeling intention.

d) The set $H_i = \emptyset$. In this case, there are no hypotheses that explain the set of facts, i. e., the theory T_j is insufficient to explain the facts.

5) The student decides whether to improve the model described by the T_j theory. If the model will be improved, the theory T_j is updated, producing a new theory T_{j+1}, and the student returns to step 2. If the student decides not to improve the theory T_j, then the work is finished. The decision on whether to improve the theory depends on the evaluation done in step 4. The 4-a situation is usually indicative that the developed theory seems to be coherent with the modeling intention. One can be said that the 4-a situation motivates the student to make the decision to end the modeling work. In contrast, situations 4-b, 4-c and 4-d motivate the student to make the decision to continue the modeling work, the first because the hypotheses calculated do not explain the phenomenon, the second because the theory T_j is incoherent with phenomena, and the third because the theory T_j is insufficient to explain the phenomenon being modeled.

This modeling cycle makes the student learn about the phenomenon being modeled, because it exposes the student to the concepts involved in the phenomenon. The presence of the computer as an automaton that executes the model is fundamental to allow the student to have an "other" that sometimes confirms the model (student knowledge) and, at other times, shows the imprecision of the model (student misconceptions).

The four situations described in (4) and the guidelines stated in (5) should be highlighted because they constitute an important support for the student. Other frameworks do not offer these tips.

V. Example of Use of the AbCM Framework

This section exemplifies the creation of a model following the guidelines of the AbCM Framework. The example is a narrative of the process of creating the model. In this example, consider that a student is using a computational environment (such as Peirce Online, Abd1 or SWI-Prolog running the CHR language) to develop the model.

Consider also that the student intends to develop a model that imitates (be consistent) how the circuit shown in Fig. 1 works. The student begins, for example, by establishing the theory T1 (first model), where

$$T_1 = \{ \; chargedBattery \wedge switch1Closed \wedge switch2Closed \rightarrow led1On, \; chargedBattery \wedge switch1Closed \rightarrow led2On \; \}.$$

Then the student proposes the sets of facts

$F_1 = \{ \, led1On \, \}$ e

$F_2 = \{ \, led1Off \, \}$,

and runs the following abductive reasonings:

- $\langle T_1, F_1 \rangle$, whose run has as answer the set of hypotheses (unitary, in this case)

  $H_1 = \{$

  $chargedBattery \wedge switch1Closed \wedge switch2Closed$

  $\}$. That is, having the theory T_1 as a model, the computer calculates for the fact that LED 1 is on (i.e., it emits light), the hypothesis that the battery is charged and switches 1 and 2 are closed.

- $\langle T_1, F_2 \rangle$, whose run has as answer the set of hypotheses (unitary, also in this case)

  $H_2 = \{ \, led1Off \, \}$. That is, taking theory T1 as a model, the computer calculates for the fact that LED 1 is off (i.e., it does not emit light), a hypothesis that is the fact itself (*led1Off*).

The evaluation of these results is as follows. The hypothesis H_1, resulting from the reasoning ⟨ T_1, F_1 ⟩, is clearly consistent with the circuit of Fig. 1 (situation 4-a of the AbCM Framework, Section IV). The consistency of this result does not motivate the student to modify the T_1 theory to improve the model.

However, the hypothesis H_2, resulting from the execution of the reasoning ⟨ T_1, F_2 ⟩, does not explain the circuit of Fig. 1, since the hypothesis computed is the same as the fact (situation 4-b of the AbCM Framework, Section IV). This is like someone answering that "a cup is a cup" to the question "what is a cup?". This leads the student to improve the model, originally described by theory T_1, to produce a new theory T_2, where

T_2 = {

chargedBattery ∧ *switch1Closed* ∧ *switch2Closed* → *led1On* ,

chargedBattery ∧ *switch1Closed* → *led2On* ,

emptyBattery → *led1Off* ∧ *led2Off* ,

switch1Open → *led1Off* ∧ *led2Off* ,

switch2Open → *led1Off*

}.

The improvement of the model from theory T_1 to theory T_2 captures the explanations for the LEDs 1 and 2 of the circuit of Fig. 1 being off. The first two lines of T_2 are equal to the two lines of T_1. The last three lines of T_2 describe the enhancement, via sentences that describe conditions that make LEDs 1 and 2 off.

The student repeats the cycle of the AbCM Framework, returning to step 2, and proposing the facts:

F_3 = { *led1Off* } e

F_4 = { *led1Off, led2On* },

and runs the following abductive reasonings:

- ⟨ T_2, F_3 ⟩, whose run has as answer the set of three hypotheses

H_3 = { *emptyBattery* , *switch1Open* , *switch2Open* }.

