Curriculum vitae

Prof. Brigitte Pientka
School of Computer Science
3480 University St.
McGill University
Montreal, Quebec, H3A 2A7

bpientka@cs.mcgill.ca
http://www.cs.mcgill.ca/∼bpientka
Fax: (514) 398-3883
Phone : (514) 398-2583

RESEARCH INTERESTS
Foundations of Programming Languages; Logical Frameworks; Type Systems; Functional Programming; Logic Programming; Logic; Automated Deduction

ACADEMIC APPOINTMENT

Associate Professor (tenured) since July 2009
School of Computer Science,
McGill University, Montreal, Canada

Assistant Professor August 2003 – June 2009
School of Computer Science,
McGill University, Montreal, Canada

EDUCATION

Carnegie Mellon University, Pittsburgh, USA December 2003
Ph.D. in Computer Science
Dissertation: *Tabled Higher-order Logic Programming*
Advisor: Prof. Frank Pfenning

Technical University Darmstadt, Germany
Diplom (M.S.) with honors in Computer Science July 1997

Technical University Darmstadt, Germany
Vordiplom (B.S.) in Computer Science July 1993

VISITING POSITIONS

LIX, Ecole Polytechnique, Paris, France
Visiting Researcher July 2010

Inria, Rocquencourt, France
Visiting Researcher January 2010 – June 2010

Computer Science Department, Cornell University, Ithaca, USA
Visiting Research Scholar September 1997 – August 1998

Computer Science Department, University of Edinburgh, UK
Visiting Student September 1993 – September 1994

HONORS

Best Student Paper Award, *19th Intern. Conference on Logic Programming*, 2003

Siebel Scholar, 2002

Gottlieb Daimler – Karl Benz Fellow, 2001

Graduate Research Fellowship, Carnegie Mellon University, 1998 – 2003

Gottlieb Daimler – Karl Benz Foundation Scholarship, 1997 – 1998

Various awards for attending conferences: Conference Scholarship; Association of Logic Programming (2002); GSA Conference Funds, Carnegie Mellon University (2002); Woody Bledsoe Travel Award, CADE (2001); Grace Hopper Conference Scholarship (2000); Conference Scholarship, Intel Foundation (1997).

GRANTS

Individual research grants

Reported grants awarded include no overhead, i.e.,the total amount is available to fund research activities, in particular to contribute towards stipends of non-scholarship graduate students. The National Science and Research Council of Canada allows a grant holder to fund a PhD student with $21,000 per year (maximum) and Msc students with $17,500 per year (maximum). These awards may be supplemented with possible teaching assistantships funded through the university.

• *NSERC Discovery Grant - Accellerator Supplement*, $120,000 total, $40,000 per year, 2012 - 2015, Awarded to exceptionally promising scientists in Canada each year; in 2012 there were 9 computer scientists in Canada who were selected.

• *NSERC Discovery Grant*, $140,000 total, $28,000 per year, 2012 – 2017.

• *PSR-SIIRI Projets conjoints de recherche et d'innovation, Ministère du Dèveloppement économique, de l'Innovation et de l'Exportation* $150,000 total for three years (2012 – 2015)

• *63e session de la Commission permanente de coopération franco-quèbècoise, Ministère du Dèveloppement économique, de l'Innovation et de l'Exportation* $15,000 , 2011 – 2013
Travel grant for participating in Inria Équipe Associée with Dale Miller (Inria) and Kaustuv Chaudhuri (Inria).

• *NSERC Discovery Grant*, $135,000 total, $27,000 per year, 2007 – 2012.

GRANTS (CONT.)

Individual research grants (cont.)

- *Vice-Principal Research Office Grant*, $24,000 total, 2005 – 2007.
- *NSERC Discovery Grant*, $77,100 total, $25,700 per year, 2004 – 2007.
- *FQRNT Nouveau Chercheur Grant*, $46,200 total, $15,400 per year, 2004 – 2007.
- *Startup grant*, School of Computer Science, McGill University, $50,000 total, 2003 – 2009.

Team grants

- *NSERC Strategic Network on Engineering Complex Software Intensive Systems*, $7,300,000 total allocation to the network 2010 – 2015
PI: Tom Maibaum (McMaster) in partnership with General Motors (GM). I am eligible to apply for for the funding of specific projects under this partnership grant.
- *FQRNT Recherche d'Equipe*, $152,100 total, $50,700 per year, 2008 – 2011,
Co-PI: Stefan Monnier (Université de Montréal)
- *Canada Foundation for Innovation New Opportunities Grant*, $394,142 total, 2005 – 2009. Team members: Joelle Pineau, Martin Robillard.

MEMBERSHIP

Association for Automated Reasoning (AAR)

Association for Computing Machinery (ACM)

Association for Logic Programming (ALP)

PUBLICATIONS Co-authors who are students or post-docs are highlighted in bold face.

Refereed Journals

[J9] Andreas Abel and Brigitte Pientka. Well-founded Recursion with Copatterns: A Unified Approach to Termination and Productivity *Journal of Functional Programming* (featuring best papers of ICFP'14), 2015 (accepted)

[J8] **Olivier Savary Belanger**, Stefan Monnier and Brigitte Pientka. Programming Type-safe Transformations using Higher-order Abstract Syntax *Journal of Formalizaed Reasoning*. 8(1): 49 – 91, 2015.

[J7] Amy Felty, Alberto Momigliano, Brigitte Pientka. The Next 700 Challenge Problems for Reasoning with Higher-Order Abstract Syntax Representations: Part 2 – A Survey *Journal of Automated Reasoning*. 55(4): 307–372, 2015.

[J6] Brigitte Pientka. Everything You (N)ever Wanted to Know about LF Type Reconstruction. *Journal of Functional Programming*. 23(1):1 – 37, 2013.

[J5] Brigitte Pientka. Higher-order Term Indexing using Substitution Trees *ACM Transactions on Computational Logic*. 11(1):1–40, 2009. (*This is an extended journal version of [C7].*

[J4] Aleksandar Nanevski, Frank Pfenning, Brigitte Pientka. Contextual Modal Type Theory. *ACM Transactions on Computational Logic* 9(3):1–60, 2008.

[J3] Brigitte Pientka. Verifying Termination and Reduction Properties about Higher-order Logic Programs. *Journal of Automated Reasoning* 34(2):179–207, 2005. (*This is an extended journal version of [C4]*)

[J2] Christoph Kreitz and Brigitte Pientka. Connection-driven Inductive Theorem Proving. *Studia Logica*, 69(2):293–326, 2001. (*This is an extended journal version of [C3]*).

[J1] Brigitte Pientka and Christoph Kreitz. Automating Inductive Specification Proofs. *Fundamenta Informatica*, 39(1–2):189–209, 1999. (*This is an extended journal version of [C2]*) .

Refereed Conferences

[C26] Brigitte Pientka and **Andrew Cave**. Inductive Beluga: Programming Proofs (System Description) *25th International Conference on Automated Deduction (CADE'15)*, Lecture Notes in Computer Science (LNCS 9195). pages 272–281, Springer, 2015.

[C25] Brigitte Pientka, Andreas Abel. Structural Recursion over Contextual Objects *13th International Conference on Typed Lambda Calculi and Applications (TLCA'15)*, pages 273-287, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (LIPICS), 2015

PUBLICATIONS (CONT.)

Refereed Conferences (cont.).

[C24] **Francisco Ferreira** and Brigitte Pientka. Bi-direcional Elaboration of Dependently Typed Programs *16th International Symposium on Principles and Practice of Declarative Programming (PPDP'14)*, pages 161–174, ACM Press, 2014.

[C23] Anton Setzer, Andreas Abel, Brigitte Pientka, **David Thibodeau**. Unnesting of Copatterns. *Joint 25th International Conference on Rewriting Techniques and Applications and 12th International Conference on Typed Lambda Calculi and Applications (RTA-TLCA'14)*, Lecture Notes in Computer Science (LNCS 8560), pages 31–45, Springer, 2014.

