

Clément Aubert
Department of Computer Science
Appalachian State University
Boone, NC 28608
☎ 828-278-4620
☎ 828-262-2386
✉ aubertc@appstate.edu
📁 cs.appstate.edu/~aubertc/

The Department of Computer Science
Bowdoin College Brunswick,
Maine 04011

December 27, 2016

Dear Members of the Search Committee,

I am writing to apply for the assistant professor position at the Department of Computer Science of Bowdoin College.

I am currently an instructor and post-doctoral researcher at Appalachian State University. Prior to moving to the United States, I had two positions in France as post-doctoral researcher one in Computer Science and the other in Mathematics. My Ph.D. dissertation and my prior education—I came to Theoretical Computer Science through Mathematics and Philosophy—are both diversified and rich, displaying evidence of my strong ability to adapt to different positions and environments. I believe this rich and diversified experience and my perspectives are strong assets to serve the Bowdoin College.

I have taught many different aspects of Computer Science in a variety of environments and crafted an array of teaching capacities that, I believe, would be useful for the University. I am currently teaching an introduction to programming class using Java, possess a strong experience in teaching multiple core areas (among which are software engineering or databases), and have a personal taste for Web programming (html, php, css) and system administration (operating systems, networking, embedded systems). Last but not least, I am interested in teaching theoretical aspects of Computer Science (automata, complexity, and mathematical logic) and Mathematics lectures. I am eager to introduce new contents and tools in my courses, being particularly sensitive to new methods that our field and students calls for. Having walked the line between multiple fields and having a good knowledge of European pedagogic techniques uniquely position me to experiment with innovative and global pedagogic methods, and to engage a diverse body, under the guidance of the University.

Computer Science is constantly evolving, and I can at the same time listen to the student's needs and provide guidance to them as a teacher, researcher, and advisor. The software engineering classes I was responsible for early in my career (Spring 2011 & 2012) trained me to guide multiple groups of students (± 8) in parallel, to monitor their progress, and to guide without imposing a direction. I believe this is a solid basis to strengthen my skills as a mentor and an advisor, and to guide students with tactfulness and discernment. My policy is to always keep an open door to students and to reply as promptly as possible to their emails; I keep the dialogue always active with my current ± 75 students. Strengthening this practice and enriching it by taking an active part at Bowdoin College would be for me an exciting opportunity to make my pedagogical principles in complete harmony with the way I practice my profession. I am self-driven and know how to build from scratch ambitious lectures that meet the Bowdoin College requirements and insert harmoniously in the curriculum, and want students to be offered the opportunity to influence the progression of our class.

Throughout my scholarly career, I actively pursued a research program of my own and successfully contributed to pre-existing programs. My research activities embrace multiple aspects of the semantics of programming language, and span from complexity theory to concurrency, from automata theory to parametricity. This compendium of expertise draws its strength from active and ongoing collaborations with an international network. My current co-authors are in Japan, Denmark, France, and the United States. This is an opportunity for this department to strengthen its international visibility, and to complete and extend its areas of expertises.

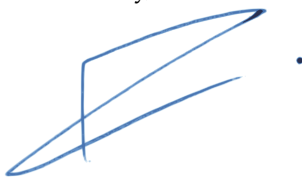
I have invested energy and time in understanding grants mechanisms, both in Europe and in the United States. I obtained small grants, totalizing $\pm \$3k$, and planted the seeds for bigger achievements. I am already in touch with the National Science Foundation (NSF), took an active part in the writing of mid-term and final term evaluations of *Agence Nationale de la Recherche* projects (the French equivalent to NSF projects), and was the sole Principal Investigator of an application to the prestigious European Marie Skłodowska-Curie Individual Fellowships. There is

no NSF grant that I could apply to, as a non-tenured non-citizen researcher, but I provided feedback on proposals and observed the process closely. These skills and my knowledge of European fundings could benefit the department in the near future.

I always volunteer to teach and mentor students, and feel that my experience in undergraduate teaching would be useful to enhance the current course offerings and educational experiences at Bowdoin College. I like to face demanding students and to connect them with the changing roles of computers in society, to guide them through their own intellectual journey and job search, and to benefit from my cultural competencies to approach a diverse student body. I would also contribute to the Bowdoin College by pursuing my own scholarly program, that is varied and rich, and comes with its own national and international networks. I believe that my experience in undergraduate teaching, including monitoring students' projects, my ongoing collaborations and my strong willingness to embrace the principles of community of Bowdoin College make me a strong candidate for the position.

Please find enclosed the material required in the position description. I will gladly provide any other supporting material such upon request. I thank you for your consideration and look forward to hearing from you.