- ⟨ T_2, F_4 ⟩, whose run has as answer the set also of three hypotheses

H_4 = {

chargedBattery ∧ *emptyBattery* ∧ *switch1Closed* ,

chargedBattery ∧ *switch1Open* ∧ *switch1Closed* ,

chargedBattery ∧ *switch1Closed* ∧ *switch2Open*

}.

The set of hypotheses H_3 indicates that theory T_2, unlike theory T_1, has explanations for the fact "LED 1 is off": the battery is empty or switch 1 is open or switch 2 is open. That is, the theory T_2 is a model more improved than the theory T_1, with respect to imitating the behavior of the circuit of Fig. 1.

However, the hypotheses set H_4 contain two hypotheses not coherent with the circuit of Fig. 1 (situation 4-c of the AbCM Framework, Section IV). The first hypothesis describes the simultaneous occurrence of the conditions "charged battery", "empty battery" and "switch 1 closed". This explanation is clearly not consistent with the circuit of Fig. 1, since it assumes that at the same time the battery is charged and empty. The second hypothesis presents a problem like the first hypothesis: it describes the simultaneous condition of the switch 1 being open and closed. The third hypothesis of the hypotheses set H_4 is coherent with the circuit, but the incoherence of the first two hypotheses leads the student again to improve the model, described by theory T_2, to develop a new theory T_3, where

T_3 = {

chargedBattery ∧ *switch1Closed* ∧ *switch2Closed* → *led1On* ,

chargedBattery ∧ *switch1Closed* → *led2On* ,

emptyBattery → *led1Off* ∧ *led2Off* ,

switch1Open → *led1Off* ∧ *led2Off* ,

switch2Open → *led1Off* ,

chargedBattery → ¬ *emptyBattery* ,

switch1Open → ¬ *switch1Closed* ,

switch2Open → ¬ *switch2Closed*

}.

The T_3 theory was developed by adding three lines to the T_2 theory (the last three lines of the T_3). Moreover, the theories T_2 and T_3 are the same. The lines added to the T_3 theory inform antonyms not modeled by the T_2 theory: charged battery is the opposite of empty battery, switch 1 is open is the opposite of switch 1 closed, and switch 2 is open the opposite of switch 2 closed.

The student repeats the cycle of the AbCM Framework, returning to step 2, and proposing the facts:

F_5 = { *led1Off* , *led2On* } e

F_6 = { *led1On* , *led2Off* },

and runs the following abductive reasonings:

- ⟨ T_3, F_5 ⟩, whose run has as answer the hypotheses set H5 with one hypothesis

H_5 = { *chargedBattery* ∧ *switch1Closed* ∧ *switch2Open* }.

- ⟨ T_3, F_6 ⟩, whose run has as answer the hypotheses set H6 (empty set)

H_6 = { } = ∅.

The hypotheses set H_5 indicates that the model, improved by the theory T_3, solves the problem of incoherence between the model, described by the theory T_2, and the circuit of Fig. 1. The hypothesis present in the set H_5 explains that the simultaneous occurrence of the conditions "charged battery", "switch 1 closed" and "switch 2 open" explains the occurrence of the facts "LED 1 off" and "LED 2 on".

However, the system computed the hypothesis set H_6, empty, as a response to the reasoning $\langle T_3, F_6 \rangle$, i.e., the model described by theory T_3 has no explanation for the simultaneous occurrence of the facts "LED 1 on" and "LED 2 off" (situation 4-d of the AbCM Framework, Section IV). This situation suggests to the student the need to improve the model described by the T_3 theory. The model can be improved in many ways; for example, the student can incorporate the concept that the circuit cables need to be connected and that they can cause failures if they are disconnected. A new theory can be proposed as an improvement of the T_3 theory, and the cycle of the AbCM Framework can be repeated theoretically indefinitely. In practice, the student interrupts the theoretically infinite process of model improvement at some point, considering the commitment over time and the satisfaction with the results that can be computed with the model.

VI. Experimental Study about Use of the AbCM Framework

An experimental laboratory study on the use of the AbCM Framework was carried out. A total of 88 students from a technical school (Electrotechnics, Computing) integrated to regular middle education participated in the experiment, conducted at Federal Institute of Education, Science and Technology of the South of the Minas Gerais, Muzambinho, Brazil. Participants were young (ranging in age from 16 to 19 years old, mean and median ages of 17 years); approximately half were female and half were male.

The study aimed to investigate difficulties in performing the modeling activity with the guidelines of the AbCM Framework. The research was performed in experimental sessions with the participation of a maximum of five students in five laboratory workstations, each station being equipped with a desk, a chair, a computer with an Intel i3 processor, 3.3 GHz, 4 GB of RAM, running Windows 10 Pro, a text editor and the SWI-Prolog system running the CHR language.