[C22] **Andrew Cave**, **Francisco Ferreira**, Prakash Panangaden, Brigitte Pientka. Fair Reactive Programming. *41th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL'14)*, pages 361–372, ACM Press, 2014.

[C21] **Olivier Savary Belanger**, Stefan Monnier and Brigitte Pientka. Programming Type-safe Transformations using Higher-order Abstract Syntax. *3rd International Conference on Certified Programs and Proofs (CPP'13)*, Lecture Notes in Computer Science (LNCS 8307), pages 243–258, Springer, 2013.

[C20] Andreas Abel, Brigitte Pientka. Well-founded Recursion with Copatterns:A Unified Approach to Termination and Productivity. *18th ACM SIGPLAN International Conference on Functional Programming (ICFP'13)*, pages 185–196, ACM Press, 2013.

[C19] Andreas Abel, Brigitte Pientka, **David Thibodeau**, and Anton Setzer. Copatterns: Programming Infinite Structures by Observations. *40th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL'13)*, pages 27–38, ACM Press, 2013.

[C18] **Andrew Cave** and Brigitte Pientka. Programming with Binders and Indexed Data-types. In M. Hicks, editor, *39th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL'12)*, pages 413–424, ACM Press, 2012.

[C17] Andreas Abel and Brigitte Pientka. Higher-Order Dynamic Pattern Unification for Dependent Types and Records. In L. Ong, editor, *10th International Conference on Typed Lambda Calculi and Applications (TLCA'11)*, Lecture Notes in Computer Science (LNCS), pages 10–26, Springer, 2011.

[C16] Brigitte Pientka and **Joshua Dunfield**. Beluga: A Framework for Programming and Reasoning with Deductive Systems (System Description). In J. Giesl and R. Hähnle, editor, *5th International Joint Conference on Automated Reasoning (IJCAR'10)*, Lecture Notes in Computer Science (LNCS 6173), pages 15–21, 2010.

[C15] Amy P. Felty and Brigitte Pientka. Reasoning with Higher-Order Abstract Syntax and Contexts: A Comparison. In M. Kaufmann and L. C. Paulson, editor, *First International Conference on Interactive Theorem Proving (ITP'10)*, Lecture Notes in Computer Science (LNCS 6172), pages 227–242, 2010.

PUBLICATIONS (CONT.)

Refereed Conferences (cont.).

[C14] Brigitte Pientka and **Joshua Dunfield**. Programming with Proofs and Explicit Contexts. In S. Antoy, editor, *10th International ACM SIGPLAN Symposium on Principles and Practice of Declarative Programming (PPDP'08)*, pages 163–173, 2008. ACM Press.

[C13] Brigitte Pientka. A Type-theoretic Foundation for Programming with Higher-order Abstract Syntax and Explicit Substitutions. In G.C. Necula and P. Wadler, editors, *35th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'08)*, pages 371–382, 2008. ACM Press.

[C12] Brigitte Pientka. Proof Pearl:The Power of Higher-order Encodings in the Logical Framework LF In K. Schneider and J. Brandt, editor, *20th International Conference on Higher-order Logics (TPHOLs'07)*, Lecture Notes in Computer Science (LNCS 4732), pages 246-261, Springer, 2007.

[C11] **Samuli Heilala** and Brigitte Pientka. Bidirectional Decision Procedures for the Intuitionistic Propositional Modal Logic **IS4** In F. Pfenning, editor, *21st International Conference on Automated Deduction (CADE'07)*, Lecture Notes in Computer Science (LNCS 4603), pages 116–131, Springer, 2007.

[C10] Brigitte Pientka. Eliminating Redundancy in Higher-order Unification: a Lightweight Approach In U. Furbach and N. Shankar, editor, *3rd International Conference on Automated Reasoning (IJCAR'06)*, Lecture Notes in Computer Science (LNCS 4130), pages 361–377, Springer, 2006.

[C9] **Susmit Sarkar**, Brigitte Pientka and Karl Crary. Small Proof Witnesses for LF In M. Gabrielli, G. Gupta, editor, *21st International Conference on Logic Programming (ICLP'05)*, Lecture Notes in Computer Science (LNCS 3668), pages 387–401, Springer, 2005.

[C8] Brigitte Pientka. Tabling in Higher-order Logic Programming In R. Niewenhuis, editor, *20th International Conference on Automated Deduction (CADE'05)*, Lecture Notes in Artificial Intelligence (LNAI 3632), pages 54–69, Springer, 2005.

[C7] Brigitte Pientka. Higher-order Substitution Tree Indexing In C. Palamidessi, editor, *19th International Conference on Logic Programming (ICLP'03)*, Lecture Notes in Computer Science (LNCS 2916), pages 377–391, Springer, 2003. **Best Student Paper Award**.

[C6] Brigitte Pientka and Frank Pfenning. Optimizing Higher-order Pattern Unification In F. Baader, editor, *19th International Conference on Automated Deduction (CADE'03)*, Lecture Notes in Computer Science (LNCS 2741), pages 473–487, Springer, 2003.

[C5] Brigitte Pientka. A Proof-theoretic Foundation for Tabled Higher-order Logic Programming. In P. Stuckey, editor, *18th International Conference on Logic Programming (ICLP'02)*, Lecture Notes in Computer Science (LNCS), pages 271–286. Springer, 2002.

PUBLICATIONS (CONT.)

Refereed Conferences (cont.).

[C4] Brigitte Pientka. Termination and Reduction Checking for Higher-order Logic Programs. In T. Nipkow, R. Gore, A. Leitsch, editor, *First International Joint Conference on Automated Reasoning (IJCAR'01)x*, Lecture Notes in Artificial Intelligence (LNAI 2083), pages 401–415. Springer, 2001. (*A preliminary version of this paper appeared in [W1]*)

[C3] Christoph Kreitz and Brigitte Pientka. Matrix-based Inductive Theorem Proving. In R. Dyckhoff, editor, *9th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods (TABLEAUX)*, Lecture Notes in Artificial Intelligence (LNAI 1847), pages 294–308. Springer Verlag, 2000.

[C2] Brigitte Pientka and Christoph Kreitz. Instantiation of Existentially Quantified Variables in Inductive Specification Proofs. In J. Plaza and J. Calmet, editors, *4th International Conference on Artificial Intelligence and Symbolic Computation*, Lecture Notes in Artificial Intelligence (LNAI 1476), pages 247–258. Springer, 1998.

[C1] Stefan Gerberding and Brigitte Pientka. Structured Incremental Proof Planning. In Brewka, G. Habel, C. Nebel, B: *Proceedings of the 21st Annual German Conference on Artificial Intelligence (KI-97): Advances in Artificial Intelligence*, Freiburg, Germany, Lecture Notes in Artificial Intelligence (LNAI 1303), pages 63–74, Springer, 1997.

Refereed Workshops

[W11] Amy Felty, Alberto Momigliano, and Brigitte Pientka. An Open Challenge Problem Repository for Systems Supporting Binders. In I. Cervesato, K. Chaudhuri, editors, *10th Workshop on Logical Frameworks and Meta-languages (LFMTP'15): Theory and Practice*, pages 3-18, EPTCS, 2015.

[W10] **Andrew Cave** and Brigitte Pientka. A Case Study on Logical Relations using Contextual Types. In I. Cervesato, K. Chaudhuri, editors, *10th Workshop on Logical Frameworks and Meta-languages (LFMTP'15): Theory and Practice*, pages 18-33, EPTCS, 2015.

[W9] **Andrew Cave** and Brigitte Pientka. First-class Substitutions in Contextual Type Theory. In A. Momigliano, B. Pientka, and R. Pollack, editors, *ACM SIGPLAN Workshop on Logical Frameworks and Meta-languages (LFMTP'13): Theory and Practice*, pages 15–24, ACM Press, 2013.

[W9] **Francisco Ferreira**, Stefan Monnier and Brigitte Pientka. Compiling Contextual Objects: Bringing Higher-Order Abstract Syntax to Programmers. In A .Abel and T. Sheard, editors, *7th ACM SIGPLAN Workshop on Programming Languages meets Program Verification (PLPV'13)* , pages 13–24, ACM Press, 2013.

