

## Daniel R. Licata

---

<b>Personal Information:</b>	E-mail: <a href="mailto:dlicata@wesleyan.edu">dlicata@wesleyan.edu</a> Web: <a href="http://dlicata.web.wesleyan.edu/">http://dlicata.web.wesleyan.edu/</a> Address: Wesleyan University Department of Mathematics and Computer Science 265 Church St. Middletown, CT 06459 Mobile Phone: +1 (412) 889-0106	
<b>Academic Positions:</b>	<b>Wesleyan University</b> Assistant Professor of Computer Science.	2013–present
	<b>Institute for Advanced Study</b> Member.	2012–2013
	<b>Carnegie Mellon University</b> Teaching Post-doctoral Fellow.	2011–2012
<b>Education:</b>	<b>Carnegie Mellon University</b> PhD in Computer Science. Advised by Robert Harper.	2004–2011
	<b>Brown University</b> Bachelor of Science in Mathematics and Computer Science.	2000–2004
<b>Internships:</b>	<b>Microsoft Research Cambridge</b> With Simon Peyton Jones.	Summer 2007
<b>Awards:</b>	FoLLI E.W. Beth Dissertation Award, 2012 Winner	
<b>Funding:</b>	Team Member on US AFOSR MURI FA9550-15-1-0053, Homotopy Type Theory: Unified Foundations of Mathematics and Computation (renewal). \$170,367 over 2 years, starting 1/1/2018.  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. 218 pages.

**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/](http://homotopytypetheory.org/book/).  
50,000 downloads; 1,600 paper copies. 484 pages.

**Conference and Workshop Papers**

Call-by-name Gradual Type Theory. Max S. New and Daniel R. Licata. *Formal Structures in Computation and Deduction (FSCD)*, 2018. 20 pages.

Internal Universes in Models of Homotopy Type Theory. Daniel R. Licata, Ian Orton, Andrew M. Pitts, and Bas Spitters. *Formal Structures in Computation and Deduction (FSCD)*, 2018. 20 pages.

Differential Cohesive Type Theory (abstract). Jacob A. Gross, Daniel R. Licata, Max S. New, Jennifer Paykin, Mitchell Riley, Michael Shulman, and Felix Wellen. *Workshop on Homotopy Type Theory and Univalent Foundations*, 2017. 2 pages.

A Fibrational Framework for Substructural and Modal Logics. Daniel R. Licata, Michael Shulman, and Mitchell Riley. *Formal Structures in Computation and Deduction (FSCD)*, 2017. 20 pages. Extended version 66 pages.

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. 10 pages.

Adjoint Logic with a 2-Category of Modes. Daniel R. Licata and Michael Shulman. *Logical Foundations of Computer Science (LFCS)*, 2015. 16 pages.

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. 12 pages.

A Cubical Approach to Synthetic Homotopy Theory. Daniel R. Licata and Guillaume Brunerie. *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2015. 13 pages.

Homotopical Patch Theory. Carlo Angiuli, Edward Morehouse, Daniel R. Licata, and Robert Harper. *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, 2014. 14 pages.

Eilenberg-MacLane Spaces in Homotopy Type Theory. Daniel R. Licata and Eric Finster. *ACM/IEEE Symposium on Logic in Computer Science (LICS)*, 2014. 9 pages.

$\pi_n(S^n)$  in Homotopy Type Theory. Daniel R. Licata and Guillaume Brunerie. *International Conference on Certified Programs and Proofs (CPP)*, 2013. Invited paper (not refereed). 16 pages.

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. 10 pages.

Canonicity for 2-Dimensional Type Theory. Daniel R. Licata and Robert Harper. *ACM SIGPLAN–SIGACT Symposium on Principles of Programming Languages (POPL)*, 2012. 12 pages.

2-Dimensional Directed Type Theory. Daniel R. Licata and Robert Harper. *Mathematical Foundations of Programming Semantics (MFPS)*, 2011. 25 pages.

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. 25 pages.

Security-Typed Programming within Dependently-Typed Programming. Jamie Morgenstern and Daniel R. Licata. *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, 2010. 12 pages.

Positively Dependent Types. Daniel R. Licata and Robert Harper. *ACM SIGPLAN Workshop on Programming Languages Meets Program Verification (PLPV)*, January 2009. 12 pages.

A Universe of Binding and Computation. Daniel R. Licata and Robert Harper. *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, 2009. 12 pages.

Focusing on Binding and Computation. Daniel R. Licata, Noam Zeilberger, and Robert Harper. *IEEE Symposium on Logic in Computer Science (LICS)*, June 2008. 12 pages.

Verifying Interactive Web Programs. Daniel R. Licata and Shriram Krishnamurthi. *Automated Software Engineering*, 2004. IEEE Press. 10 pages.

The Feature Signatures of Evolving Programs. Daniel R. Licata, Christopher Harris, and Shriram Krishnamurthi. *Automated Software Engineering*, 2003. IEEE Press. 5 pages.