Each experimental session was performed in about 2 hours. Initially, participants were introduced to the concepts of abductive logic modeling and logical sentence writing using the CHR language. The students who participated in the experiment were asked to develop a model for the ignition system and headlights of a car. During the session, the researchers made quantitative and qualitative observations on the student's ability:

- To abstract concepts about the systems to be modeled.
- To describe the model in the computer, i.e., ability to write the model using a language which, in the case of the experiment, is the CHR language.
- To propose facts and run the model.
- To evaluate the results of the run (i.e., read and interpret the results) and decide whether to improve the model.

The experimental session consisted of five phases detailed below. For each phase, the scores attributed to the activities were obtained comparing the answers given by the students with those assumed as the pattern of responses. The theme of the study done by the students was the operation of the car's light and ignition system.

Phase 1. Some problems requiring students' ability to abstract concepts were proposed. The researcher explained the functioning of the headlight system and also about the starter system of the car, and asked the students to respond some questions. The exercises requested the description of the concepts and the proposition of logical relationships among them in order to establish models. The answers were analyzed according to the completeness of abstracted concepts and sentences considered pertinent. It was assigned value 0 for incomplete answers, and value 1 for answers considered complete. The values were summed up and the final grade normalized according to the number of questions.

Phase 2. The researcher first explained about CHR language and the SWI-Prolog environment, and presented some example of their application. After, the students developed in CHR language, a model explaining why the car headlights are on/off, using logical relations between the concepts suggested in the problem. So they wrote a program in CHR language, compiled, executed and modified the template as many times as they wanted to. The programs were analyzed according to syntactic and semantic correctness. In case of a syntax error, the value 0 was assigned. If there was no syntax error, the results of execution were compared with the result of the execution of standard model. For each result that coincided with that of the standard model, we added the value 1 and, at the end, the grade was normalized.

Phase 3. Some problems requiring students' ability to propose facts and execute a model were presented. They received a file containing a computing model explaining the fact that the headlights are on or off. The ability to propose facts and execute a model was analyzed according to the correctness of the responses. We compared the answers with those assumed as correct and the values assigned were summed up and then normalized.

Phase 4. Students solved problems that required their ability to evaluate the results of the execution of a model. First, they executed a computer model that explains why the car headlights are on/off. After, they solved multiple choice questions about the running results. The correctness of the response was evaluated. The values assigned were summed up and then normalized.

We assigned grades ranging from 0 to 10 for student performance in the experiment. Table I presents the mean and standard deviation of the scores obtained by the students in the different abilities analyzed.

These results suggest a regular performance of the students in the ability to abstract concepts, and a good performance of the students in the other abilities required for the abductive computational modeling process using the AbCM Framework guidelines. Assuming that each of these abilities has equal weight in the modeling process, the overall performance of the participants is equal to 7.7, and this can be considered a good overall performance.

TABLE I. RESULTS (MEAN AND STANDARD DEVIATION) OF THE SCORES OBTAINED BY PARTICIPANTS IN DIFFERENT ABILITIES

Ability	Mean (0 – 10) ± SD
Abstract the concepts (Phase 1)	5.8 ± 2.6
Write the model computationally (Phase 2)	8.3 ± 2.7
Propose facts (Phase 3)	9.4 ± 1.6
Evaluate the results and decide about improvement (Phase 4)	7.5 ± 1.9

For the study population, the ability to abstract concepts stands out negatively among the other abilities needed for modeling. Unfortunately, this ability is critical in the modeling process, because an eventual failure in the abstraction of concepts has a strong negative influence on the other activities of the modeling process. Failure in the activity of concept abstraction may prevent a student from developing a model, and this may cause student demotivation with modeling. In this study, the ability to abstract concepts was studied independently of the other abilities, so that failures in this activity had no negative effect on other activities of the modeling process. However, the results suggest that, for the population studied, the ability to abstract concepts is the "critical point" of the modeling process.

Qualitatively, considering interactions with participants during the experimental sessions, researchers interpret that participants engaged in the proposed activities with enthusiasm and demonstrated gratitude for what the abductive modeling activity provided them in terms of learning. The freedom to create models led the student to demonstrate a relation of affection with the model he created.

VII. CONCLUSIONS AND FUTURE WORK

The AbCM Educational Abductive Computational Modeling Framework acts as a guide for actors (students and teachers) in relation to activities that occur within the educational environment. Guidance on the decisions that the student must make in the different situations that may arise from evaluating the results of running a model (items 4 and 5 of the AbCM Framework, Section IV) are valuable and deserve to be highlighted.