PUBLICATIONS (CONT.)

Refereed Workshops (cont.).

[W8] **Mathieu Boespflug** and Brigitte Pientka. Multi-level Contextual Type Theory. In G. Nadathur and H. Geuvers, editors, *International Workshop on Logical Frameworks and Meta-Language:Theory and Practice*, Electronic Proceedings in Theoretical Computer Science, pages 1–15, 2011.

[W7] Andreas Abel and Brigitte Pientka. Explicit Substitutions for Contextual Type Theory,. In K. Crary and M. Miculan, editors, *International Workshop on Logical Frameworks and Meta-Language:Theory and Practice (LFMTP'10)*, Electronic Proceedings in Theoretical Computer Science (EPTCS), vol 34, page 5–21, 2010.

[W6] **Joshua Dunfield** and Brigitte Pientka. Case Analysis on Higher-order Data. In A. Abel and C. Urban, editors, *International Workshop on Logical Frameworks and Meta-Language:Theory and Practice (LFMTP'08)*, Electronic Notes in Theoretical Computer Science (ENTCS), 228: 69-84 (2009).

[W5] Brigitte Pientka, **Florent Pompigne** and **Xi Li**. Focusing the Inverse Method for LF: a Preliminary Report. In C. Schürmann and B. Pientka, editors, *International Workshop on Logical Frameworks and Meta-Language:Theory and Practice (LFMTP'07)*, Electronic Notes in Theoretical Computer Science 196: 95-112 (2008).

[W4] Brigitte Pientka. Functional Programming with Higher-order Abstract Syntax and Explicit Substitutions In A. Stump and Hongwei Xi, editor, *Programming Languages meets Program Verification*,Electronic Notes in Theoretical Computer Science (ENTCS), pages 41–60, Elsevier, 2007.

[W3] Aleksandar Nanevski, Brigitte Pientka and Frank Pfenning. A modal Foundation for Meta-variables In F. Honsell and M. Miculan and A. Momigliano, editors, *2nd Workshop on Mechanized Reasoning about Languages with Variable Binding (MERLIN'03), Uppsala, Sweden*, ACM Press, pages 159–180, 2003.

[W2] Brigitte Pientka. Memoization-based Proof Search in LF: an Experimental Evaluation of a Prototype. In *Third International Workshop on Logical Frameworks and Meta-Languages (LFM'02), Copenhagen, Denmark*, Electronic Notes in Theoretical Computer Science (ENTCS), Volume 70, Issue 2, pages 1–14, 2002.

[W1] Brigitte Pientka and Frank Pfenning. Termination and Reduction Checking in the Logical Framework. In C. Schürmann, editor, *Workshop on Automation of Proofs by Mathematical Induction, Pittsburgh, USA*, pages 1-15, 2000.

PUBLICATIONS (CONT.)

Edited Journals

[EJ2] Special issue: Best papers of the 22nd International Conference on Automated Deduction. Renate Schmidt and Brigitte Pientka, guest editors. Journal of Automated Reasoning 47(2): 107-109, 2011.

[EJ1] Special issue: Intuitionistic Modal Logics and Applications Valeria de Paiva and Brigitte Pientka, guest editors. Information and Computation 209(12): 1435-1436, 2011.

Edited Proceedings

[E4] Proceeding of the 9th International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice, Vienna, Austria. Amy Felty and Brigitte Pientka, editors. ACM Press, 2014.

[E3] Proceeding of the 8th International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice, Boston, USA. Alberto Momigliano, Brigitte Pientka and Randy Pollack, editors. ACM Press, 2013.

[E2] Proceeding of the 2nd International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice, Bremen, Germany. Brigitte Pientka and Carsten Schürmann, editors. Electronic Notes in Theoretical Computer Science (ENTCS), Elsevier, 2007.

[E1] Proceeding of the International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice. Brigitte Pientka and Alberto Momigliano, editors. Electronic Notes in Theoretical Computer Science (ENTCS), Elsevier, 2006.

Book Chapters and Invited Contribution

[B4] Brigitte Pientka. Programming Inductive Proofs: a New Approach Based on Contextual Types. *Induction, Verification, and Termination– - Festschrift for Christoph Walther on the Occasion of His 60th Birthday*, Lecture Notes in Computer Science (LNCS 6463), 1–16 pages, Springer, 2010.

[B3] Brigitte Pientka. Beluga: Programming with Dependent Types, Contextual Data, and Contexts (Invited talk). In M. Blume and N. Kobayashi and G. Vidal, editor, *10th International Symposium in Functional and Logic Programming (FLOPS'10), Sendai, Japan*, Lecture Notes in Computer Science (LNCS 6009), 1–12 pages, Springer, 2010.

[B2] Brigitte Pientka. Overcoming Performance Barriers: Efficient Verification Techniques for Logical Frameworks (Invited Tutorial) In S. Etalle and M. Truszczynski, editor, *22nd International Conference on Logic Programming (ICLP'06), Seattle, USA*, Lecture Notes in Computer Science (LNCS 4079), 3–10 pages, Springer, 2006.

PUBLICATIONS (CONT.)

Book Chapters and Invited Contributions (cont.).

[B1] Christoph Kreitz, Jens Otten, Stephan Schmitt, and Brigitte Pientka. Matrix-based Constructive Theorem Proving. In Steffen Hölldobler, editor, *Intellectics and Computational Logic. Papers in honor of Wolfgang Bibel*, Applied Logic Series, vol. 19, pages 189–205. Kluwer, 2000.

Submitted

[S3] **Andrew Cave**, Brigitte Pientka. Mechanizing Proofs with Logical Relations – Kripke-style (submitted to *Mathematical Structures in Computer Science* for a special issue on Logical Frameworks and Meta-Languages in Jan 2016).

[S2] **David Thibodeau**, **Andrew Cave**, Brigitte Pientka. Indexed Codata Types. submitted in July 2015.

[S1] Andreas Abel, Brigitte Pientka, Anton Setzer, **David Thibodeau**. Unnesting Copatterns (submitted to *Logical Methods in Computer Science* in March 2015).

Theses

[T3] Brigitte Pientka. Tabled Higher-order Logic Programming. PhD thesis, Technical report CMU-CS-03-185, Carnegie Mellon University, Dec 2003.

[T2] Brigitte Pientka. Structuring and Optimizing Incremental Proof Planning. Master's thesis, Technical University Darmstadt, 1997.

[T1] Brigitte Pientka. A Heuristic for Case analysis. Undergraduate thesis, Technical Paper 37, Department of Artificial Intelligence, University of Edinburgh, 1995.

Other Publications (Gender Studies/Outreach)

[G2] Brigitte Pientka. How to Prove It. In *Proceeding of the Third International Conference on Women, Work and Computerization, Bonn, Germany*. Springer, 1997.

[G1] Brigitte Pientka. Ist die Informatik männlich? Eindrücke, Erfahrungen, Forderungen. In A. Paul-Kohlhoff and C. Walter, editors, *"Eine Frau, die Maschinenbau studiert ist kein Wesen vom Mars . . . " – Studentinnen motivieren Schülerinnen*, volume 17. Darmstädter Beiträge zur Berufspädagogik, 1996.

SOFTWARE SYSTEMS

- **Twelf**: A Logical Framework for Formal Systems

Major contributions to the **Twelf** system and developer of **Twelf 1.5R1** a tabled higher-order logic programming environment. The system is one of the most successful systems in supporting specification of formal systems and proofs about them. Among the universities and research groups using Twelf are: Carnegie Mellon University, Princeton, Yale, Cornell University, University of Pennsylvania, and many other universities.

- **Beluga**: A Functional Programming and Proof Language supporting Higher-order Abstract Syntax

The system is currently under active development at McGill University and supports programming with proofs and higher-order data. It may be viewed as an alternative to the **Twelf** system, and is based on the publications [C16, C13, C14, C18, C26]. Latest release July 2015.