Sincerely,



Clément Aubert

Please find attached the following documents:

- *A curriculum vitæ (pp. 3–7)*
- *A research statement (pp. 8–9)*
- *A teaching statement (pp. 10–12)*
- *A statement of contributions to diversity (pp. 13–14)*
- *Names and addresses of three references (p. 15)*

Clément Aubert

Curriculum Vitæ

Department of Computer Science
 Appalachian State University
 Boone, NC 28608
 ☎ 828-278-4620
 ☎ 828-262-2386
 ✉ aubertc@appstate.edu
 🏠 cs.appstate.edu/~aubertc/

Research Interests

Programming languages, Type theory, Complexity, Automata, Concurrent systems, Category theory

Current & Previous Positions, Education

- 2015–present **Post Doctoral Researcher & Instructor**, *Appalachian State University, Computer Science Department*, funded by the NSF grant 1420175 and the College of Arts & Science. Supervisor: Patricia Johann.
- 2014–2015 **Post Doctoral Researcher & Instructor**, *INRIA, SPADES – Université Paris-Est (UPEC, Paris 12), Laboratoire d’Algorithmique, Complexité et Logique (LACL)*, funded by the ANR Rever & Faculté des Sciences et technologie. Supervisor: Daniele Varacca.
- 2013–2014 **Post Doctoral Researcher**, *CNRS – Aix-Marseille Université, Institut de Mathématiques de Marseille (I2M) – UMR 7373 CNRS, Logique de la Programmation (LDP) team*, funded by the ANR ReCré. Supervisor: Myriam Quatrini.
- 2010–2013 **Ph.D. in Computer Science, with the highest honours (“mention Très honorable”)**, “LINEAR LOGIC AND SUB-POLYNOMIAL CLASSES OF COMPLEXITY”, *École Doctorale Galilée (146) – Université Paris 13 – Laboratoire d’Informatique de Paris Nord (LIPN), UMR 7030*, Supervisors: Stefano Guerrini and Virgile Mogbil.
- 2009–2010 **M.S. in Mathematics, cum laude**, *Université Paris 7 – Denis Diderot*. “Mathematical Logic and Computer Science Foundations” (LMFI), specializations in Proof Theory (P.-L. Curien) and Lambda-Calcul (T. Joly).
- 2007–2010 **B.S. in Philosophy, specialization in Logic**, *Université Paris 1 – Panthéon-Sorbonne*.
- 2005–2006 **Three-years Degree in History**, *Université de Reims*.
- 2000–2005 **Baccalauréat in sciences with distinction in German, then Khâgne & Hypokhâgne (Preparatory classes to the grandes écoles, focused on Humanities)**, *Jean-Jaurès – Reims*.

Teaching Experiences

-  cs.appstate.edu/~aubertc/teaching_effectiveness.pdf
- ASU **CS1440 Computer Science I**, in Java, Beginner, Fall 2016
- UPEC **Initiation to Algorithm and Complexity**, in C, Intermediate, Spring 2015
- Imperative Programming**, in C, Beginner, Fall 2014
- Paris 13 **Databases**, using SQL, Intermediate, Spring 2013
- System Administration & Network**, Intermediate, Spring 2012 & Spring 2013
- Software Engineering**, Beginner, Spring 2011 & Spring 2012
- Algorithms and Programming**, in C, Beginner, Fall 2011
- New Ways of Learning Mathematics**, Beginner, Fall 2010

Grants & Awards

- \$1k Research Development Travel Grant from Appalachian State University. Sole PI, 2016
- \$1.5k PICS: Logique linéaire et applications award to visit U. Dal Lago in Bologna, 2014
- \$0.5k GDR IM’s “Visiting PhD student” program. PI shared with another Ph.D. student, 2011
- \$100k Funding for my Ph.D. from the *École doctorale Galilée*, 2010

Publications

 [lacl.fr/~caubert/#publications](https://github.com/lacl/fr/~caubert/#publications)

Co-authors Marc Bagnol, Ioana Cristescu, Paolo Pistone, and Thomas Seiller were Ph.D. students or recent Ph.D. laureates at the time of our collaborations. Patricia Johann and Daniele Varacca are full professors.

Refereed Journals

- 2017 **C. Aubert** and I. Cristescu. “Contextual equivalences in configuration structures and reversibility”. In: *Journal of Logical and Algebraic Methods in Programming* 86.1, pp. 77–106. ISSN: 2352-2208. DOI: 10.1016/j.jlamp.2016.08.004.
- 2016 **C. Aubert** and T. Seiller. “Characterizing co-NL by a group action”. In: *Mathematical Structures in Computer Science* 26, pp. 606–638. ISSN: 1469-8072. DOI: 10.1017/S0960129514000267.
- 2016 **C. Aubert** and T. Seiller. “Logarithmic space and permutations”. In: *Information & Computation* 248. Communication at DICE 2013 and LCC 2013, pp. 2–21. ISSN: 0890-5401. DOI: 10.1016/j.ic.2014.01.018.