Regardless of whether the SWI-Prolog / CHR language system was employed in the experimental study, the AbCM Framework is general and can be instantiated in different ways. For example, one can use the AbCM Framework with workstations running systems like Peirce Online [16] or Abd1 [12].

For the population studied, the results suggest the feasibility of the AbCM Framework. The results also showed that, among the four modeling abilities investigated, the students' performance is not so good in the ability to abstract concepts. This alerts us to the need to technically and/or pedagogically treat the modeling environment to offer the student help to overcome the difficulties imposed to the practice of this ability.

It should be emphasized that there is no guarantee that the results of this study can be reproduced in other populations with different levels of knowledge. An instantiation of the AbCM Framework with a computational system for modeling different from the one used can also lead to results different from those obtained. However, the results obtained create a positive expectation, due to the good performance of the studied population in the modeling work carried out.

Future works may contribute to reduce the difficulties, revealed by the experimental study, of students in abstracting concepts necessary for the modeling process. Experimental studies of using the AbCM Framework, instantiated in educational environments with different computational systems (such as Peirce Online, Abd1), can bring valuable information to the use of abductive computational modeling in school practice.

ACKNOWLEDGMENTS

We would like to thank all subjects who volunteered for the experiment for their invaluable contribution to this work.

REFERENCES

[1] S. Abdennadher and H. Christiansen, "An experimental CLP platform for integrity constraints and abduction," in Proceedings of the Flexible Query Answering Systems: Advances in Soft Computing Series (FQAS), 2000, pp. 141–152.

[2] T. Campbell and P. Oh, "Engaging students in modeling as an epistemic practice of science," Journal of Science Education and Technology, vol. 24, n. 2, 2015, pp. 125–131.

[3] J. Clement and M. A. Rea-Ramirez, Model based learning and instruction in science: models and modeling in science education, J. Clement and M. A. Rea-Ramirez, Eds. Dordrecht: Springer, 2007.

[4] A. B. Downey, Physical modeling in MATLAB. Needham: Green Tea Press, 2011.

[5] T. Frühwirth, Constraint handling rules. Cambridge: Cambridge University Press, 2009.

[6] T. Helikar et al., "Integrating interactive computational modeling in biology curricula," PLOS computational biology, vol. 11, n. 3, e1004131, 2015.

[7] P. Howard, *Introduction to Logic: propositional logic*. New Jersey: Prentice-Hall, 1999.

[8] A. C. Kakas, R. A. Kowalski, and F. Toni, Abductive logic programming. Journal of Logic and Computation, vol. 2, n. 6, 1992, pp. 719–770.

[9] A. C. Kakas, A. Michael, and C. Mourlas, ACLP: Abductive constraint logic programming. Journal of Logic Programming, vol. 44, 2000, pp. 129–177.

[10] S. W. Kim and Y. Lee, "Development and application of arduino-based education program for high school students'," Journal of Theoretical and Applied Information Technology, vol 95, n. 18, 2017, pp. 4367-4375.

[11] V. Lopez, and M. I. Hernandez, "Scratch as a computational modelling tool for teaching physics," Physics Education, vol. 50, n. 3, 2015, pp. 310–316.

[12] O. L. Oliveira, C. E. A. Oliveira, R. J. Martins, and M. Matsumoto, "A logic programming language designed for the modeling of abductive reasoning in an educational context," in Procedings of the Mexican International Conference on Artificial Intelligence (MICAI), 2016, pp. 1–13.

[13] O. L. Oliveira and R. J. Martins, "Computational Modeling of Abductive Reasoning: an experimental study employing the CHR language in an

educational context," in Proceedings of the 47th IEEE Annual Frontiers in Education (FIE), 2017, DOI 10.1109/FIE.2017.8190684.

[14] S. Papert, Mindstorms: children, computers, and powerful ideas. 2nd ed. New York: Basic Books, 1993.

[15] B. Richmond, S. Peterson, and P. Vescuso, An academic user's guide to STELLA. Lyme: High Performance Systems, 1987.

[16] F. Rodrigues, C. E. A. Oliveira, and O. L. Oliveira, "Peirce: an algorithm for abductive reasoning operating with a quaternary reasoning framework," Research in Computer Science, vol. 82, 2014, pp. 53–66.

[17] SWI-Prolog-Org, SWI-Prolog organization, 2018. Available in http://www.swi-prolog.org. Acessed: 22 April 2018.

[18] J. Toth-Chernin, "Raspberry Pi," Knowledge Quest, vol. 43, n. 3, 2015, pp.74-75.