A Strategy to Early Identify Potential Failing Students in Introductory Programming Courses

Márcio Ribeiro[a], Rodrigo Paes[a,*], Rohit Gheyi[b]

[a]*Federal University of Alagoas, Maceió, Brazil*
[b]*Federal University of Campina Grande, Campina Grande, Brazil*

Abstract

[...]

Keywords: Programming courses, clustering

2010 MSC: 00-01, 99-00

1. Introduction

- Context: learning during programming courses
- Problem: *early* identification of potential failing students
- Solution: exercises system, metrics measurement, and clustering algorithms.
- Evaluation: metrics collection during 3,5 years of real programming courses at UFAL, Brazil. Main results: in the first 30 days of the courses we analyzed, our approach could identify students that did not complete the course successfully. We believe that providing early support to these students right after this identification in 30 days, they will have better chance of completing the course.
- Contributions: approach/strategy?

*Corresponding author
Email addresses: marcio@ic.ufal.br (Márcio Ribeiro), rodrigo@ic.ufal.br (Rodrigo Paes), rohit@dsc.ufcg.edu.br (Rohit Gheyi)

2. Problem

The task of identifying the skills and deficiencies of each student in particular is definitely not easy. This happens specially when professors have lots of students with different backgrounds. Since it is difficult to identify these students, professors are not aware of what actually hinder them during the learning process. Consequently, students get frustrated and disappointed—specially when they see their friends excited about the course—causing shame and timidity, which may lead them to, among other things, not ask questions or not participate in class. When considering introductory programming courses, this frustration commonly lead students to fail the course [].

In this context, there is no easy way to *early* identify potential dropout students during introductory programming courses. When not early identifying students which tend to drop out the programming course, professors and assistants may act late (or even may not act, since there is no time anymore) and the students drop out the course anyway. In addition, [consider more consequences! Why this is painful?!]

3. Early detection of potential dropout students

This way, the problem we address in this paper consists of a strategy to *early* identify potential failing students. Notice that identifying early is very important in the sense that professors still have enough time to act and avoid students to fail the course. This way, we focus exclusively on early identifying these students. Providing or reporting a technique useful to reduce the failure rate is outside the scope of this paper. Nevertheless, notice that when identifying them we may study the data and afterwards provide a technique to do so. Actually, based on the results of this paper, we intend to propose a technique to reduce the failure rate as future work.

Next, we detail our strategy to identify these students.

3.1. Online exercises system

As a side effect, we believe that this kind of system will be useful to improve the overall performance of students:

To evaluate the students performance during the course, we use an online exercises system named Huxley []. [algum texto ja pronto?]

Problem-solving approach

The more practice the more learning

Programming is learned by programming, not from books []. The Huxley allows student to avoid the "learned helplessness" problems. This problem happens when a student who has missed a basic concept and who then cannot follow the next lecture. She believes that there is no going back; the course is behaving very much as a high-speed train with no brakes. Such a student will quickly come to the view that "they just can't do programming", and will attribute this to the perceived difficulty of the subject.

3.2. Metrics

We use two metrics in our strategy. We explain them in what follows:

- **Number of submissions:** this metric represents the number of submissions a student do during the course. As explained, Huxley provides more than 300 programming exercises. To solve a particular problem, a student needs to submit at least one solution.
- **Number of correct submissions:** if a student solves one problem, we increase this metric by one.

Depending on the level of difficulty of a problem, students may submit several times to solve it. However, notice that this is not necessarily a bad thing regarding the learning process. For example, submitting many times means that students are somehow practicing and studying continuously. Thus, although she is not hitting a right solution, she is trying hard and will eventually hit one. Nevertheless, we need to carefully analyze [. . .]

[Por que essas metricas sao boas? Por que escolhemos elas? O texto acima ajuda a explicar isso? Essas metricas nao sao obvias?]

3.3. Clustering algorithm

To identify potential failing students, we use the well-known clustering algorithm k-means []. As input, we set the algorithm to compute three groups. To compute the groups, the algorithm takes into account the metrics we present in Section 3.2.