Selective Conferences

- 2016 **C. Aubert**, M. Bagnol, and T. Seiller. “Unary Resolution: Characterizing Ptime”. In: *Foundations of Software Science and Computation Structures (FOSSACS 2016)*. Vol. 9634. Lecture Notes in Computer Science. Springer, pp. 373–389. DOI: 10.1007/978-3-662-49630-5_22. Acceptance rate: 27.4%.
- 2014 **C. Aubert** and M. Bagnol. “Unification and Logarithmic Space”. In: *Rewriting and Typed Lambda Calculi (RTA-TLCA 2014)*. Vol. 8650. Lecture Notes in Computer Science. Springer, pp. 77–92. DOI: 10.1007/978-3-319-08918-8_6. Acceptance rate: 35% (31/87).
- 2014 **C. Aubert**, M. Bagnol, P. Pistone, and T. Seiller. “Logic Programming and Logarithmic Space”. In: *12th Asian Symposium on Programming Languages and Systems (APLAS 2014)*. Vol. 8858. Lecture Notes in Computer Science. Springer, pp. 39–57. DOI: 10.1007/978-3-319-12736-1_3.

Workshops with Proceedings

- 2015 **C. Aubert** and I. Cristescu. “Reversible Barbed Congruence on Configuration Structures”. In: *8th Interaction and Concurrency Experience (ICE 2015)*. Vol. 189. Electronic Proceedings in Theoretical Computer Science, pp. 68–95. DOI: 10.4204/EPTCS.189.7.
- 2011 **C. Aubert**. “Sublogarithmic uniform Boolean proof nets”. In: *Proceedings Second Workshop on Developments in Implicit Computational Complexity*. Ed. by J.-Y. Marion. Vol. 75. Electronic Proceedings in Theoretical Computer Science, pp. 15–27. DOI: 10.4204/EPTCS.75.2.

Articles Submitted to Refereed Journals

- 2015 **C. Aubert** and M. Bagnol. “Unification and Logarithmic Space”. In: *Logical Methods in Computer Science, special issue of RTA/TLCA 2014*.

Dissertations

- 2013 **C. Aubert**. “Linear Logic and Sub-polynomial Classes of Complexity”. PhD thesis. Université Paris 13–Sorbonne Paris Cité. 182 pp. Supervisors: Stefano Guerrini, Virgile Mogbil (UMR CNRS 7030 — Paris 13).
- 2010 **C. Aubert**. “Réseaux de preuves booléens sous-logarithmiques”. MA thesis. LIPN: LMFI, Paris VII. 29 pp. Supervisors: Virgile Mogbil, Paulin Jacobé de Naurois (UMR CNRS 7030 — Paris 13).
- 2009 **C. Aubert**. “L’élimination des coupures dans la Logique des Domaines Constants”. MA thesis. Paris 1. 29 pp. Supervisor: Jean-Baptiste Joinet (UMR CNRS 7126 — Paris 7).

Research Reports

- 2015 **C. Aubert**. *An in-between “implicit” and “explicit” complexity: Automata*. Research Report. 5 pp. Communication at DICE 2015.
- 2015 **C. Aubert**, M. Bagnol, and T. Seiller. *Memoization for Unary Logic Programming: Characterizing Ptime*. Research Report RR-8796. INRIA. 28 pp. arXiv: 1501.05104 [cs.LG].

In Preparation

- 2016 **C. Aubert**. “The Implicit Computational Complexity Flavor of Automata”. Journal version of [13].
- 2016 **C. Aubert** and I. Cristescu. “Weak Contextual Equivalences in Configuration Structures and Reversibility”. Extension of [1].
- 2016 **C. Aubert** and P. Johann. “Bifibrational parametricity for polymorphism and computational effects”.
- 2016 **C. Aubert** and D. Varacca. “Processes, Systems and Tests: Three Layers of Concurrent Computation”.

Reviews and PC

PC Member

- Logic and Computational Complexity (LCC 2016)
- Developments in Implicit Computational Complexity (DICE 2015)

Reviewer for **Conferences**

- 27th International Conference on Concurrency Theory (CONCUR 2016)
- Conference on Foundations of Software Science and Computation Structures (FoSSaCS 2015 and 2016)
- Automata, Languages and Programming (ICALP 2015)
- Italian Conference on Theoretical Computer Science (ICTCS 2014)





Journals

- Journal of Logical and Algebraic Methods in Programming
- Mathematical Structures in Computer Science
- Information & Computation

Societies

- American Mathematical Society
- zbMATH




Languages

- French  Mother tongue, skills in typography
- English  Perfectly read and understood, fluent speaker, TOIEC’s score: 975
- German  Goethe Institut’s *Zertifikat Deutsch* in 2002, specialization and distinction in high school
- Russian  Basic concepts

Computer Skills

- O.S.** Linux, Mac OS, Windows
- Prog.** Java, C, SQL, shell scripts, notions in COQ, Prolog and Python
- ⒶT_EX** Daily use, drawings with TikZ
- Web dvpment** HTML5, CSS3, W3C’s specifications, WAI, PHP, MySQL

