Daniel R. Licata

Personal Information:

E-mail:	dlicata@wesleyan.edu
Web:	http://dlicata.web.wesleyan.edu/
Address:	Wesleyan University Department of Mathematics and Computer Science 265 Church St. Middletown, CT 06459
Mobile Phone:	+1 (412) 889-0106

Academic Positions:

Wesleyan University 2013–present
Assistant Professor of Computer Science.

Institute for Advanced Study 2012–2013
Member.

Carnegie Mellon University 2011–2012
Teaching Post-doctoral Fellow.

Education:

Carnegie Mellon University 2004–2011
PhD in Computer Science. Advised by Robert Harper.

Brown University 2000–2004
Bachelor of Science in Mathematics and Computer Science.

Internships:

Microsoft Research Cambridge Summer 2007
With Simon Peyton Jones.

Awards:

FoLLI E.W. Beth Dissertation Award, 2012 Winner

Funding:

Co-PI of NSF CCF-1618203 Cost Recurrences For Higher-Order Functional Programs. With Norman Danner. $461,830 over 3 years, starting 6/1/2016.

PI of AFOSR Young Investigator Program grant, Software Verification with Directed Type Theory, 2016. $357,046 over 3 years, starting 7/15/2016.

Team Member on US AFOSR MURI FA9550-15-1-0053, Homotopy Type Theory: Unified Foundations of Mathematics and Computation. $217,703 over 3 years, starting 1/1/2015.

Cowrote NSF Grant CCF-1116703, Foundations and Applications of Higher-Dimensional Type Theory, which funded part of my post-doc.

Cowrote NSF Grant CCF-0702381, Integrating Types and Verification, which funded part of my dissertation work.

Publications:

Dissertation
Dependently Typed Programming with Domain-Specific Logics. February, 2011.
Committee: Robert Harper, Frank Pfenning, Karl Crary, Greg Morrisett.

Books

Homotopy Type Theory: Univalent Foundations of Mathematics. May, 2013.
Co-authored with participants in IAS Univalent Foundations program.
Open-source research textbook available from `homotopytypetheory.org/book/`.
50,000 downloads; 1,600 paper copies.

Conference and Workshop Papers

A Mechanization of the Blakers–Massey Connectivity Theorem in Homotopy Type Theory. Kuen-Bang Hou (Favonia), Eric Finster, Daniel R. Licata, and Peter LeFanu Lumsdaine. *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2016.

Adjoint Logic with a 2-Category of Modes. Daniel R. Licata and Michael Shulman. *Logical Foundations of Computer Science (LFCS)*, 2015.

Denotational Cost Semantics for Functional Languages with Inductive Types. Norman Danner, Daniel R. Licata, and Ramyaa. *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, 2015.

A Cubical Approach to Synthetic Homotopy Theory. Daniel R. Licata and Guillaume Brunerie. *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2015.

Homotopical Patch Theory. Carlo Angiuli, Edward Morehouse, Daniel R. Licata, and Robert Harper. *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, 2014.

Eilenberg-MacLane Spaces in Homotopy Type Theory. Daniel R. Licata and Eric Finster. *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2014.

$\pi_n(S^n)$ in Homotopy Type Theory. Daniel R. Licata and Guillaume Brunerie. *International Conference on Certified Programs and Proofs (CPP)*, 2013. Invited paper (not refereed).

Calculating the Fundamental Group of the Circle in Homotopy Type Theory. Daniel R. Licata and Michael Shulman. *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2013.

Canonicity for 2-Dimensional Type Theory. Daniel R. Licata and Robert Harper. *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL)*, 2012.

2-Dimensional Directed Type Theory. Daniel R. Licata and Robert Harper. *Mathematical Foundations of Programming Semantics (MFPS)*, 2011.

A Monadic Formalization of ML5. Daniel R. Licata and Robert Harper. *Workshop on Logical Frameworks and Meta-Languages: Theory and Practice (LFMTP)*, EPTCS 34, 2010.

Security-Typed Programming within Dependently-Typed Programming. Jamie Morgenstern and Daniel R. Licata. *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, 2010.