TALKS

All refereed and published conference and workshop papers listed previously were presented at their respective meeting. Below, I only list invited talks at conferences and workshops where I have been explicitly invited to give a special lecture. In 2015, I was invited to give the plenary invited talk for the conference "Certified Proofs and Programs" which I had to decline for personal reasons.

Invited Tutorials

[T1] Beluga: Programming proofs about formal systems (3h) 25th International Conference on Automated Deduction (CADE-25), Berlin, Germany, 2015.

[T2] "Overcoming performance barriers: efficient verification techniques in logical frameworks". 22nd International Conference on Logic Programming(ICLP), Seattle, USA, Invited tutorial (1h plenary talk), August 2006.

Invited Lectures at Conferences and Workshops

[L41] "Mechanizing Meta-Theory in Beluga". Second International Workshop on Rewriting Techniques for Program Transformations and Evaluation, Warsaw, Poland, July 2015.

[L40] "Programming logical relations proofs". Certification of high-level and low-level programs, Paris, France, July 2014.

[L29] "Beluga:Programming with contextual objects, contexts, and ...". International Workshop on Logical Frameworks and Meta-Languages:Theory and Practice, Copenhagen, Denmark, Sep 2012.

[L38] "Beluga: Programming with Dependent Types, Contextual Data, and Contexts" (Plenary Invited Talk). 10th International Symposium in Functional and Logic Programming (FLOPS'10), Sendai, Japan, 2010

[L37] "Beluga:Programming with contextual objects, contexts, and ...". International Workshop on Programming Languages for Mechanized Mathematics Systems, Paris, France, July 2010.

[L36] "Beluga:Programming with dependent types and higher-order data". Workshop on Interaction versus Automation: The two faces of deduction, Schloß Dagstuhl, Germany, Oct 2009.

[L35] "Beluga:Programming with dependent types and higher-order data". Workshop on Interaction vs Automation, Schloß Dagstuhl, Germany, Oct 2009.

[L34] "Contextual modal type theory: a foundation for access control". Workshop on Mobility, Ubiquity and Security, Schloß Dagstuhl, Germany, Feb 2007.

[L33] "Contextual modal type theory: a foundation for meta-variables". Workshop on Automated Deduction, Schloß Dagstuhl, Germany, October 2005.

[L32] "Tabling in higher-order logic programming". Mobile Code Safety and Program Verification Using Computational Logic Tools, Workshop at the International Conference of Logic Programming (ICLP), Barcelona, Spain, 2005.

[L31] "Contextual modal type theory: a foundation for meta-variables". (joint work with F. Pfenning and A. Nanevski). Workshop on Implementations of Proof Search (WIPS'05), University of Minnesota, Minneapolis, USA, 2005.

Invited Lectures at Summer Schools

Invited Lecturer on "Proof-theoretic Foundations" at the Oregon Programming Languages Summer School, Eugene, USA, June 2014 (I gave 4 lectures each 90min)

Invited Lectures at Universities

[L33] *"Beluga: Programmme und Beweise im Kontext"*. Wolfgang Göthe Universität Frankfurt, Colloquium, July 2015.

[L33] *"Beluga$^{\mu}$: Programming logical relations proofs"*. POPL PC Workshop, Princeton University, Princeton, USA, Sept 2014.

[L32] *"Beluga$^{\mu}$: Programming proofs in context"*. Programming Languages Seminar, Carnegie Mellon University, Pittsburgh, USA, May 2014.

[L31] *"Structural Recursion over Contextual Objects"*. INRIA Sophia-Antipolis, Antibes, France, April 2014.

[L30] *"Beluga$^{\mu}$: Programming proofs in context"*. Programming Systems Seminar, Max-Planck Institute for Software Systems, Saarbrücken, Germany, Nov 2012.

[L29] *"Beluga$^{\mu}$: Programming proofs in context"*. Seminar on Games and Decisions for Rigorous Systems Engineering, Schloß Dagstuhl, Saarbrücken, Germany, Nov 2012.

[L28] *"Programming with recursive types and binders"*. Seminaire Thèorie des types et rèalisabilitè, Inria - Place d'Italie, Paris, France, Feb 2012.

[L27] *"Programming with recursive types and binders"*. École Polytechnique, LIX, Paris, France, May 2011.

[L26] *"Beluga:Programming with contextual objects, contexts, and ..."*. Inria Saclay, Paris, France, April 2010.

[L25] *"Beluga:Programming with contextual objects, contexts, and ..."*. Seminaire Preuves, Programmes et Systémes, Universitè Paris Diderot, Paris 7, France, Feb 2010.

[L24] *"Beluga:Programming with dependent types and higher-order data"*. Seminaire de Logique, UQUAM, Montreal, Quebec, Dec 2009.

[L23] *"Contextual modal logic and its applications"*. Invited Mini-Series, IT University of Copenhagen, Denmark 2009.

TALKS (CONT.)

Invited Lectures at Universities (cont.)

[L22] *"Beluga:Programming with dependent types and higher-order data"*. Parsifal Seminar, LIX, Ecole Polytechnique, Paris, France, Oct 2009.

[L21] *"Beluga:Programming with dependent types and higher-order data"*. Programming Systems Seminar, University of Saarbrücken / Max-Planck Institute for Software Systems, Saarbrücken, Germany, Oct 2009.

[L20] *"Beluga:Programming with dependent types and higher-order data"*. Gallium Seminar, Inria Rocquencourt, Paris, France, Oct 2009.

[L19] *Programming with higher-order data, proofs and explicit contexts* Parsifal Seminar, Ecole Polytechnique/INRIA Futurs, Paris, France, May 2008.

[L18] *Programming with higher-order abstract syntax and environments* Programming Languages Seminar, University of Chicago/Toyota Technology Institute, Chicago, USA, Nov 2007.

[L17] *Programming with higher-order abstract syntax and environments* Parsifal Seminar, Ecole Polytechnique/INRIA Futurs, Paris, France, June 2007.

[L16] *Programming with higher-order abstract syntax and environments* Programming Languages Seminar, Carnegie Mellon University, Pittsburgh, USA, May 2007.

[L15] *Contextual modal type theory: a foundation for meta-variables*, (joint work with F. Pfenning and A. Nanevski) Logic Seminar, University of Saarbrücken, Saarbrücken, Germany, April 2006.

[L14] *Overcoming performance barriers: efficient proof search in logical frameworks* Computer Science Colloquium, Technical University Darmstadt, Darmstadt, Germanym Dec 2005.

[L13] *Overcoming performance barriers: efficient proof search in logical frameworks* Parsifal Seminar, Ecole Polytechnique/INRIA Futurs, Paris, France, May 2005.

[L12] *Overcoming performance barriers: efficient proof search in logical frameworks* Computer Science Colloquium, University of Vermont, Burlington, USA, March 2005.

[L11] *Overcoming performance barriers: efficient proof search in logical frameworks* Computer Science Colloquium, Clarkson University, Pottsdam, USA, January 2005.

[L10] *Overcoming performance barriers: efficient proof search in logical frameworks* Computer Science Colloquium, McMaster University, Hamilton, Canada, March 2003.

[L9] *Overcoming performance barriers: efficient proof search in logical frameworks* Computer Science Colloquium, University of Toronto, Toronto, USA, March 2003.

[L8] *Overcoming performance barriers: efficient proof search in logical frameworks* Computer Science Colloquium, University of Indiana, Bloomington, USA, March 2003.

TALKS (CONT.)

Invited Lectures at Universities (cont.)

[L7] *Overcoming performance barriers: efficient proof search in logical frameworks* Computer Science Colloquium, Johns Hopkins University, Baltimore, USA, April 2003.

[L6] *Overcoming performance barriers: efficient proof search in logical frameworks* Computer Science Colloquium, Toyota Technology Institute and University of Chicago, Chicago, USA, April 2003.

[L5] *Overcoming performance barriers: efficient proof search in logical frameworks* Computer Science Colloquium, McGill University, Montreal, Canada, April 2003.

[L4] *Termination and reduction in higher-order logic programming* Seminar on Deduction, Schloß Dagstuhl, Germany, March 2001.