### **Journal Articles**

Homotopical Patch Theory. Carlo Angiuli, Edward Morehouse, Daniel R. Licata, and Robert Harper. *Journal of Functional Programming*, 26, 2016. 42 pages.

Robert Harper and Daniel R. Licata. Mechanizing Metatheory in a Logical Framework. *Journal of Functional Programming*. 17(4-5), pp 613-673, July 2007. 62 pages.

### **Submissions/Preprints**

Cartesian Cubical Type Theory. Carlo Angiuli, Guillaume Brunerie, Thierry Coquand, Kuen-Bang Hou (Favonia), Robert Harper, and Daniel R. Licata. Preprint, 2017. 43 pages.

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

ming 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**

Small Proofs. Workshop on Computer-aided Mathematical Proof, Big Proof Program, Isaac Newton Institute for Mathematical Sciences, 2017.

A Functional Programmer's Guide to Homotopy Type Theory. Keynote, International Conference on Functional Programming, 2016.

A Taste of Dependent Types. Programming Languages Mentoring Workshop, International Conference on Functional Programming (ICFP), 2016.

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**

A Fibrational Framework for Substructural and Modal Logics. FSCD; ICFP Program Committee Meeting Workshop, 2017.

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. LFMTTP, 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. 2017.

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**

Bicubical Directed Type Theory. Carnegie Mellon University, 2018.

Dependent Modal Type Theory, Syntactically. Carnegie Mellon University, 2018.

A Fibrational Framework for Substructural and Modal Logics. Carnegie Mellon Uni-

versity, 2017.

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 Research in Sciences Seminar, 2017.

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**

IEEE/ACM Conference on Logic in Computer Science (LICS), 2018

ACM International Conference on Functional Programming (ICFP), 2017.

Formal Structures for Computation and Deduction (FSCD), 2016.

ACM International Conference on Functional Programming (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, 2016.

### **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**

AMS Math Research Communities on Homotopy Type Theory, 2017.

Programming Languages Mentoring Workshop (PLMW) at ICFP, 2017–2018.

Oregon Programming Languages Summer School (OPLSS), 2016–2017. 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<sup>th</sup> 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 115: How to Design Programs**

Fall, 2017 and Spring, 2018

This is an introductory computer science course for non-majors and potential majors. The course introduces students to functional programming with numbers, strings, images, aggregates, choices, and lists. Throughout the course, students use two methodologies, one for designing functions, and the other for designing reactive programs, to go from informal problem descriptions to code. This methodology emphasizes the use of examples in designing functions, and good software engineering practices, such as writing documentation and tests.

#### **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, Spring 2016, Spring 2017

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 312: Algorithms**

Spring, 2017

This is a required core course on the design and analysis of algorithms. Students learn a toolbox of algorithm design techniques, such as reduction, inductive techniques (contraction, divide and conquer, dynamic programming, greediness), and randomization. The course emphasizes parallel algorithms, and integrates both work and span analysis throughout the semester.

**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, 2014–2018.

Graduate Council BA/MA Admissions Subcommittee, 2015–2016, 2017–2018.

**Department Service**

CS search committee, 2016–2017, 2017–2018.

Math/CS Graduate Education Committee, 2014–2018.

Library committee, 2013–2014.

**Research  
Advisees:**

**Wesleyan University**



Mitchell Riley PhD, 2016–present  
Working on modal type theory.

Joesph Cutler Undergraduate, 2017–present  
Learning basic programming language theory.

Emily Black Undergraduate, 2015–2017  
Formalized red-black tree deletion in Agda. BA honors thesis, *Towards a Formal Verification of Courcelle’s Theorem*, 2017. Proceeded to PhD program at Carnegie Mellon.

Joomy Korkut Undergraduate, 2015–2017, MA 2017–2018  
Formalization of regular expression matching in Agda. BA honors thesis, *Verified Compilation of ML5 to Javascript*, 2017. MA thesis on programmable editors for proof assistants, 2018. Proceeding to PhD program in 2018–2019.

Max Trifunovski Undergraduate, 2015–2017  
Formalization of regular expression matching in Agda. BA honors thesis, *An Interactive Proof Assistant for Linear Logic*, 2017.

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.

### **Princeton University**

Matthew Weaver PhD, 2017–present  
Working on directed type theory.

### **Northeastern University**

Max S. New PhD, 2017–present  
Working on category theoretic semantics of gradual typing, a recent research area that blends static and dynamic typing.

### **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  $\lambda$ -calculus using a logical technique called focusing. Proceeded to the CS PhD program at CMU.

**PhD Thesis Committees:** Carlo Angiuli expected 2018  
Computational Higher-Dimensional Type Theory, Carnegie Mellon University.

Kuen-Bang Hou (Favonia) 2017

Higher-Dimensional Types in the Mechanization of Mathematics, Carnegie Mellon University.