Reflecting on teaching and scholarship after a student says “Thank You”

Hello!

I am Andres Bejarano

Currently a Visiting Assistant Professor in CS@Purdue teaching CS251 – Data Structures and Algorithms

Content

- Career and Teaching
- Teaching Experiences
- Teaching Challenges
- Scholarship Interests

Wholesome Student #1

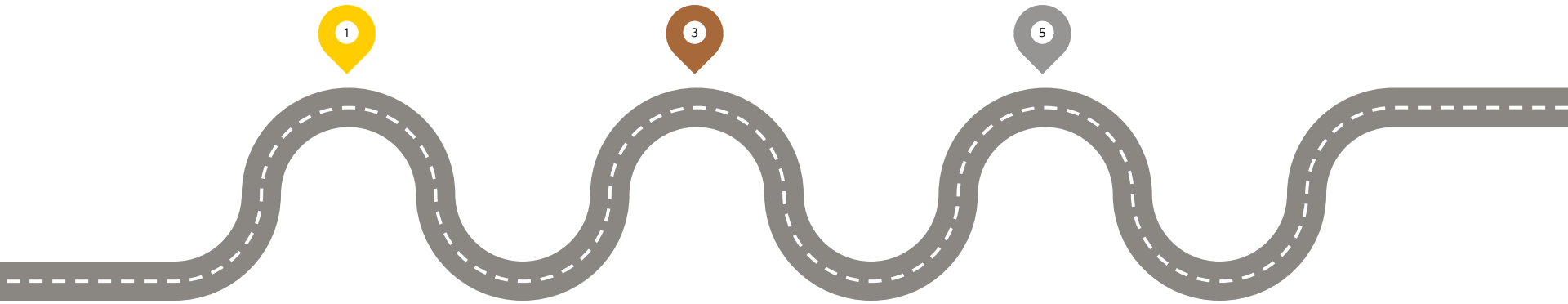
I just wanted to send you a quick thank you for an awesome semester in 251. [...] Additionally, on a personal level, the class this semester helped me feel a lot more like a "Computer Scientist." As you said many times, 251 was not about learning every single data structure or algorithm we will ever encounter, but being able to understand the conceptual and fundamental aspects of these topics; and thus being able to better understand newer and more complex applications beyond the class. [...]

1

Career and Teaching

How did I get here?

Career Path



1
Research Assistant / Software Engineer
Universidad del Norte, Colombia

2
Adjunct Professor
Dept. Systems Engineering and Comp.
Universidad del Norte, Colombia

3
Graduate Teaching Assistant
Department of Computer Science
Purdue University

4
Software Engineer Intern
DirectX Driver Dev. Team
NVIDIA Co.

5
Research Associate
Rosen Center for Advanced Computing
Purdue University

6
Visiting Assistant Professor
Department of Computer Science
Purdue University

Teaching Experience (Instructor of Record)

Algorithms and Programming I

Fall 2010, Spring 2011, Fall 2013

95 students

Mobile App Programming

Fall 2012, Spring 2013, Fall 2013, Spring 2014

84 students

Fundamentals of Computer Graphics

Fall 2012, Spring 2013

19 students

Multimedia (Communications)

Spring 2012, Fall 2012, Spring 2013, Fall 2013, Spring 2014

133 students

Data Structures and Algorithms

Fall 2020, Spring 2021, Summer 2021, Fall 2021, Spring 2022

1383 students

109 TAs

Foundations of Computer Science

Summer 2021

96 students

8 TAs

Wholesome Student #2

Thank you for teaching Algorithm class this semester. I learned a lot of things. I not only learned class materials, but also learned how to think more in Computer Scientists way. Other than that, I learned how to express what I think in a clear way. Moreover, I learned […]. You didn't teach them directly in the class, but I could learn those things from the way you teach. […]. Thank you […].

2

Teaching Experiences

Adapting teaching through time

At Universidad del Norte, Col.

- **Algorithms and Programming I**
  - First teaching experience.
  - Teaching such topics to students who didn't like programming.
  - Curriculum managed directly by the department.
  - Small courses.