We choose three groups because ???

[Por que escolhemos clustering? Por que o k-means?]

[Por que 3 grupos? O do meio seria impossvel prever algo. Mas isso nao e obvio?]

3.4. Summary

Figure 1 combines all steps of our strategy. As the classes are happening, students are encouraged to solve exercises using the problems of Huxley. Although solving exercises is not mandatory, there some particular activities where the professor forces students to use Huxley. Next, we collect the metrics we detail in Section 3.2. To identify potential failing students *early*, we collect the metrics for the first 30 days of the introductory programming courses. Then, we execute the k-means clustering algorithm.

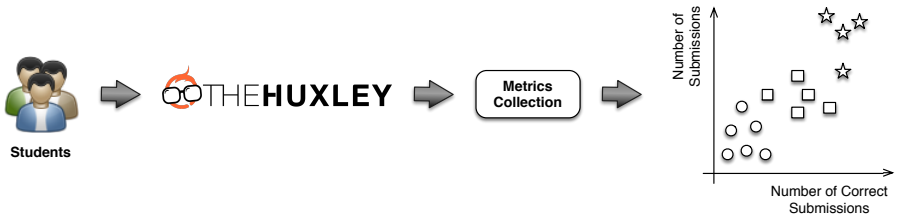

Figure 1: Summary of our strategy to identify potential failing students.

4. Empirical Evaluation

To evaluate our strategy, in this section we present the empirical study we conduct. Before explaining our study, we first introduce the terminology we use throughout this paper. In particular, we define in what follows four categories: abort, skip, fail, and pass.

- **Abort:** students that aborted the course before the final exam;
- **Skip:** students that did not show up for the final exam, but were allowed to;
- **Fail:** students who failed the course;
- **Pass:** students who successfully passed the course.

Now, we present the objetives and hypotheses of our study, then the participants and material we use, and finally we detail the procedure we use during the evaluation.

4.1. Objective and Hypotheses

The objective of this study is to evaluate to what extent our strategy is capable of identifying potential failing students. This way, based on this objective, our hypotheses are the following:

- **RH 1:** In the first 30 days, students with lower number of submissions and correct submissions tend to fail the course.
- **RH 2:** In the first 30 days, students with higher number of submissions and correct submissions tend to pass the course;

Although this paper focuses on identifying potential *failing* students, we also study and report the opposite case according to our second hypothesis.

4.2. Participants and Material

The participants of our study consist of students of introductory programming courses at the Federal University of Alagoas, Brazil. We ministered these courses during 3.5 years and collected the results of each student by using Huxley. The professor informed all students that the use of Huxley was mandatory during the courses. Table 1 distribute the number of participants per semester.

Course	**Number of enrolled students**
2010.02	34
2011.01	38
2011.02	35
2012.01	34
2012.02	29
2013.01	28
2013.02	31

Table 1: Participants per course.

The material of this study consists of almost 300 programming exercises in Huxley. They were available to all students of all courses we use in this paper.

4.3. Procedure

Figure 2 illustrates how we proceed with our evaluation. The result of executing our strategy consists of three groups. Now, according to the groups, we have the potential failing students, the ones that have a small number of submissions and correct submissions (represented by circles). Notice we also have students that are potential candidates to successfully pass (represented by stars): after 30 days, they seem to be studying hard, due to the high number of submissions to Huxley.