International Visits & Exchanges

- 2014  Visit to J. G. Simonsen to work on Implicit Computational Complexity and Algebraic characterizations of complexity classes – Datalogisk Institut, Copenhagen, funded by the COLA Project (1 week)
- 2012  Visit to U. Dal Lago to work on Quantum Calculus, Geometry of Interaction and Implicit Complexity – FoCUS, Bologna, funded by the PICS project *Logique linéaire et applications* (2 weeks)
- 2011  Exchange with T. Seiller to work on Operator algebra and Complexity – Institut Mathématiques de Luminy, Université d’Aix-Marseille, funded by the GDR-IM (1 week)

Invited Talks (Selection)

 lacl.fr/~caubert/#exposes

- 2015 ■ ■ Elica meeting – Laboratoire d’Informatique de Paris Nord (LIPN) – Université Paris 13
- 2014 ■ ■ Algorithmic, Complexity and Logic Laboratory (LACL) seminar – Université Paris-Est Créteil (UPEC)
- ■ Logic, Computer Science, and Discrete Mathematics (LIMD) seminar – Laboratoire de Mathématiques de l’Université de Savoie (LAMA), Université de Savoie
- ■ Logique de la Programmation (LDP) seminar – Institut de Mathématiques de Marseille (I2M), Aix-Marseille Université
- ■ Seminar of the *Methodes formelles* team – Laboratoire lorrain de recherche en informatique et ses applications (Loria), Université de Lorraine
- 2013 ■ ■ International Workshop *Logic and Computational Complexity* (LCC) – Turin
- ■ Complexité, Logique et Informatique (CLI) seminar – Équipe de Logique Mathématique, Université Paris 7
- ■ Young Researchers’ seminar – LIPN
- 2012 ■ ■ 9th project meeting of the ANR Implicit Computational Complexity, Concurrency and Extraction (Complice) – LIPN
- ■ LDP seminar – Institut de Mathématiques de Luminy (IML), Aix-Marseille Université
- ■ Foundations of Component-based Ubiquitous Systems (FoCUS) meeting – Università Di Bologna
- ■ Logique, Calcul et Raisonnement (LCR) seminar – LIPN
- ■ Logic and Interactions 2012 – Centre International de Rencontres Mathématiques (CIRM)
- 2011 ■ ■ International Workshop Second Workshop on Developments in Implicit Computational Complexity (Dice 2011) – Saarbrücken, ETAPS 2011
- ■ Multidisciplinary research group *Vérité et preuves* – Université Paris 1
- ■ 16th meeting of the Logique, Algèbre et Calcul (LAC) group – Preuves, Programmes, Systèmes (PPS), Université Paris 7

Memberships

ANR-funded projects correspond to NSF-funded projects, and involvement in ANR projects implies active scientific collaborations, travel funding, invitation to Schools and scientific discussions.

- Productive member
- ANR Expanding Logical Ideas for Complexity Analysis (ELICA)
 - ANR Realizability for Classical Logic, Concurrency, References and Rewriting (Recré)
 - ANR Logic and Geometry of Interaction (Logoi)
 - ANR Implicit Computational Complexity, Concurrency and Extraction (Complice)
 - Ph.D. Students working group *Vérité et preuves*
 - Research Group in Mathematical Computer Science (GDR-IM)
 - ANR Programming reversible and dependable systems (Rever)
- Attending member
- ANR Computing with Quantitative Semantics (Coquas)
 - ANR Parallel and Distributed Analysis (Panda)
 - Curry-Howard: Logic and Computation (Chocola) meetings

Summer & Winter Schools

- 2016 ■ ■ **From Theory to Practice of Algebraic Effects and Handlers**, Dagstuhl Seminar 16112, with 6 days lectures by A. Bauer, A. Filinski, G. Plotkin, A. Simpson, ...
- 2014 ■ ■ **Mathematical Structures of Computation, Concurrency, Logic and Types** – Lyon, with lectures by 5 days U. Dal Lago, M Hofmann, L. Ong, D. Sandiorgi, ...

- 2014 ■■ **Sémantique des preuves et des programmes et formalisation des mathématiques** – Luminy, 12 days with lectures by A. Miquel, T. Coquand, P.-L. Curien, ...
- 2012 ■■ **Réalisabilité à Chambéry #5** on Realizability – Bourget du Lac, with lectures by A. Miquel, 4 days M. Hofmann, J.-L. Krivine, H. Herbelin, ...
- 2011 • **Workshop on Linear Logic** on Geometry of Interaction, Traced Monoidal Categories and Implicit Complexity – Kyoto, with lectures by J.-Y. Girard, S. Guerrini, U. Dal Lago, ...
- 2011 🇺🇸 **10th Annual Oregon Programming Languages Summer School** on Types, Semantics and Verification – Eugene, with lectures by H. Herbelin, X. Leroy, P. Melliès, B. Pierce, D. Scott, ...
- 2011 ■■ **ANR Logoi Summer School** on Geometry of Interaction, Operator Algebra – Carry-le-Rouet, 3 days with lectures by P.-L. Curien and J.-Y. Girard.