[L3] *Termination and reduction in the logical framework (including a general introduction to the logical framework)* German Center for Artificial Intelligence, Saarbrücken, Germany, Dec 2000.

Invited Presentations (Gender Studies/Outreach)

[L5] *Commemorating Women in Technology Pioneers Anita Borg and Grace Hopper* Celebrating Women in Computing, IEEE Women in Engineering McGill University, Montreal, March 2015

[L4] *Computational Thinking: What is the Science in Computer Science?* Annual Cognitive Science Research Day, McGill University, Montreal, October 2014

[L3] *Computational Thinking: What is the Science in Computer Science?* Women in Science Day, Dawson College, Montreal, March 2013

[L2] Participant on the panel *"Women in Computer Science: The Carnegie Mellon Experience"* Grace Hopper Celebration of Women in Computing, Cape Cod, USA, September 2000.

[L1] *How to prove it* International Conference on Women, Work and Computerization, Bonn, Germany, May 1997.

PROFESSIONAL SERVICE

Grant review

NSF Grant review panel 2013.

External NSERC Grant reviewer 2013.

Dutch Granting agency reviewer 2011.

Steering Committee

Trustee on the board of AAR (Association of Automated Reasoning), 2012 - 2015.

Elected trustee to the board of CADE (Conference on Automated Deduction), 2011. reelected in 2013

Steering committee member for the International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice (LFMTP) (chair of the steering committee since 2015)

Editor

Editorial Board of the Journal of Functional Programming (Feb 2014 - Dec 2018)

Editorial Board of the IfCoLog Journal of Logics and their Applications (started in 2015)

Special issue of the Journal of Automated Reasoning with best papers from CADE-22 (2009)

Special issue in the journal "Information and Computation" on Intuitionistic Modal Logics and Applications (IMLA) (2009)

Association of Logic Programming Newsletter (May 2005 - May 2009)

Program Chair

International Conference on Formal Structures for Computation and Deduction (FSCD'16), Porto, Portugal, 2016.

Conference Chair

22nd International Conference on Automated Deduction (CADE), Montreal, Canada, 2009

Conference Program Committee Member

43rd ACM Symposium on Principles of Programming Languages (POPL 2016), Philadelphia, USA, Extended Review Committee, 2015

25th International Conference on Automated Deduction (CADE-25), Aug 2015, Berlin, Germany.

42nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'15), Jan 2015, Mumbai, India.

PROFESSIONAL SERVICE (CONT.)

Conference Program Committee Member (cont.)

Joint 25th International Conference on Rewriting Techniques and Applications and 12th International Conference on Typed Lambda Calculi and Applications, July, 2014, Vienna, Austria

24th International Conference on Automated Deduction (CADE-24), Lake Placid, USA, June 2013

International Conference on Interactive Theorem Proving (ITP'13), Rennes, France, Aug 2013

17th ACM SIGPLAN International Conference on Functional Programming, Copenhagen, Denmark, 2012

First International Conference on Certified Programs and Proofs (CPP 2011), co-located with APLAS'11, Taiwan, Dec 2011

2nd International Conference on Interactive Theorem Proving (ITP), Princeton, USA, 2012

Turing Centenary Conference: CiE 2012 - How the World Computes, University of Cambridge, Cambridge, UK, 2012

39th ACM Symposium on Principles of Programming Languages (POPL 2012), Philadelphia, USA, Extended Review Committee, 2012

1st International Conference on Interactive Theorem Proving (ITP), Nijmegen, Netherlands, 2011

23nd International Conference on Automated Deduction (CADE), Wroclaw, Poland, 2011

5th International Joint Conference on Automated Reasoning (IJCAR), Edinburgh, Scotland, 2010

1st International Conference on Interactive Theorem Proving (ITP), Edinburgh, Scotland, 2010

26th International Conference on Mathematical Foundations of Programming Semantics (MFPS-10), Ottawa, Canada, 2010

19th International Workshop on Functional and (Constraint) Logic Programming (WFLP'10), Madrid, Spain, 2010.

22nd International Conference on Automated Deduction (CADE), Montreal, Canada, 2009

13th ACM SIGPLAN International Conference on Functional Programming (ICFP'08), Victoria, British Columbia, Canada, 2008

24th International Conference on Logic Programming (ICLP'08), Udine, Italy, 2008

PROFESSIONAL SERVICE (CONT.)

14th International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR'07), Yerevan, Armenia, 2007

22nd International Conference on Logic Programming (ICLP'06), Seattle, USA, 2006

21st Conference on the Mathematical Foundations of Programming Semantics, (MFPS'05), Birmingham, UK, 2005

6th ACM International Conference on Principles and Practice of Declarative, (PPDP'04), Verona, Italy, 2004

Conference Organizing Committee Member

20th International Conference of Automated Deduction, (CADE'05), Talinn, Estonia, 2005
(Publicity Chair)

Workshop Chair/Organizer

International Workshop on Logical Frameworks and Meta-Languages: theory and practice (LFMTP'14), June 2014, Vienna, Austria, co-located with LICS/CSL'14 and IJCAR'14.
(Workshop Organizer and Workshop Program Chair)

International Workshop on Logical Frameworks and Meta-Languages: theory and practice (LFMTP'13), Sept 2013, Boston, USA, co-located with ICFP'13.
(Workshop Organizer and Workshop Program Chair)

International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice (LFMTP), Bremen, Germany, 2007,
affiliated with CADE'07
(Workshop organizer)

International Workshop on Logical Frameworks and Meta-Languages: Theory and Practice (LFMTP), Seattle, USA, 2006,
affiliated with LICS'06 and IJCAR'06, as part of FLOC'06
(Workshop Program Chair)

Quebec Programming Languages Seminaire (QCPLS), Montreal, Quebec, Nov 2004
(Workshop organizer)

Quebec Programming Languages Seminaire (QCPLS), Montreal, Quebec, April 2004
(Workshop organizer)

Workshop Program Committee Member

ACM SIGPLAN Workshop on Dependently Typed Programming (DTP'13),
co-located with ICFP'13.

2nd ACM SIGPLAN Workshop on Higher-Order Programming with Effects (HOPE'13),
co-located with ICFP'13.

Intuitionistic Modal Logics and Applications Workshop (IMLA'08),
affiliated with LICS'08

PROFESSIONAL SERVICE (CONT.)

Workshop Program Committee Member

Workshop on Practical Aspects of Automated Reasoning (PAAR'08), affiliated with IJCAR'08

International Workshop on Logical Frameworks and Meta-Languages (LFMTP'08), affiliated with LICS'08

Programming Languages meets Program Verification (PLPV'07), affiliated with ICFP'07

ACM Workshop on Programming Languages and Analysis for Security (PLAS'06), affiliated with PLDI'06

Journal Reviewer

Journal of Higher-Order Symbolic Computation (HOSC), Theory and Practice of Logic Programming (TPLP), Logical Methods in Computer Science, Journal of Functional Programming

Conference reviewer International Conference on Logic in Computer Science (LICS), ACM SIGPLAN Symposium on Principles and Practice of Declarative Programming (PPDP), ASIAN Symposium on Programming Languages and Systems (APLAS), International Joint Conference on Automated Reasoning (IJCAR), Annual Conference on Computer Science Logic (CSL), International Conference on Rewriting Techniques and Applications (RTA), International Conference on Foundations of Software Science and Computation Structures (FOSSACS), International Conference on Logic for Programming Artificial Intelligence and Reasoning (LPAR), International Conference on Typed Lambda Calculi and Applications (TLCA)

SUPERVISION OF STUDENTS

Post-doctoral student

Joshua Dunfield (PhD Carnegie Mellon University) (September 2007 – August 2010)
"Programming and reasoning with higher-order data"

Mathieu Boespflug (PhD École Polytechnique, France) (January 2011 - Dec 2012)
Beluga modulo

Levi Lucio (PhD Switzerland) (August 2011 - August 2014, jointly supervised with Prof. Hans Vangheluwe)
Modelling and reasoning about automotive software

Tao Xue (PhD University of London, UK) (June 2012 - August 2013)
Proof-carrying authorization in Beluga