Positively Dependent Types. Daniel R. Licata and Robert Harper. *ACM SIGPLAN Workshop on Programming Languages Meets Program Verification (PLPV)*, January 2009.

A Universe of Binding and Computation. Daniel R. Licata and Robert Harper. *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, 2009.

Focusing on Binding and Computation. Daniel R. Licata, Noam Zeilberger, and Robert Harper. *IEEE Symposium on Logic in Computer Science (LICS)*, June 2008.

Verifying Interactive Web Programs. Daniel R. Licata and Shriram Krishnamurthi. *Automated Software Engineering*, 2004. IEEE Press.

The Feature Signatures of Evolving Programs. Daniel R. Licata, Christopher Harris, and Shriram Krishnamurthi. *Automated Software Engineering*, 2003. IEEE Press.

Journal Articles

Homotopical Patch Theory. Carlo Angiuli, Edward Morehouse, Daniel R. Licata, and Robert Harper. *Journal of Functional Programming*, forthcoming 2016.

Robert Harper and Daniel R. Licata. Mechanizing Metatheory in a Logical Framework. *Journal of Functional Programming.* 17(4-5), pp 613-673, July 2007.

Submissions

Adjoint Logic with a 2-Category of Modes (extended version). Daniel R. Licata and Michael Shulman. Submitted, 2016.

Functional Pearl: Intrinsic Verification of a Regular Expression Matcher. Joomy Korkut, Maksim Trifunovski, and Daniel R. Licata. Submitted, 2016.

Technical Reports

Tom Murphy VII, Daniel Spoonhower, Chris Casinghino, Daniel R. Licata, Karl Crary, and Robert Harper. The Cult of the Bound Variable: The 9th Annual ICFP Programming Contest. Technical Report CMU-CS-06-163, 2006.

Daniel R. Licata and Robert Harper. A Formulation of Dependent ML with Explicit Equality Proofs. Technical Report CMU-CS-05-178, 2005.

Talks:

Invited Conference and Workshop Talks

Cubical Type Theory Mini-Course. Workshop on Homotopy Type Theory and Univalent Foundations of Mathematics, The Fields Institute, 2016.

Verification and Parallelism in Introductory Computer Science. Unifying Speaker, European Joint Conferences on Theory and Practice of Software (ETAPS), 2015.

A Cubical Infinite-Dimensional Type Theory. Oxford Homotopy Type Theory Workshop, 2014.

What is Homotopy Type Theory? Coq Workshop, Vienna Summer of Logic, 2014.

Computation in Homotopy Type Theory. Workshop on Semantics of Programming Languages, Thematic trimester on Semantics of Proofs and Certified Mathematics, Institute Henri Poincaré, 2014.

Eilenberg-MacLane Spaces in Homotopy Type Theory. Workshop on Formalization of Mathematics in Proof Assistants, Thematic trimester on Semantics of Proofs and Certified Mathematics, Institute Henri Poincaré, 2014.

Homotopy Type Theory and Univalent Foundations. Two lectures at Mathematical Structures of Computation, Lyon. 2014.

Programming and Proving with Higher Inductive Types. International Conference on Certified Programs and Proofs (CPP), 2013.

Computer-Checked Proofs in the Logic of Homotopy Theory. Association for Symbolic Logic North American Meeting, 2013.

Contributed Conference and Workshop Talks
Adjoint Logic with a 2-category of Modes. LFCS, 2016.

Verification and Parallelism in Intro CS. Daniel R. Licata. *POPL PC Meeting Workshop*, 2014.

Eilenberg-MacLane Spaces in Homotopy Type Theory. LICS, 2014.

Calculating the Fundamental Group of the Circle in Homotopy Type Theory. LICS, 2013.

Computing with Univalence. Daniel R. Licata and Robert Harper. *Workshop on Higher-Dimensional Algebras, Categories, and Types*, 2012.

Canonicity for 2-Dimensional Type Theory. POPL, 2012.

2-Dimensional Directed Type Theory. MFPS, 2011.

Security-Typed Programming within Dependently-Typed Programming. *Dependently Typed Programming*, 2010 and ICFP, 2010.