Services

- 2012 & 2016 Responsible for courses: composing and grading quizzes, homeworks and exams, coordination with the department, leading role in the end-term evaluation.
- 2014 Member of the board for the [Agence Nationale de la Recherche \(ANR\)](#) during the evaluation of the Logic and Geometry of Interaction (Logoi) project.
- 2012 Member of the board for the [Evaluation Agency for Research and Higher education \(AERES\)](#) during the evaluation of the Laboratoire d'Informatique de Paris Nord (LIPN), Paris 13.

Civic Engagement

- [micr0lab](#) Founding member, member of the board, webmaster, and active participant in this ten-member non-benefit association that exists since 2011.
- [La goutte d'Ordi](#) Volunteer (2010–2013) three to six hours per week: teaching classes and lab sessions to newly arrived immigrants, trying to bridge the digital divide.
- [French Data Network](#) Adherent, user and active member of this historical Internet provider acting for net neutrality.

Clément Aubert
 Department of Computer Science
 Appalachian State University
 Boone, NC 28608
 ☎ 828-278-4620
 ☎ 828-262-2386
 ✉ aubertc@appstate.edu
 🏠 cs.appstate.edu/~aubertc/

Research Statement

Dear Members of the Search Committee,

My research focuses on giving tools to design better programming languages, to (i) ease the programmer’s tasks, (ii) provide certifications, and (iii) design optimized compilers. To this end, I develop **semantics of programming languages**, that reveals the underlying mathematical structure of programs.

Past Achievements

Making the Mathematics Dynamic

In a first approximation, Mathematics objects are static (i.e., $2 + 3 = 5$ independently of any “execution”) and Computer Science studies dynamic objects, programs that need to be evaluated to produce a result (i.e., $2 + 3$ *reduces* to 5). However, Mathematics can be made dynamics by considering as an object of study proofs and *rewriting of proofs*. Different programs can have the same input / output behavior, and, similarly, different proofs can prove the same statement under the same hypothesis; this analogy is made precise by the so-called “proof-as-program correspondence”. Thanks to this isomorphism, abstract models of computation endowed with a precise cost model can be represented by proof systems. Starting from here, it is possible to impose limitations on the typing system, i.e., on the valid rules of the proof system, and to constrain the expressiveness of the corresponding class of programs.

Using a quantitative version of this fruitful correspondence between programs and proofs, I was able to prove, with Ph.D. students and recent Ph.D. laureates, that “balanced” logic programs are as expressive as 2-ways multi-head automata [3], and that further allowing unary logic program corresponded to the addition of a stack [2]. It is known that the former class of program captures Logarithmic Space, whereas the latter characterizes Polynomial Time, and this is where complexity theory kicks in. The limitations of those basics models of computation actually ensures that their corresponding proof systems will produce only program of such or such complexity. Thanks to Implicit Computational Complexity, you know *by design* that your programming language won’t let you write programs that are going to need more than a given bound on space or time. By abstracting away implementation details, we managed to design high-level programming languages that are reliable and certified.

Making the Programs Static

Semantics of programs also allows to “flatten” programs. Imagine a concurrent scenario (resources and data are shared) where programs can compute both forward and backward (i.e., we are in a reversible setting). Mathematics, and more precisely configuration structures, gives a tool to express all the possible scenario that could result from the interaction of multiple reversible programs. By writing down all the events that can occur and their connections when a program is confronted with a specific environment, you can decide whenever two programs have the same observational behavior: if the structures of the events provoked by a reversible program is the same as the structure given by another program, then we know that those two programs—no matter their actual content—are equivalent.

Actually, one can be finer than this: interesting relations on such structures determine a variety of quotients on programs [4, 1]. Whereas syntactical comparison of reversible and concurrent programs is nearly impossible and most of the time irrelevant, their semantics comparison gives instantly a reliable verdict. By tweaking what the environment can observe, one can model man-in-the-middle attacks, saturated channels, or add quantitative reasoning. By knowing whenever two part of programs are equivalent, and which events are going to occur, one can also cut dead branches in programs: this work leads to guarantees on the absence of deadlock and open the way to optimized compilation.

Research Projects

The most theoretical aspects of my previous achievements reached an exciting stage of maturity. They can support grant proposals, be extended, move toward more practical realizations, but also be the starting point of thrilling students projects.

Each one of those lines support interesting and relevant extensions. For instance,

- We isolated conditions on logic programs and proof system that correspond to a particular complexity classes twice, and could illustrate the reproducibility and usability of our theory by isolating other complexity classes (Polynomial Space (**PSPACE**) seems reasonably accessible).
- Having a more fine-grained definition of the difference between environments and processes would provide more insights on equivalences on reversible and concurrent processes.

