Murray Patterson
1 Park Place, Room 636
Atlanta, Georgia 30303 (USA)
mpatterson30@gsu.edu
murraypatterson.github.io

Teaching Statement and Philosophy

Computer Science is about problem solving and abstract reasoning, while being very hands-on at the same time – involving the development of a working implementation for a specific task, and it is important for students to have both this reasoning and the more concrete skills of coding up a solution. Regarding the first aspect, I want to teach students to think about a problem at an abstract level, so they are able to come up with a model for solving it that is independent from any programming language. While the second, more concrete aspect comes with experience, I believe that design practices, *e.g.*, writing modular programs, and approaches for debugging code are important guides in this journey. For non-majors or first year students, what is also important is stressing that computing skills are something worthwhile to have in any discipline, and that they too are capable of having. To achieve these goals, I take a very interactive approach to teaching, and try to create an environment where discussion is encouraged. This certainly stems from my undergraduate experience at a small, mostly teaching-focused university, and I have found it quite effective on several accounts, in my own teaching. Engaging students to get involved in the discussion stimulates the students to learn, rather than to lose focus or to be bored in the classroom. Moreover, it also allows me to asses if the students are indeed learning the material, or how I could better explain something. I illustrate this with the following experiences.

The best example of this is the most recent, since taking a role as Visiting Assistant Professor at Fairfield University in September 2019. In the fall semester gone by, I taught an introductory programming course, an intermediate programming course focusing on object-oriented design, as well as a graduate course in database systems. These three courses were a nice cross-section of the courses offered at Fairfield, and indeed the students had different backgrounds and goals as a result. The introductory course was aimed at first year computer science students, but also non-majors – in all cases, it would be their first programming course. Because of this, my approach was to nurture basic computational thinking and problem solving skills, but also of writing programs in Java and C. I had a Midterm Assessment of Teaching (MAT) for this course – a service offered by the Center of Academic Excellence (CAE) at Fairfield University. Key positive points that came from this were that I am very approachable and helpful with questions, and that they liked that I would go over the homework after it was due, in a live coding session where I would code up a solution as I discussed my reasoning and answered any questions which would arise. The intermediate programming course was a core second year course for computer science majors. Here, the students were very good programmers, and so the focus was on the concepts of object-oriented design. This also meant weekly programming assignments which I went over in a live coding session, focusing even more on the concepts behind the solution. Finally, the grad course was composed mostly of young professionals who already completed an undergraduate degree. As a result, they were highly-skilled, independent, and very motivated to get a high grade towards their master's degree. For this reason, I made the course challenging, while making it project-focused to allow the students to use their own strengths to obtain a good result.

Student evaluations for these courses were positive overall. I obtained an overall mean score (over all courses) of 3.81 on a scale of 1 (poor) to 5 (excellent) for the evaluation "Overall, I rate this instructor as an excellent teacher", while I obtained an overall mean score of 3.95 for "Overall, I rate this course as excellent".

In my first important teaching experience, at Simon Fraser University, I was the TA for a third year course on functional and logic programming – a course I enjoy because it offers a different perspective on programming from the imperative approach learned in earlier years. Indeed, it allows the students to reflect upon what it means to solve a computational problem – not tied to a particular language. The biggest surprise was, contrary to my undergraduate experience at a small university, in this much larger university, the amount of one-on-one time the students had with the professor was very little. I hence felt the need to ensure that the students were as comfortable with the course material as I was during my undergrad experience. I did this by making myself available at the office hours, advertising this availability and even organizing problem-solving sessions before exams – also a way to see if students were prepared for the exam. The professor of this course kept me for the entire time of my master's degree because the students gave me good reviews each semester. What is interesting is that I now teach this course myself at Fairfield University in the spring semester of 2020.