Matthias Püch (PhD Bologna, Italy) (April 2014 - April 2014)
Logical foundation for open code generation

Doctoral students

Ian Clement (NSERC scholarship, Winter 2010 - Fall 2012)
Reasoning inductively with contextual objects

Andrew Cave (NSERC scholarship, started in Winter 2012)
Programming and reasoning with proofs

Francisco Ferreira (FQRNT scholarship, started in Winter 2012)
Compiling proofs and programs

David Thibodeau (NSERC Scholarship, Sept 2015 - present)
Programming coinductive proofs

Master students

Ye Henry Tian (completed August 2005)
"Mechanically verifying correctness of CPS compilation"

Ahmer Ahmedani (completed October 2006, jointly supervised with Prof. Verbrugge)
" Information flow in a Java intermediate language"

Jacques le Normand (completed Feb 2007, jointly supervised with Prof. Panangaden)
" Guarded Abstract Data-types"

Xi Li (David) (completed Nov 2007)
"Inverse method for logical frameworks"

Samuli Heilala (completed in Feb 2008)
"A path characterization of validity for multimodal logics"

Ian Clement (completed in August 2008)
"Constructive description logics"

Samuel Gelineau (FQRNT scholarship, completed in April 2010, jointly supervised with Prof. Jörg Kienzle)
"Types for Aspects"

SUPERVISION OF STUDENTS (CONT.)

Master students

Renaud Germain (completed April 2010)
"Implementation of a dependently typed programming language"

Andrew Cave (NSERC scholarship, started Sep 2010, fast-tracked to PhD in 2011)
Programming with binders and recursive types

Francisco Fereira (Sep 2009 - Jan 2012) (student at Unversite de Montreal, jointly supervised with Prof. S. Monnier)
Compiling Beluga

Olivier Savary Bélanger (May 2012 - Aug 2014) (FQRNT Scholarship, jointly supervised with Prof. S. Monnier)
Implementing type safe program transformations in Beluga

David Thibodeau (NSERC and FQRNT Scholarship, Jan 2013 -present)
Programming coinductive proofs

Rohan Jacob-Rao (Sept 2014 - present)
Developing proofs about formal systems semi-automatically

Shawn Otis (Sept 2014 - present) (Arbour Foundation Scholarship, FQRNT Scholarship)
Explicit substitution calculi supporting contexts and contextual types

Stefan Knudsen (Sept 2015 - present)
Dependently Typed Reactive Programming

Undergraduate students

Xi Li (David) (May 2005 – August 2005)
"Decision procedures for propositional logic and congruence closure"

Benjamin Azan (May 2005 – September2005)
"Type systems for Featherweight Java: theory and implementation"

Samuli Heilala (Honors Theses) (January 2006 – August 2006)
"Decision procedures for intuitionistic modal logic"

Dustin Wehr (Honors Theses) (January 2007 – April 2007)
"Refinement types for logical frameworks"

Dustin Wehr (Science Undergraduate Research Award) (May 2007 – August 2007)
"Practical programming with higher-order abstract syntax"

Maja Frydrychowicz (CDMP and NSERC USRA Award) (May 2007 – August 2007)
"A logical foundation for enforcing access control"

Andres Franceschi (Science Undergraduate Research Award) (May 2007 – August 2007)
"A practical comparison of LambdaProlog and Twelf"

SUPERVISION OF STUDENTS (CONT.)

Undergraduate students

Maja Frydrychowicz (NSERC USRA Award) (May 2008 – August 2008)
"Termination checking when programming with proofs"

Abbie Desrosier (Honors Theses) (May 2009 – August 2009)
"Compiler construction with higher-order abstract syntax"

Ali Assaf (Science Undergraduate Research Award) (May 2009 – August 2009)
"Designing an interpreter for a higher-order dependently typed language"

Shen Chen Xu, (Honors thesis) (Jan 2011 - April 2011)
Coverage checking for dependently typed programs

Rafik Draoui (Undergraduate Research Assistant, May 2011 - August 2011)
Exploring Sized types

David Thibodau (SURA 2011, May 2011- August 2011)
Implementing pure type systems in Beluga

Olivier Savary Bélanger (NSERC USRA 2011, Undergraduate Research Assistant January 2011 - April 2012)
Implementing compiler transformations in Beluga

Marie-Andrée B.Langlois (Undergraduate Research Assistant May 2011 - April 2012)
SasyBel: Interactive proof tutor for Beluga

Costin Badescu (NSERC USRA, May 2011 - August 2011)
Implementing a higher-order logic programming interpreter

Costin Badescu (Undergraduate Research Assistnat, Jan 2012 - April 2012)
Developing and implementing Beluga1.0

Costin Badescu (Undergraduate Research Assistant, May 2012 - Aug 2012)
Structural termination checking for contextual objects

Esma Balkir (Honors student poject, Sep 2012 - Dec 2012)
Coinductive reasoning

Gabriel Gaudreault (Honors student poject, May 2012 - Aug 2012)
Linear contextual logic

David Thibodeau (NSERC USRA, May 2012 - August 2012)
Copatters: programming coinductivly

Adam Wilks (Undergraduate Research Assistant, May 2012 - Aug 2012)
Contextual types for staged computation

Shawn Otis (Undergraduate Research Assistant, May 2013 - Aug 2013)
Explicit substitutions and contextual type theory

Shanshan Ruan (SURA Award, May 2013 - Aug 2013)
Well-founded recursion over contextual objects and contexts

Steven Thephsourinthone (May 2013 - Aug 2013)
Producing human-readable proofs

SUPERVISION OF STUDENTS (CONT.)

Exchange student

Sabrina Chantrelle (May 2005 – September 2005)
"A focusing prover for bunched implications"

Florent Pompigne (March 2007 – August 2007)
"Proof-theoretic foundation for magic sets"

Dimitri Kirchner (May 2009 – August 2009)
"Building a testing framework for Beluga"

Agatha Murawska (Febuary 2015 – August 2015)
"Linear Contextual Type Theory"

Nicolas Jeannerod (April 2015 - August 2015)
"Mechanizing Proofs in Beluga"

External thesis reviewer or committee member

Choon Kyu Kim (McGill University)
PhD Thesis: Parallel Semantic Tree Theorem Proving with Resolutions (2004)

Sam Bakhtiar Sanjabi (McGill University)
Master Thesis: Dataflow analysis of the pi-calculus (2004)

Caitlin Phillips (McGill University)
Master Thesis: An Algebraic Approach to Dynamic Epistemic Logic (2009)

Layouni Mohamed (McGill University)
PhD Thesis: Privacy-Preserving Personal Information Management (2009)

Ronan Saillard (École National Supérieur de Mines de Paris, France): Rapporteur
PhD Thesis: Type Checking in the $\lambda\Pi$-Calculus Modulo: Theory and Practice (2015)

Ali Assaf (École Polytechnique Paris, France): Examiner
PhD Thesis: A Framework for Defining Computational Higher-order Logics (2015)

Awards won by supervised students

David Thibodeau
NSERC Scholarship (2015 - 2018)

Shawn Otis
FQRNT Scholarship (2015 - 2016)

Shanshan Ruan
First prize in the ACM Student Research Competition in Programming Languages at ICFP'14.

SUPERVISION OF STUDENTS (CONT.)

Awards won by supervised students (cont.)

Olivier Savary Belanger
Second Place in the ACM Student Research Competition Grand Finals - Undergraduate Research for 2013.

Francisco Ferreira
First prize in the ACM Student Research Competition in Programming Languages at ICFP'12.

Olivier Savary Bélanger
Second prize in the ACM Student Research Competition in Programming Languages at ICFP'12.

David Thibodeau
NSERC Scholarship (2013) and FQRNT Scholarship (2013 - 2015)

Olivier Savary Bélanger
FQRNT Scholarship (2012 - 2014)

Francisco Ferreira
FQRNT Scholarship (2012 - 2015)

Andrew Cave
NSERC Scholarship (2009 - 2012)

Ian Clement
FQRNT Scholarship (2009 - 2012), NSERC Scholarship (2009 - 2012)

Ali Assaf
CRA Outstanding Research Award – Honorable Mention, Male Award (2009).