At Universidad del Norte, Col.

- **Fundamentals of Computer Graphics**
  - Started as a student group.
  - Motivation grew in the department.
  - Designed curriculum and course logistics.
  - Too late for most of interested students.
  - Smallest courses I’ve had.

At Universidad del Norte, Col.

- **Mobile App Programming**
  - Urgent update!
  - From J2ME to Android and PhoneGap.
  - Reworked curriculum and course logistics.
  - One of most popular courses (due of relevance).
  - Joint Final Projects with Web Development course.

At Purdue University

- **Data Structures and Algorithms**
  - Largest courses I've ever taught.
  - Great rapport with students.
  - Some students did their research!
  - Solid TAs and Instructional staff.

Non-Academic Experiences

- **Communications Office, Universidad del Norte**
  - Developing and maintaining services.
- **NVIDIA**
  - Internal tools development.
- **Rosen-Center for Advanced Computing**
  - Sophisticated users with real data.

Wholesome Student #3

Earlier today, I had a technical interview with [...]. I was able to talk about data structures [...] and calculate runtime complexity with ease due to the extensive projects [...]. In simpler terms, I just wanted to express how thankful I am that this class goes over such important interview topics and how although this class can be very challenging and time-consuming at times, it truly does pay off in the end. I can already tell how much I've learned during the first few weeks of this course. Thank you!

3

Teaching Challenges

Let’s face it, sometimes it is not easy

Some Teaching Challenges

- **Predisposition**

"Is the course hard? Everybody says so."

- **Student Experiences**

"*Why you grade so harsh?* –Vs- *Why you grade so lenient?*"

- **Current Trends in Tech**

"Coding Interview Process."

Seriously. Code Everything!

If you understand the concepts, then you will know how to implement them.

"My code doesn't work, and I don't know why!" Then **debug it!**

PEBCAK: Problem Exists Between Chair and Keyboard.

One of my first slides every semester

Practicing For Coding Interviews

- Listen! we don’t solve everything with hash tables, recursion, and dynamic programming.
- Not even scratching the surface.
- Good for getting a job, but not for keeping it!

Online Teaching Challenges

- *“It doesn’t feel we have a real instructor”*
- Solved with more office hours.

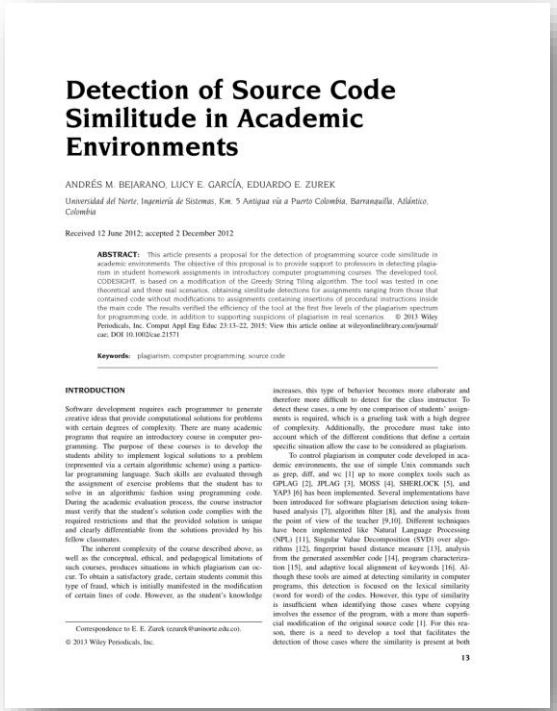
- *“It doesn’t feel we are in a real class”*
- Solved with a document camera.