I taught a section of mathematics for the life sciences to first-year students during my time in Lyon. Teaching in French was certainly a challenge, however the greater challenge was due to cultural differences and expectations. In the classroom, the students only spoke when spoken to, and there seemed to be a great deal of social pressure not to make a mistake. While this was the most disciplined group of students I ever taught, there lacked a certain level of interaction, making it difficult to asses if the students were learning the material, and which concepts required more attention. For this reason, I took extra care to nurture an environment where discussion was encouraged by asking questions during the lesson, and offering to go into more detail if people seemed to have trouble with something. After several sessions, the students became more comfortable with this style, and even started to ask questions themselves. The students who asked the most questions and were most involved in discussing the problems where the ones who tended to perform better on the exams.

During my postdoc at the University of Milano-Bicocca, I occasionally supervised students together with my supervisor. This has been a great experience on how to properly organize a project, so that we are both working as effectively as possible with our respective skillsets and experiences. This typically involves discussing the big ideas of the project together, while understanding the specific tasks this will entail. While not getting into the details of these tasks, the challenge is then to effectively delegate and direct these tasks, and being available, should the student be stuck on something, or have technical difficulties. Last semester, I supervised a project student on helping out with some aspects of our haplotyping software (`whatshap.readthedocs.io`), and another on cancer progression models. These are two of the major research projects I am currently involved in, and now they too have had the chance to be involved, and to contribute to these research-level activities. Both of these students have since enrolled in the master's program after this experience.

Some of my volunteer experiences in teaching have been very valuable. My supervisor at the University of Milano-Bicocca helps organize software carpentry (`software-carpentry.org`) workshops, where the goal is to teach important basic computing skills to researchers in any field. I have helped out with several workshops now: one on Unix tools and Python for automating data

analysis workflows, and another on SQL and Python for data science. In both cases, my role was to answer any questions and to resolve any technical issues during the hands-on part of the sessions. It was an eye-opening experience to understand, and to help out with the difficulties people have with the utilities that I am accustomed to and use every day – I think this has helped me in all walks of life, from explaining what I do to a layperson, to teaching introductory programming courses. I have since been invited to participate in a software-carpentry training session, so that I may be certified to hold my own workshops in the future. I have also volunteered at an international school, advising highschool students in a math program which culminates in an annual conference where they present their work. It is a interesting experience to introduce students to advanced concepts that they would otherwise encounter years later – it provides great insights on how to better explain things at a broader and more general level, engaging a wider audience. Finally, in addition to the MATs, the CAE at Fairfield University offers many workshops around teaching and advising, including invited speakers which come to talk about topics such as active learning. Being a small undergraduate university that is highly rated for its teaching, many professors are very dedicated to and well-versed in education. These workshops, and interactions with such professors have really opened my mind towards good teaching practices, and how I can improve mine. In particular, the value of the active learning pedagogy: while I see that my interactive approach to teaching incorporates some of this, I also realize how much further I can move in this direction – something I plan to do in the coming years.

I believe that my teaching and research experiences in various institutions and countries have exposed me to a diverse set of environments and cultural settings. It has allowed me to be comfortable with, to value and to promote a diverse and inclusive learning environment. These experiences have helped me to be flexible in my teaching and are a great resource for building a balanced research team – something I believe an interdisciplinary field of study such as bioinformatics needs, since it intersects with real problems such as health and medicine which affect all demographics. From the above teaching experiences, I can teach introductory and intermediate programming courses to majors or non-majors, mathematics courses, database systems, as well as alternative programming paradigms and courses concerning computational thinking. Given my research background, I can teach courses in bioinformatics, algorithms and optimization, as well as software design. My educational background allows me also to teach numerical analysis, computational logic and computability, as well as combinatorics and complexity theory. More recently, from experiences in my own research, and from the software carpentry workshops, I see a great need for reproducibility in many experimental fields. I strive to make my own research reproducible, writing easy-to-use and modular data analysis pipelines (see, `hapchat.algolab.eu`), and I think teaching a course on reproducibility and open science would have an important positive impact. In all of these cases, an interactive approach to teaching has been effective, and also leads to a more pleasant classroom experience for everyone. I plan to continue incorporating interactive approaches into my teaching, and with my recent, more in-depth exposure to the active learning pedagogy, I realize how many more ideas and techniques I can incorporate – something I am focused on exploring more deeply, and implementing in my own teaching.