Samuel Gelineau
FQRNT Scholarship (2008 - 2011)

Samuli Heilala
Governor-General's Silver Medal, one of the most prestigious awards for a graduating undergraduate student (2006); Full scholarship to support his graduate studies at McGill (2006 - 2008)

Maja Frydrychowicz
2nd prize at the Computer Science Undergraduate Research Symposium (2007); CRA Outstanding Research Award – Honorable Mention, Female Award (2007); Scholarship from the School of Computer Science and the Provost to support her graduate studies (2008).

Benjamin Azan
1st Prize at the Computer Science Undergraduate Research Symposium (2005)

COURSES TAUGHT

Undergraduate Course: COMP-302 Programming Languages and Paradigms
School of Computer Science, McGill University

Advanced undergraduate level course in the area of programming languages and paradigms, developed course outline, weekly lectures (25 lectures, 1h30min each), 5 home work assignments, this course is centered around the typed functional programming in SML and theoretical foundations of programming languages.

Taught in Fall'05, Winter'06, Winter'07, Fall'07, Fall 2010, Winter 2012, Fall 2012. Fall 2014 (enrolement: 155 students), Fall 2015 (enrolement: 178 students)

Undergraduate Course: COMP-426 Automated reasoning
School of Computer Science, McGill University

Advanced undergraduate level course in the area of theorem proving and logical foundations, developed course outline, weekly lectures (25 lectures, 1h30min each), 5 home work assignments, this course is centered around a series of implementations of theorem provers.

Taught in Fall'04, Fall'05, Fall'06, Fall'07

Graduate Course: COMP-523 Language-based security
School of Computer Science, McGill University

Graduate level course in the area of type-systems, operational semantics of programming languages, logic and computation, and logical frameworks. Developed course outline, weekly lectures (25 lectures, 1h30 each), 8 home work assignments (4 theory and 4 implementation), this course is centered around a series of implementations of type-checkers and evaluators using advanced programming languages such as SML and Twelf.

Taught in Winter 2005, Winter'06, Fall 2010, Winter 2013, Winter 2015

Graduate Course: COMP-527 Logic and Computation
School of Computer Science, McGill University

Graduate level course in the area of logic and its applications to programming languages and automated reasoning. Developed course outline, weekly lectures (25 lectures, 1h30 each), 8 home work assignments.

Taught in Winter 2008, Winter 2012, Winter 2014

Graduate Course: COMP 599 Dependent Types
School of Computer Science, McGill University

Graduate level seminar course on dependent types. The course was structured as a reading and research course. Students had to read 2 research papers for each class, lead the in-class discussion and present a paper (3 times in the semester), complete a mini-research project, peer-review projects from other students in class and participate in the workshop day where each student presented their project.

Taught in Fall 2011

COURSES TAUGHT (CONT.)

Graduate Course: COMP 762B Computation and Deduction
School of Computer Science, McGill University

Advanced graduate level course in the area of logical frameworks and their applications in programming languages, developed course outline, weekly lectures(25 lectures, 1h30min, 4 home work assignments, final project). Taught in Winter 2004, Fall 2006

TEACHING EXPERIENCE

Eberly Center for Teaching Excellence, Carnegie Mellon University
Teaching Fellow May 2001 – August 2003

Computer Science Department, Carnegie Mellon University Pittsburgh, USA
Teaching Assistant September 1999 – December 1999
September 2001 – December 2001
January 2001 – May 2001

Computer Science Department, Technical University Darmstadt
Research Assistant October 1994 – September 1995

UNIVERSITY SERVICE

School of Computer Science, McGill University
Organizer of the Undergraduate Computer Science Research Symposium (UCORE) August 2014, August 2015
Member of the MSc Committee Sept 2015 – now
Member of the Undergraduate Committee Sept 2010 – now
Member of the Web Committee Sept 2010 – now
Chair of the Women's Committee Sept 2010 – now
Liaison to Computer Science Undergraduate Society August 2006 – July 2009
Liaison to Computer Science Undergraduate Society August 2006 – July 2009
Member of the Undergraduate Committee August 2006 – July 2009
Chair of the Undergraduate Committee January 2007 – July 2007
Chair of the Women@SOCS Committee May 2005 – July 2009
Co-Founder of the Computer Science Colloquium Sept. 2004 - May 2005
Co-founder of the Undergraduate Summer Research Symposium August 2004
Member of the PhD Committee August 2003 - August 2006

Rhodes Scholarship Committee, McGill University
Member of the Rhodes Scholarship Committee September 2011 – May 2014

Faculty of Education, McGill University
Member of the Hiring Committee for Math Education Position
Representative from the Faculty of Science September 2005 – May 2007

Faculty of Science, McGill University
Judge for the Faculty of Science Undergraduate Research Conference October 2014
Guest lectures for Soup & Science Sept. 2006, Sept. 2008, Sept. 2013
Guest lecture for Mathematics Undergraduate Society October 2007
Organizer of Freshman Interests Groups (FIG) September 2006 - December 2006

School of Computer Science, Carnegie Mellon University *Founding member of Women@SCS and the Women in School of Computer Science Advisory Committee*
September 1999 – July 2003

Technical University Darmstadt
Elected Member of the Election Commission October 1996 – October 1997
Elected Equal Opportunities Officer October 1996 – October 1997

Research

My principal research interest lies in developing a theoretical and practical foundation for building and reasoning about reliable software. A central aspect of achieving this goal is the language in which the software is written. Well-designed languages support fast program development, ease software maintenance, allow effective reasoning about the runtime behaviour of programs, and increase confidence in the correctness of implementations. Hence I believe understanding the principles of programming languages and logics to write and reason about programs is key to develop safe and reliable code.

My research focuses on type systems, a standard static analysis to approximate the expected runtime behaviour of programs statically. More specifically, my work aims at making the type system more powerful so that arbitrary properties can be expressed, verified, and certified, making it possible to seamlessly cover the range from traditional types to more complex correctness properties.

There are clear advantages of such static techniques: they are cheap, provide precise error messages to the code developers giving them the opportunity to correct problems early on in the development, and they can easily and cheaply reexamine the code every time the code changes. This will lead to more reliable software and faster development by reducing the amount of time spent debugging a program.

My research is driven by two interconnected questions:

1. *How can we design languages (programming languages and logics) which allow us to statically track and certify a rich set of properties and allow more program errors to be caught at compile time?*
   My work often draws upon (constructive) logic to provide a type-theoretic foundation for reasoning about programs via the Curry-Howard isomorphism.
2. *How can we implement languages and tools to reason about programs effectively?*
   I believe theory cannot exist in a vacuum, but we must validate our ideas in practice. Hence, I strive to develop programming and proof environments based on solid theoretical foundations and evaluate them using case-studies from certifying compiler to building a proof-carrying authorization infrastructure.

In the following I will briefly describe some of my recent contributions in four areas: the design of Beluga, a dependently typed programming language which supports the specification of formal systems and proofs about them; contextual type theory to give an account for reasoning within a context; programming and reasoning about infinite structures to provide guarantees about distributed computing infrastructures; efficient data-structures and algorithms to implement programming languages and automated reasoning tools.

Beluga: Programming proofs

The Beluga project aims to change the way we develop and implement reliable software systems by extending a general purpose programming language with the ability to directly represent, generate, and manipulate proof certificates. The objective is to design a foundation for certifying programs based on dependent types where programmers can specify sophisticated safety policies and weave proofs into programs. Our goal is twofold: 1) to make it routine work for the programmer to specify and mechanically verify complex behavioural properties of their programs and ensure that these properties are preserved during compilation. 2) to make it common practice to communicate, exchange, and verify proofs to establish trust and guarantee reliability and safety of software systems.

Over the last decade we have come closer to narrowing the gap between programming software systems and reasoning about them. Yet, most existing approaches lack rich abstractions that allow users to describe formal systems and proofs on a high-level, factor out common and recurring issues, make it easy to use, and at the same time have a small trusted kernel.