After applying the clustering algorithm, we now need to check if the entire strategy correctly predicted the failing students. To do so, we use the academic system of the Federal University of Alagoas to look for grades and check whether

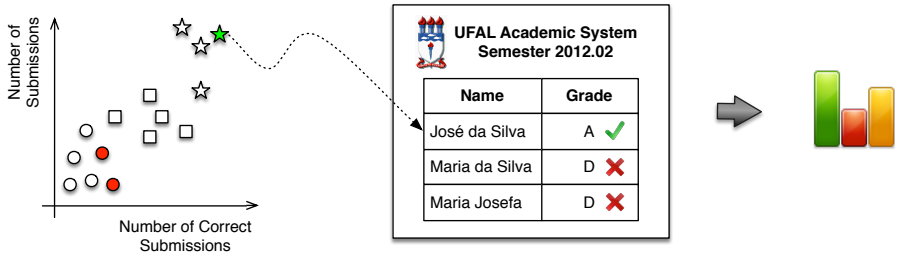

Figure 2: Checking the strategy results against the actual grades.

the students passed or not. For example, for the detached circles, the strategy successfully identified that, after 30 days, both students would not pass and they indeed did not. Notice that the strategy may be analogously applied the other way around: when considering the detached star, the strategy identified that such a student would pass and she indeed passed.

[No estou conseguindo explicar bem a regra de associacao. Tem que melhorar muito mais isso!]

To better structure and analyze our results, we consider association rules [], which take the form of $X \Rightarrow Y$, where X and Y are sets of items. In this paper, these items consist of students. Let S be a set of students. The rules we consider are:

- **Rule 1:** Strategy pointed S as "will Fail" $\Rightarrow$ S indeed Failed
- **Rule 2:** Strategy pointed S as "will Pass" $\Rightarrow$ S indeed Passed

To compute the strength or reliability of these rules, we use the confidence [], which represents the probability of finding the right-hand side of the rule under the condition that these transactions contain the left-hand side as well. This way, the confidence is defined as follows:

$$Confidence(X \Rightarrow Y) = \frac{Support(X \cup Y)}{Support(X)},$$

where $Support$ is a function to [...]

Nevertheless, notice that our strategy is susceptible to yield false positives (the strategy pointed the student would pass, but she did not) and false negatives (the strategy pointed the student would not pass, but she did). We also consider these cases as well.

In summary, after confronting the strategy results with the academic system and summarizing all confidences, false negatives, and false positives, we apply statistical tests to check for significance. Here, we rely on the binomial statistical test based on the Bernoulli distribution [].

5. Results and Discussion

In this section, we describe the results and test our hypotheses before discussing their implications (All data, materials, and R scripts are available at `http://www.ic.ufal.br/`). We now proceed separately, reporting the results where our strategy pointed out students who would fail the course and who would pass the course.

5.1. Fail

Course	**Confidence(Rule 1)**
2010.02	100%
2011.01	84.62%
2011.02	83.33%
2012.01	75%
2012.02	66.67%
2013.01	58.33%
2013.02	81.82%

Table 2: Confidences for the Rule 1 of Section 4.3.

5.2. Pass

5.3. Discussion

- Olhar se, para os alunos onde o algoritmo errou, eles foram pra prova final.

Course	Confidence(Rule 2)
2010.02	33.33%
2011.01	100%
2011.02	88.89%
2012.01	50%
2012.02	66.67
2013.01	100%
2013.02	100%

Table 3: Confidences for the Rule 2 of Section 4.3.

- Olhar se, para os alunos onde o algoritmo acertou, quantos deles falharam.

5.4. Threats to validity

6. Related Work

Reducing the failure rate in programming courses has been achieved in previous work []. The authors performed a study during four semesters. They identified three main factors to reduce the failure rate: (i) using Python as the first introductory programming language, which, according to the authors, may lead the students to better focus on algorithms and problem solving, instead of spending time with advanced concepts of other existing languages at an early stage of the learning process; (ii) using visualization environments to support students and improve the way they can understand abstract concepts related to programming; and (iii) assigning individual problems to each student, hindering students sharing or borrowing solutions to their friends. Like our work, the authors are concerned with the high rate of failure students. However, while they focus on a solution to decrease such numbers, we provide a strategy—evaluated in seven semesters—to early (30 days) identify potential failing students. Then, professors and assistants may act to help these students with additional classes and particular conversations. After studying and understanding why exactly they tend to fail, we then are ready to provide a solution. [achei esse final meio

estranho. Nao vendi bem!]

7. Concluding Remarks

[. . .]