A Monadic Formalization of ML5. LFMTP, 2010.

Positively Dependent Types. PLPV, 2009.

A Universe of Binding and Computation. ICFP, 2009.

Focusing on Binding and Computation. LICS, 2008.

Mechanizing a Decision Procedure for Coproduct Equality. Arbob Ahmad, Daniel R. Licata, and Robert Harper. *ACM Workshop on Mechanizing Metatheory*, 2007.

Talks for Professional Groups
A 2-Categorical Framework for Substructural and Modal Logics. *Meeting of the IFIP Working Group 2.8, Functional Programming*, 2016.

Cubical Type Theory. *Meeting of the IFIP Working Group 2.8, Functional Programming*, 2014.

Git as a HIT. *Meeting of the IFIP Working Group 2.8, Functional Programming*, 2013.

Programming in Homotopy Type Theory. *Meeting of the IFIP Working Group 2.8, Functional Programming*, 2012.

Lectures at Summer/Winter Schools

Programming Languages Foundations. Four lectures at Oregon Programming Languages Summer School. 2016.

Dependently Typed Programming in Agda. Five lectures at Oregon Programming Languages Summer School. 2013.

The Twelf Tutorial, co-located with POPL 2009.

Departmental Colloquia/Seminars
Computer-Checked Programs and Proofs. Williams College, 2016.

Adjoint Logic with a 2-category of Modes. Carnegie Mellon University, 2016.

Cubical Type Theory. Carnegie Mellon University, 2015.

Functional Programs that Prove Theorems About Spaces, Brown University. 2014.

Verification and Parallelism in Intro CS. Microsoft Research Redmond, 2014.

Lectures on Homotopy Type Theory. Microsoft Research Redmond, 2014.

Mathematical and Computational Applications of Homotopy Type Theory. University of Iowa, 2013.

Programming and Proving in Homotopy Type Theory. Wesleyan, Princeton, and Cornell, 2013.

Wesleyan Talks
Computer-checked Programs and Proofs. Wesleyan Mathematics and Science Scholars lab tour, 2015.

Structural Proof Theory of Adjoint Functors. Wesleyan University Topology Seminar, 2015.

Verification and Parallelism in Intro CS. Wesleyan University Natural Sciences and Mathematics Lunch, 2015.

Computer-Checked Programs and Proofs. Wesleyan University Natural Sciences and Mathematics Lunch, 2014.

Eilenberg-MacLane Spaces in Homotopy Type Theory. Wesleyan University Topology Seminar, 2014.

Homotopy Theory in Type Theory. Wesleyan University Topology Seminar, 2013.

Professional Activities:

Professional Groups
Member, IFIP Working Group 2.8, Functional Programming.

Journal Editorial Boards
Mathematical Structures in Computer Science, 2016 – present.

Program Committees
Formal Structures for Computation and Deduction (FSCD), 2016.
ACM International Conference on Functional Programmming (ICFP), 2016 (external

review committee).
ACM Workshop on Logical Frameworks and Metalanguages: Theory and Practice (LFMTP), 2016 (co-chair).
Certified Programs and Proofs (CPP), 2016.
Homotopy Type Theory/Univalent Foundations Workshop (HoTT/UF), 2015.
New England Programming Languages and Systems Symposium (NEPLS), 2015
Foundations of Software Science and Computation Structures (FoSSaCS), 2015.
ACM Symposium on Principles of Programming Languages (POPL), 2015.
Trends in Functional Programming in Education (TFPIE), 2014.
Principles and Practice of Declarative Programming (PPDP), 2014.
ACM Workshop on Logical Frameworks and Metalanguages: Theory and Practice (LFMTP), 2012 and 2013.
ACM Workshop on Programming Languages meets Program Verification, 2012.
ACM Workshop on Types in Language Design and Implementation (TLDI), 2012.
ACM Workshop on Mechanizing Metatheory (WMM), 2009.

Funding Panels
NSF, 2014.

External Reviewer
External reviewer for journals (JFP, HOSC, TOSEM, TCS, MSCS), conferences and workshops (POPL, ICFP, LICS, TLCA, RTA, PLPV, LFMTP, AOSD, MFPS, ESOP, CPP, CSL, APLAS, ITP, ICALP), and grants (Canada NSERC).