Those perspectives and the international networks that come with each one of them, could support timely and ambitious projects. For instance, the [NSF 16565](#) solicitation seems accessible, since it targets young researchers willing to develop independent research and to integrate students. Regarding this aspect, each one of those lines could be turned into an object of study for motivated students:

- The correspondence between automata and logic programs offers a nice and clear set-up to understand what it means for a model of computation to simulate another. This would be a theoretical, but accessible, subject of research, that could lead to a publication in an international workshop.
- Basics knowledge in reversible and concurrent programming might soon become vital in the job market, and reversible imperative programming language such as [Janus](#) could be an interesting object of study, in complement of an initiation to multi-threading in Java for instance.

I am a passionate researcher that likes to share his enthusiasm and to work on multiple projects at the same time. I don't conceive my profession as being two-sided, and have always imagined my career as a blend of theoretical progresses and practical experiences, both shared with students. Being driven by my own program, I think I have the potential to pursue my own research, and would be excited to have the opportunity to actively nurture and exchange with the research led at Bowdoin College.

References

- [1] **C. Aubert** and I. Cristescu. “Contextual equivalences in configuration structures and reversibility”. In: *Journal of Logical and Algebraic Methods in Programming* 86.1 (2017), pp. 77–106. ISSN: 2352-2208. DOI: 10.1016/j.jlamp.2016.08.004.
- [2] **C. Aubert**, M. Bagnol, and T. Seiller. “Unary Resolution: Characterizing Ptime”. In: *Foundations of Software Science and Computation Structures (FOSSACS 2016)*. Vol. 9634. Lecture Notes in Computer Science. Springer, 2016, pp. 373–389. DOI: 10.1007/978-3-662-49630-5_22. Acceptance rate: 27.4%.
- [3] **C. Aubert** and M. Bagnol. “Unification and Logarithmic Space”. In: *Logical Methods in Computer Science, special issue of RTA/TLCA 2014* (2015).
- [4] **C. Aubert** and I. Cristescu. “Reversible Barbed Congruence on Configuration Structures”. In: *8th Interaction and Concurrency Experience (ICE 2015)*. Vol. 189. Electronic Proceedings in Theoretical Computer Science. 2015, pp. 68–95. DOI: 10.4204/EPTCS.189.7.

Clément Aubert
 Department of Computer Science
 Appalachian State University
 Boone, NC 28608
 ☎ 828-278-4620
 ☎ 828-262-2386
 ✉ aubertc@appstate.edu
 📁 cs.appstate.edu/~aubertc/

Teaching Statement

Dear Members of the Search Committee,

Since the beginning of my academic career, I have always taught and learned; the chemistry of a good student-teacher exchange continuously re-enforces my ability to analyze, communicate, and question myself. I have taught Computer Science and Mathematics, in formal lectures, small groups sessions, and lab sessions, in Universities and *Institut Universitaire de Technologie* (i.e., technical college) with different characteristics both in France and in the US; I hence know how to face a large variety of trainings, levels, interests, cultures and learning styles. Being confronted with the challenge of teaching diverse groups of students has helped me to foster (i) my own teaching practices, and (ii) qualities to engage students.

Teaching Practices

Having taught formal lectures, lab sessions and small classes gave me the opportunity to embrace our mission as a whole. Lectures and lab share the same goal with different emphasis, and I slightly adapt my way of teaching to take full benefit from the difference in their set-up. Small, project-oriented, inquiry-based classes requires different techniques.

During Lectures

When I'm facing large audience, two of my main objectives are to teach *how* to learn, and *why*.

- I believe every student is entitled to a clear and accessible syllabus that makes every lecture *predictable*; the unwinding must be natural and logical. Providing the students with detailed and marked materials gives them the possibility to *foresee* and *look back*, so that our progression is acknowledged and contextualized. Once this base, this toolbox, has been set, students can grab the tools made available and learn how to use them by practicing and exercising.
- Opening my lectures to the outside world constantly reminds students of the impact and scope of their knowledge. They are multiple ways to insert the lecture in a wider range:
 - Using the Fibonacci sequence as an example in an undergraduate lecture connects to Mathematics.
 - Setting aside time to explain the main mechanism of the Heartbleed security bug to intermediate students in a Networking lecture relates to hot topics.
 - Sketching how Implicit Computational Complexity solves problems graduate students are facing in a programming class exposes to scholarly activities.

In practice, I always teach on the board, to maximize interactions, and ask students to formulate themselves what we learned at the end of every lectures.

During Lab Sessions

In my experience, many students, especially in undergraduate studies, are facing problems that are unrelated to the content we are addressing. Troubles with basics Filesystem Hierarchy Standard, file naming conventions, usage of interfaces, make it difficult for students to organize their efforts. Too much time and energy are wasted by students googling again and again variations of the same keywords. I chose to address this dimension of the lab sessions and I now always introduce students to general methodology regarding

- How to organize their workflow (by using keyboard shortcuts, appropriate software and efficient interfaces),
- Where to find documentation and resources (man pages, official websites, [Stackexchange](#) websites, handbooks, specifications), and how rewarding it can be to read a manual and the log messages,

- How to keep track of the resources consulted (from simple text file to bookmarks, or others reference management softwares),
- How to automate redundant tasks (scripting, file synchronizers, mailing-lists),
- How to share, backup and version data (using [git](#), [svn](#), [Etherpad](#)).