4

Scholarship Interests

Some venues I would like to pursue as PoP

Computer Science Education



Detection of Source Code Similitude in Academic Environments

ANDRÉS M. BEJARANO, LUCY E. GARCÍA, EDUARDO E. ZUREK

Universidad del Norte, Ingeniería de Sistemas, Km. 5 Antigua vía a Puerto Colombia, Barranquilla, Atlántico, Colombia

Received 12 June 2012; accepted 2 December 2012

ABSTRACT: This article presents a proposal for the detection of programming source code similitude in academic environments. The objective of this proposal is to provide support to professors in detecting plagiarism in student homework assignments in introductory computer programming courses. The developed tool, CODESIGHT, is based on a modification of the Greedy String Tiling algorithm. The tool was tested in one theoretical and three real scenarios, obtaining similitude detections for assignments ranging from those that contained code without modifications to assignments containing insertions of procedural instructions inside the main code. The results verified the efficiency of the tool at the first five levels of the plagiarism spectrum for programming code, in addition to supporting suspicions of plagiarism in real scenarios. © 2013 Wiley Periodicals, Inc. Comput Appl Eng Educ 23:13–22, 2015; View this article online at wileyonlinelibrary.com/journal/cae; DOI 10.1002/cae.21571

Keywords: plagiarism; computer programming; source code

INTRODUCTION

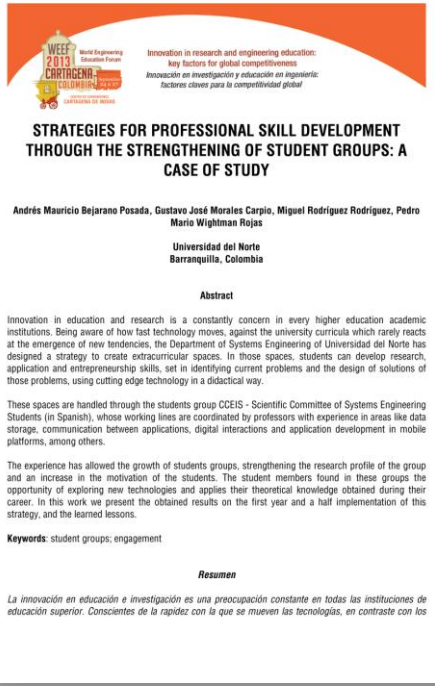
Software development requires each programmer to generate creative ideas that provide computational solutions for problems with certain degrees of complexity. There are many academic programs that require an introductory course in computer programming. The purpose of these courses is to develop the students ability to implement logical solutions to a problem (represented via a certain algorithmic scheme) using a particular programming language. Such skills are evaluated through the assignment of exercise problems that the student has to solve in an algorithmic fashion using programming code. During the academic evaluation process, the course instructor must verify that the student's solution code complies with the required restrictions and that the provided solution is unique and clearly differentiable from the solutions provided by his fellow classmates.

The inherent complexity of the course described above, as well as the conceptual, ethical, and pedagogical limitations of such courses, produces situations in which plagiarism can occur. To obtain a satisfactory grade, certain students commit this type of fraud, which is initially manifested in the modification of certain lines of code. However, as the student's knowledge increases, this type of behavior becomes more elaborate and therefore more difficult to detect for the class instructor. To detect these cases, a one by one comparison of students' assignments is required, which is a grueling task with a high degree of complexity. Additionally, the procedure must take into account which of the different conditions that define a certain specific situation allow the case to be considered as plagiarism.

To control plagiarism in computer code developed in academic environments, the use of simple Unix commands such as grep, diff, and wc [1] up to more complex tools such as GPLAG [2], JPLAG [3], MOSS [4], SHERLOCK [5], and YAP3 [6] has been implemented. Several implementations have been introduced for software plagiarism detection using token-based analysis [7], algorithm filter [8], and the analysis from the point of view of the teacher [9,10]. Different techniques have been implemented like Natural Language Processing (NPL) [11], Singular Value Decomposition (SVD) over algorithms [12], fingerprint based distance measure [13], analysis from the generated assembler code [14], program characterization [15], and adaptive local alignment of keywords [16]. Although these tools are aimed at detecting similarity in computer programs, this detection is focused on the lexical similarity (word for word) of the codes. However, this type of similarity is insufficient when identifying those cases where copying involves the essence of the program, with a more than superficial modification of the original source code [1]. For this reason, there is a need to develop a tool that facilitates the detection of those cases where the similarity is present at both

Correspondence to E. E. Zurek (ezurek@uninorte.edu.co).