Event Organization
Oregon Programming Languages Summer School (OPLSS), 2016. 2-week school.

New England Programming Languages and Systems Symposium. 1-day workshop at Wesleyan, 2015.

The Twelf Tutorial, co-located with POPL 2009. 1-day workshop teaching participants to use the Twelf proof assistant.

The 9^{th} Annual ICFP Programming Contest was a three-day competition associated with the International Conference on Functional Programming. The 2006 contest asked participants to uncover the secrets of a (fictional) ancient society of computer scientists by solving a series of puzzles based on programming languages research. 700 participants on 365 teams from all over the world competed.

Teaching:

Wesleyan COMP 211: Principles of Imperative Computation
Spring, 2014 and Fall, 2015
This is the first computer science course for potential CS majors. My version of the course is an adaptation of the Principles of Imperative Computation course that was designed as part of Carnegie Mellon's curriculum revisions in 2011. A major novelty is integrating program specification and verification using contracts throughout the course.

Wesleyan COMP 212: Functional Programming
Spring and Fall, 2015 and Spring 2016
This is our second CS course for potential majors. The course integrates functional programming, parallel algorithm design and analysis, and inductive proofs of mathematical specifications of program behavior. The materials are an adaptation of CMU 15-150.

Wesleyan COMP 321: Principles of Programming Languages
Spring, 2014
This is a required core course on the theory and practice of programming languages. In my version, students learn to organize programming concepts around types; to define the syntax, static, and dynamic semantics of programming languages using inductive definitions; to prove theorems about programming languages, such as relating static and dynamic semantics by type safety; and to implement type systems and operational semantics. Students apply these skills to study programming language features such as functions, data types, dynamic typing, polymorphism, abstract types, objects, control effects, and state.

Wesleyan COMP 360: Computer-Checked Programs and Proofs
Fall, 2014 and Spring, 2016
This course is an elective on my research area. In this course, students learn to use a proof assistant to write computer-checked programs and proofs, which improves their ability to reason about their code. Students also learn about some advanced functional programming techniques that are possible in a dependently typed language.

Carnegie Mellon 15-150: Principles of Functional Programming
Course Designer and Instructor, Spring 2011 - Spring 2012
This course covers introductory course on functional programming, with an emphasis on parallelism and program verification.

Wesleyan service:

University Service
Graduate Council, 2014–2016.
Graduate Council BA/MA Admissions Subcommittee, 2015–2016.

Department Service
Math/CS Graduate Education Committee, 2014–2016.
Library committee, 2013–2014.

Research Advisees:

Wesleyan University

Ayelet Zaidenberg — PhD, 2014–present
Working on formalization of mathematics using proof assistants.

Ben Hudson — Undergraduate, Masters, 2014–present
BA honors thesis *Certified Cost Bounds in Agda: A Step Towards Automated Complexity Analysis*, 2015. MA thesis, *Computer-Checked Recurrence Extraction For Functional Programs*, 2016.

Joomy Korkut — Undergraduate, 2015–present
Completed project on formalization of regular expression matching in Agda.

Max Trifunovski — Undergraduate, 2015–present
Completed project on formalization of regular expression matching in Agda.

Emily Black — Undergraduate, 2015–present
Working on formalizing red-black tree deletion in Agda.

Carnegie Mellon University

Joseph Lee — Undergraduate, 2012

Worked on a proof that the torus is homotopy-equivalent to the product of two circles in homotopy type theory.

Jamie Morgenstern — Undergraduate, 2009-2010
Embedded a security-typed programming language in a proof assistant as a summer REU from the University of Chicago. Proceeded to the CS PhD program at CMU.

Arbob Ahmad — Undergraduate, 2007-2008
Worked on deciding coproduct equality in the λ-calculus using a logical technique called focusing. Proceeded to the CS PhD program at CMU.

PhD Thesis Committees:

Kuen-Bang Hou (Favonia) — expected 2016
Higher-Dimensional Types in the Mechanization of Mathematics, Carnegie Mellon University.