Again, I believe that providing students with a solid basis to construct their knowledge is the key to a successful learning, and the best way to form orderly students.

Setting Goals

My global philosophy is to spell out our goals and to aim big. Computer Science is always taught incrementally, every lecture standing on the previous ones; and I know that we need to lay solid foundations to co-construct everlasting knowledges, and to make the most out of our time spent together.

- To monitor their understanding of the lectures, I quiz them on a regular basis and adapt the lectures accordingly. Answering short and simple questions helps them to isolate the core of what they are learning and to understand what are the lecture's expectations. Beside, asking them to write code by hand on a regular basis helps them to focus on logical structures more than on semicolons, and prepare them to the professional world; refer for instance to the thread "[Hand-writing code on exams](#)" on the [SIGCSE mailing-list](#), where it is reported that major companies like Google or Microsoft ask interviewees to write code on board.
- I want students to be self-driven and to understand how *they* can make a difference in the world. Using Internet and some imagination, it is easy to offer them projects at every level with concrete impacts: developing a website for a local non-profit, coding a bot for Wikipedia, contributing to open-source projects are just a couple of examples among endless possibilities.

Engaging Students

Finding the appropriate tone, leveling the discourse to the audience without lowering the expectations is the key to a successful lecture. Having been a volunteer (2010–2013) in a non-profit association (*la Goutte d'Ordinateur*), in one of the most multi-cultural and poor neighborhoods of Paris taught me to be patient, comprehensive, to use plain but precise vocabulary, and to work with complete beginners. This helped me greatly in the academic world to coach students groups as a whole as well as to engage individuals: I can anticipate special needs, adopt different etiquette to respect them. I take pleasure in adapting lessons and explanations to diverse bodies, and pay great care to regularly take the pulse of the audience using quizzes, questions, evaluations, and by asking directly for feedback.

My personal reflexion on being a teacher and a researcher taught me that students need to develop

1. **Independence**, to sharpen their capacities to solve problems by themselves
2. **Self-efficacy**, to believe in their capacities but also to identify when to ask for help and how to obtain it
3. **Curiosity**, to drive *their own* intellectual journey

And to me, the best way to provide students that support is:

1. **To value skills**, more than facts, by observing how students try to solve problems and suggest improvements.
2. **To respect their individuality**, by always celebrating their victories, and leaving the door (and the inbox!) constantly open to their doubts, questions and joys.
3. **To encourage interactions**, by letting them drive an interactive and peer learning.

One way to construct this support is to use collaborative platforms to supply all the material in [markdown](#) format, so that students can easily edit, comment, print, display and export it. Making the content and the source code available offers multiple advantages: students can focus on *understanding*, instead on taking notes, they are provided with examples of the sort of deliverables I expect from them, they are given coding samples to hack and to appropriate, they are invited to write their own exercises. From the teacher's point of view, a collaborative platform offers a continuous stream of feedback that helps to adjust the content of the lectures, and gives opportunities to provide direct access to student's code.

I am ready to teach introductory and technical classes on a variety of subjects (programming, networking, operating systems, algorithmic, databases) and eager to develop more advanced courses, such as functional programming, automata theory, mathematical logic or complexity. My specific care to methodology and my education also make me perfectly suitable to teach software engineering, proof theory, or any Mathematical-related lecture. Finally, having

multiple areas of expertise allows me to use research tools as teaching medium, and to mentor graduate students with a variety of research subjects that will foster their independence and provide good support for scholarship research.

I would be excited to be given the opportunity to design new and ambitious classes, using modern tools, but I also know how to observe and insert my course in a pre-existing syllabus. I care for the success of students, but also know how to handle a group as a whole, and how to insert lectures in departments. My curiosity leads me to explore with the course catalog of my Departments and to care about cohesion and harmony among colleagues. Finally, my enthusiasm drives me to experiment innovative teaching and to offer to students the best tools to shape long-lasting skills.

Clément Aubert
 Department of Computer Science
 Appalachian State University
 Boone, NC 28608
 ☎ 828-278-4620
 ☎ 828-262-2386
 ✉ aubertc@appstate.edu
 🏠 cs.appstate.edu/~aubertc/

Statement of Contributions to Diversity

Dear Members of the Search Committee,

Being a white male with both parents teachers places me on the privileged side of the society. I benefited from this origin and the possibilities it offered to study, but was conscious of the opportunity I was given, and of the inequities in our society. As an immigrant in the United States, my experiences and origins make me contribute to the diversity of my workplaces, and my will to achieve equity makes me care about diversity. The environments I worked in helped me to raise awareness about my methods of teaching.