© 2013 Wiley Periodicals, Inc.



STRATEGIES FOR PROFESSIONAL SKILL DEVELOPMENT THROUGH THE STRENGTHENING OF STUDENT GROUPS: A CASE OF STUDY

Andrés Mauricio Bejarano Posada, Gustavo José Morales Carpio, Miguel Rodríguez Rodríguez, Pedro Mario Wightman Rojas

Universidad del Norte
Barranquilla, Colombia

Abstract

Innovation in education and research is a constantly concern in every higher education academic institutions. Being aware of how fast technology moves, against the university curricula which rarely reacts at the emergence of new tendencies, the Department of Systems Engineering of Universidad del Norte has designed a strategy to create extracurricular spaces. In those spaces, students can develop research, application and entrepreneurship skills, set in identifying current problems and the design of solutions of those problems, using cutting edge technology in a didactical way.

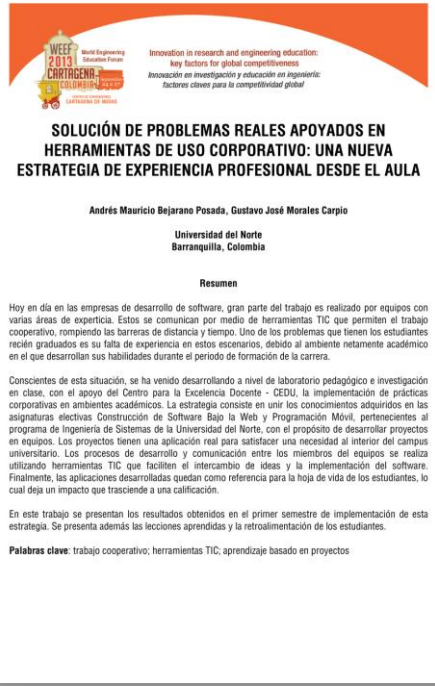
These spaces are handled through the students group CCEIS - Scientific Committee of Systems Engineering Students (in Spanish), whose working lines are coordinated by professors with experience in areas like data storage, communication between applications, digital interactions and application development in mobile platforms, among others.

The experience has allowed the growth of students groups, strengthening the research profile of the group and an increase in the motivation of the students. The student members found in these groups the opportunity of exploring new technologies and applies their theoretical knowledge obtained during their career. In this work we present the obtained results on the first year and a half implementation of this strategy, and the learned lessons.

Keywords: student groups; engagement

Resumen

La innovación en educación e investigación es una preocupación constante en todas las instituciones de educación superior. Conscientes de la rapidez con la que se mueven las tecnologías, en contraste con los



SOLUCIÓN DE PROBLEMAS REALES APOYADOS EN HERRAMIENTAS DE USO CORPORATIVO: UNA NUEVA ESTRATEGIA DE EXPERIENCIA PROFESIONAL DESDE EL AULA

Andrés Mauricio Bejarano Posada, Gustavo José Morales Carpio

Universidad del Norte
Barranquilla, Colombia

Resumen

Hoy en día en las empresas de desarrollo de software, gran parte del trabajo es realizado por equipos con varias áreas de experticia. Estos se comunican por medio de herramientas TIC que permiten el trabajo cooperativo, rompiendo las barreras de distancia y tiempo. Uno de los problemas que tienen los estudiantes recién graduados es su falta de experiencia en estos escenarios, debido al ambiente netamente académico en el que desarrollan sus habilidades durante el periodo de formación de la carrera.

Conscientes de esta situación, se ha venido desarrollando a nivel de laboratorio pedagógico e investigación en clase, con el apoyo del Centro para la Excelencia Docente - CEDU, la implementación de prácticas corporativas en ambientes académicos. La estrategia consiste en unir los conocimientos adquiridos en las asignaturas electivas Construcción de Software Bajo la Web y Programación Móvil, pertenecientes al programa de Ingeniería de Sistemas de la Universidad del Norte, con el propósito de desarrollar proyectos en equipos. Los proyectos tienen una aplicación real para satisfacer una necesidad al interior del campus universitario. Los procesos de desarrollo y comunicación entre los miembros del equipos se realiza utilizando herramientas TIC que faciliten el intercambio de ideas y la implementación del software. Finalmente, las aplicaciones desarrolladas quedan como referencia para la hoja de vida de los estudiantes, lo cual deja un impacto que trasciende a una calificación.