One exception is the dependently-typed programming and proof environment Beluga [C13, C14, C16, C18] which is being developed in my research group. Beluga supports the specification of domain-specific logics and proofs in the logical framework LF, a rich dependently-typed meta-language. To manipulate and analyze LF objects Beluga provides a functional language which supports pattern matching. What makes Beluga unique is its support for contextual objects, i.e. an object in a context of assumptions [J4]. This allows us to concisely represent and manipulate proof states, i.e. a goal formula (object) together with the assumptions we are allowed to use to establish the goal. Our type-theoretical foundation provides a rich infrastructure for elegantly representing formal systems and proofs; in particular it supports modelling bound variables, renaming, substitution, first-class contexts to model assumptions and first-class simultanous substitutions. As such it is at this point the system with the richest support for modelling formal systems and proofs.

I believe it is vital to combine theoretical research with practical applications. Such synergy often provides new insights and allows the validation of theoretical results. We have used Beluga extensively on known benchmarks. It has also been used in teaching programming languages and logic at McGill. Over the past year, jointly with A. Felty (University of Ottawa) and A. Momigliano (University of Milan), I have embarked on a systematic study of comparing the strengths and weaknesses of proof assistants which provide a rich supporting infrastructure for modelling formal systems and proofs. In particular we have proposed ORBI, an Open Repository for Reasoning about Binders and surveyed four major systems, Twelf, Beluga, Abella and Hybrid [J7]. This is a first step towards understanding the commonalities and differences between these different approaches and arrive at a common foundation.

We have also tackled two challenge problems which were thought to be out of reach for systems based on LF: first, we demonstrated that Beluga is ideally suited for

implementing a type-preserving compiler for the simply typed lambda-calculus which supports key phases such as closure conversion and hoisting [C21] and second we show that Beluga elegantly supports encoding sophisticated normalization proofs using logical relations [W9].

Since its publication, the core papers on Beluga ([C13, C14, C16, C18]) have received over 250 citations. The ideas underlying Beluga have also inspired languages such as VeriML (Yale University), a tactic language for the Coq proof assistant.

In 2010 I was invited to give one of the keynote lectures about this work at FLOPS'10 [B3]; in 2012 I gave one of the invited talks at the Workshop of Logical Frameworks and Meta-Languages; In 2014, I was invited to participate in the workshop on "Certification of Low-level and High-level Programs" as part of the thematic trimester on Semantics of proofs and certified mathematics at the Institut Henri Poincaré in Paris; recently, I was asked to give one of the invited talks at "International Conference on Certified Proofs and Programs (CPP'15)" which I had to decline due to other existing commitments and gave an invited lecture at the "International Workshop on Rewriting Techniques for Program Transformations and Evaluation (WPTE'15)". I also gave a three hour tutorial about Beluga at the "International Conference on Automated Deduction (CADE'15)", the top conference in the field.

Core: Contextual Reasoning

In the Core project, we investigate and develop a logical and type-theoretic foundation for describing and manipulating open code which can be understood within a context. This not only plays an important role when describing derivation trees in logic and reasoning about the structure of derivation, it also plays a key role in modelling code generation and staged computation in programming languages and understanding proof search. More generally, contextual reasoning plays also an important role in theorem proving, knowledge representation, and linguistics. In collaboration with A. Nanevski (IMDEA) and F. Pfenning (Carnegie Mellon University) we develop a constructive type theoretic foundation for modelling contextual objects based on modal logic S4. This work was published in *ACM Transactions on Computational Logic* in 2008; since then has received over 163 citations and has played a major role in explaining proofs with holes and developing a foundation for programming with binders; it has also been used to provide a foundation for tactic languages such as VeriML (Yale University) and to reason about information flow. Jointly with my post-doc M. Boespflug I have generalized this work to a multi-level contextual type theory [W8]. Together with A. Cave (PhD student), we have extended this work to account for first-class simultanous substitutions [W9] together with a rich equational theory of substitutions and have demonstrated that this is key to elegantly encode normalization proofs using logical relations. Currently, my postdoc, M. Puech and myself, are using contextual type theory to give a foundational systematic account for staged computation.

Programming and reasoning about infinite structures

Representing and reasoning about infinite computation plays a crucial role in our quest for designing and implementing safe software systems, since we often want to model and reason about systems which react and respond to inputs such as games or web servers, and establish behavioural properties such as liveness properties. Reasoning about interactive and often distributed systems is becoming more and more important with the prominence and rise of distributed computing infrastructures and cloud computing. Yet, not no practical overarching methodologies exist to establish (partial) correctness properties.

In my research, I have tackled this problem from two angles: First, jointly with my colleague P. Panangaden and my PhD students A. Cave and F. Ferreira I have proposed a novel logical foundation for fair reactive functional programming which seeks to provide guarantees about interactive systems by explicitly modelling discrete time abstractly [C22]. Our type system provides a sound proof system for linear temporal logic (LTL) and expands on existing Curry-Howard interpretations of LTL. In particular, we demonstrate that distinguishing between and interleaving of least and greatest fixed points is key to statically guarantee liveness properties, i.e. something will eventually happen, by type checking. To illustrate the power and elegance of this idea, we describe the type of a fair scheduler – any program of this type is guaranteed to be fair, in the sense that each participant is guaranteed that his requests will eventually be served.

Second, jointly with A. Abel (Chalmers), my graduate student D. Thibodeau and A. Setzer (Swansea) we are building a general type-theoretic foundation for programming with infinite structures via observations [C19, C20, C23, J9]. Our work draws on ideas in category theory where we model finite objects via initial algebras and infinite objects via final coalgebras, but shows hot to seamlessly integrate them into existing languages like Haskell and ML. In functional languages, finite data is defined via (inductive) data types and we use pattern matching to analyze it. Dually, we propose to define infinite data via coinductive types and make observations about it via *copattern* matching. Technically, our theoretical foundation for patterns and copatterns takes inspiration from the growing body of work which relates classical and linear logic to programming language theory via the Curry-Howard-Isomorphism and avoids many of the pitfalls and shortcomings which currently exist in systems such as Coq, a proof assistant based on the Calculus of (Co)Inductive Construction, or Agda, a dependently typed programming environment based on Martin Löf type theory.

Efficient data-structures and algorithms

I believe theory cannot exist in a vacuum, but we must validate our ideas in practice. Hence, I believe implementing programming environments which allow us to explore the usefulness and effectiveness of our proposed theoretical foundations is key. Efficient data-structures and implementation techniques play a crucial role in utilizing the full

potential of a reasoning environment in real applications. Although this need has been widely recognized for simply-typed first-order languages, efficient algorithms for dependently typed higher-order languages are still a central open problem.

The implementation technology underlying existing dependently typed programming and proof environments such as Twelf, Agda, or Coq is woefully ill-understood and sophisticated, well-understood data-structures and algorithms are urgently needed. I have made several contributions in this area: For example, term indexing aims at overcoming program degradation by sharing common structure and factoring common operations. Higher-order term indexing has been a central open problem, limiting the application and the potential impact of higher-order reasoning systems. I have designed and implemented higher-order term indexing techniques. They improve performance by up to a factor of 10, illustrating the importance of indexing. This paper won *the best student paper award at the 20th International Conference on Logic Programming (ICLP'03).[J5]*

Unification lies at heart of automated reasoning systems and dependently typed proof assistants. In [C10], I have designed a strategy which allows us to eliminate redundant type information. Experiments show a substantial speed-up making the execution of some examples feasible. The presented ideas have subsequently been used by colleagues to compile dependently-typed higher-order logic programs to an efficient first-order higher-order logic programming engine, LambdaProlog. Together with A. Abel, I have recently described an extension to higher-order unification to handle a richer class of problems [C17]. This work is already in use in Beluga and has been incorporated into Agda, a popular dependently-typed programming system. Finally, developed a theoretical foundation for type-reconstruction [J6] for the logical framework LF and jointly with my PhD student F. Ferreira we have extended these ideas to describe type reconstruction for a dependently typed functional language which supports pattern matching [C24].