My French origin arouses curiosity and I can, when I feel confident, slip a French word in my lecture: it can be a variable name, an example of `String`, or a cultural reference. This helps me in two ways. First, by breaking the ice and addressing my French origins, I can connect with students about who I am and who they are. Secondly, it helps me putting our field into context: most students, for instance, are surprised when they discover that programming languages are not translated in French. Acknowledging this fact helps them understand the cultural advantage they have when learning programming languages, for instance, because `for`, `while`, etc., are native words to them.

I was a volunteer (2010–2013) in the non-profit association *Goutte d'Ordinateur*, in the “Goutte d'Or”, one of the most multi-cultural and poor neighborhood of Paris, with an estimated 28% of foreigners [2, p. 35] and almost 50% of unemployment rate [2, p. 36]. My goal was to reduce the digital divide through weekly free lessons and sessions with newly arrived migrants: some of them never used a keyboard, some of them had difficulties with French, most of them barely had a practical use of cellphones and smartphones. This 2-years experience taught me to be patient, comprehensive, to use plain but precise vocabulary, and to work with complete beginners. This helped me greatly in my teaching experiences: I could anticipate some special needs, knew different etiquettes and paid attention to respect them.

I was also in multicultural environments during my professional activities. I worked three years in the University of Paris 13, located in Villetaneuse, in Paris' suburbs, and faced an over-representation of modest or disadvantaged backgrounds students, comparatively to other Universities [4, p. 33]: 40% of the students come from socially disadvantaged backgrounds, and nearly 20% are holding a foreign *Baccalauréat* [1, p. 15]. In Créteil, where I taught during a year, the *Université Paris-Est Créteil* (U-PEC) attracts over 1000 foreigners students each year [3, p. 48] and integrates them to the population of its region. This helped me to give body to my principles of tolerance and openness.

My cultural competencies and sensibilities also cover gender-related questions. I was a student in the pioneer program in gender studies in France (in Paris 8): being asked to think about our relations between genders early in my life pushed me to care about it, and to understand how to acknowledge the discrepancy between our genders. I am truly committed to working toward achieving equity between genders, and am particularly attentive to it.

References

- [1] Agence d'évaluation de la recherche et de l'enseignement supérieur. *Rapport d'évaluation de l'université Paris 13*. 2013. URL: <http://www.aeres-evaluation.com/content/download/21997/337535/file/AERES-S1-Paris13.pdf> (visited on 11/28/2015).
- [2] Atelier Parisien d'Urbanisme. *Paris 1954-1999 Données statistiques, Population, logements, emploi. 18ème arrondissement*. Sept. 2005. URL: http://www.apur.org/sites/default/files/documents/A_18.pdf (visited on 11/21/2015).

- [3] Université Paris-Est Créteil Val-de-Marne. *UPEC - Rapport d'activités 2014*. Ed. by L. Hittinger. URL: http://www.u-pec.fr/servlet/com.univ.collaboratif.utils.LectureFichiergw?ID_FICHIER=1259768749086 (visited on 11/25/2015).
- [4] Université de Paris 13. *Paris 13 en chiffres 2012 / 2013*. URL: <http://www.univ-paris13.fr/images/paris13en-chiffres2014.pdf> (visited on 05/16/2013).

Clément Aubert
 Department of Computer Science
 Appalachian State University
 Boone, NC 28608
 ☎ 828-278-4620
 ☎ 828-262-2386
 ✉ auberc@appstate.edu
 📁 cs.appstate.edu/~auberc/

Professional References

Stefano GUERRINI

Full Professor
 Head of the [Logique, Calcul et Raisonnement \(LCR\) team](#)
Ph.D. advisor (2010–2013)
 Laboratoire d'Informatique de Paris-Nord
 CNRS, UMR 7030
 Institut Galilée – Université Paris 13
 99, avenue Jean-Baptiste Clément
 93430 Villetaneuse
 France
 ☎ +33 1 49 40 28 05
 ✉ stefano.guerrini@univ-paris13.fr
 📁 <https://lipn.univ-paris13.fr/~Guerrini/>
 📄 stefanoguerrini

Laure PETRUCCI

Full Professor
 Head of the [Laboratoire d'Informatique de Paris Nord \(LIPN\)](#)
Supervisor of my teaching activities (2010–2013)
 Laboratoire d'Informatique de Paris Nord (LIPN)
 CNRS, UMR 7030
 Institut Galilée – Université Paris 13
 99, avenue Jean-Baptiste Clément
 93430 Villetaneuse
 France
 ☎ +33 1 49 40 35 79
 ✉ laure.petrucci@lipn.univ-paris13.fr
 📁 <https://lipn.univ-paris13.fr/~petrucci/>

Patricia JOHANN

Full Professor
Post-Doc supervisor (2015–current)
 Computer Science Department
 Belk Hall 312M
 Appalachian State University
 Boone, NC 28608
 United States of America
 ☎ 828-262-7008
 ✉ johannp@appstate.edu
 📁 <https://cs.appstate.edu/~johannp/>
 📄 patriciaj09