En este trabajo se presentan los resultados obtenidos en el primer semestre de implementación de esta estrategia. Se presenta además las lecciones aprendidas y la retroalimentación de los estudiantes.

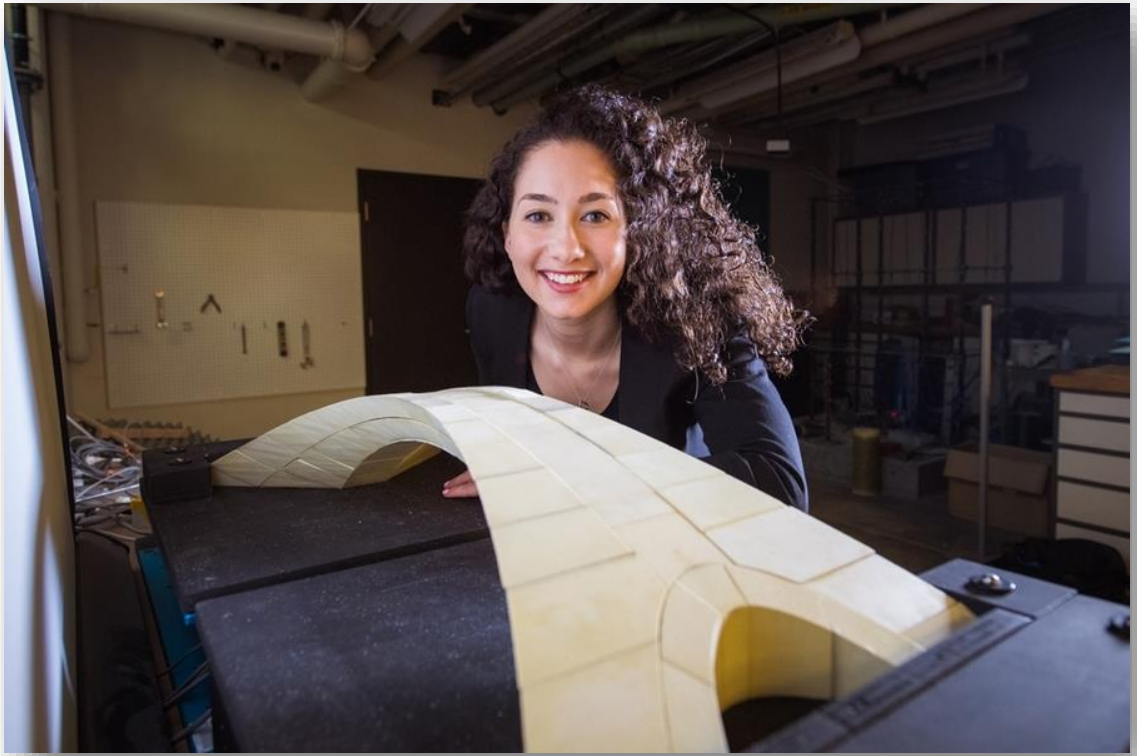
Palabras clave: trabajo cooperativo; herramientas TIC; aprendizaje basado en proyectos

Supervising Students Beyond Class Projects

- Had students that worked in **Deep Learning and Image Processing**, and **Interactive Animations**.
- Currently advising the **Flutter Dev Club**.
- Previous student projects showed during **Bogota SIGGRAPH** 2012 and 2013.

Research In Geometric Design

- Topological Interlocking (concave and convex geometries).
- General Interlocking Design.
- Differential Growth.



Leonardo's Bridge

https://news.mit.edu/2019/leonardo-da-vinci-bridge-test-1010

Thanks!

Any questions ?

You can find me at

- http://andresbejarano.name/
- abejara@purdue.edu