NORMAN DANNER

Department of Mathematics and Computer Science
Wesleyan University, Science Tower 655
265 Church St.
Middletown, CT 06457

Phone: +1 860.685.2185
Fax: +1 860.685.2571
ndanner@wesleyan.edu
http://ndanner.web.wesleyan.edu

Education

Ph.D. Indiana University, Bloomington, 1999, Mathematics. Thesis title: *Ordinal Notations in Typed λ-Calculi*. Thesis advisor: Daniel Leivant.

B.A. University of California, Berkeley, 1991, Mathematics.

Employment

July 2009–present. Associate Professor of Computer Science, Wesleyan University, Department of Mathematics and Computer Science.

August 2002–June 2009. Assistant Professor of Computer Science, Wesleyan University, Department of Mathematics and Computer Science.

August 1999–August 2002. VIGRE Assistant Professor and Adjunct Assistant Professor, University of California, Los Angeles, Mathematics Department (Program in Computing).

Research Interests

My primary research interest is the development of formal tools for reasoning about complexity of higher-order languages. This research splits into several subareas. I am interested in developing techniques for automatic extraction of complexity information from higher-order programs into proof assistant environments, where formal reasoning can be carried out to provide certified complexity bounds. I am also interested in understanding complexity in less-explored type domains such as higher types and coinductively-defined data. I also apply these techniques to analyzing languages that are restricted by ramification and safe/normal distinctions in the style of implicit computational complexity.

Refereed Publications

Joseph W. Cutler, Daniel R Licata, and Norman Danner. Denotational recurrence extraction for amortized analysis. To appear in *ICFP 2020*, 2020.

G. A. Kavvos, Edward Morehouse, Daniel R. Licata, and Norman Danner. Recurrence extraction for functional programs through call-by-push-value. *Proceedings of the ACM on Programming Languages*, 4(POPL), 2019. doi:10.1145/3371083.

Norman Danner, Daniel R. Licata, and Ramyaa Ramyaa. Denotational cost semantics for functional languages with inductive types. In Kathleen Fisher and John Reppy, editors, *Proceedings of the 20th ACM SIGPLAN International Conference on Functional Programming, ICFP 2015*, page 140–151. ACM, 2015. doi:10.1145/2784731.2784749.

Norman Danner, Jennifer Paykin, and James S. Royer. A static cost analysis for a higher-order language. In Matthew Might and David Van Horn, editors, *Proceedings of the 7th workshop on Programming languages meets program verification*, page 25–34. ACM Press, 2013. doi:10.1145/2428116.2428123.

Norman Danner, Sam DeFabbia-Kane, Danny Krizanc, and Marc Liberatore. Effectiveness and detection of denial of service attacks in Tor. *Transactions on Information and System Security*, 15(3):11:1–11:25, 2012. doi:10.1145/2382448.2382449.

Norman Danner and James S. Royer. Two algorithms in search of a type system. *Theory of Computing Systems*, 45(4):787–821, 2009. doi:10.1007/s00224-009-9181-y.

Ralph Morelli, Allen Tucker, Norman Danner, Trishan R. De Lanerolle, Heidi J. C. Ellis, Ozgur Izmirli, Danny Krizanc, and Gary Parker. Revitalizing computing education through free and open source software for humanity. *Communications of the Association for Computing Machinery*, 52(8):67–75, 2009. doi:10.1145/1536616.1536635.

Norman Danner, Danny Krizanc, and Marc Liberatore. Detecting denial of service attacks in Tor. In Roger Dingledine and Philippe Golle, editors, *Financial Cryptography and Data Security: 13th International Conference, FC 2009*, volume 5628 of *Lecture Notes in Computer Science*, page 273–284. Springer-Verlag, 2009. doi:10.1007/978-3-642-03549-4_17.

Norman Danner and James S. Royer. Adventures in time and space. *Logical Methods in Computer Science*, 3(9):1–53, 2007a. doi:10.2168/LMCS-3(1:9)2007.

Norman Danner and James S. Royer. Time-complexity semantics for feasible affine recursions. In S.B. Cooper, B. Löwe, and A. Sorbi, editors, *Computation in the Real World (Proceedings Computability in Europe 2007, Sienna)*, volume 4497 of *Lecture Notes in Computer Science*, Berlin, 2007b. Springer-Verlag.

Chris Pollett and Norman Danner. Circuit principles and weak pigeonhole variants. *Theoretical Computer Science*, 383:115–131, 2007. doi:10.1016/j.tcs.2007.04.017.

Norman Danner and James S. Royer. Adventures in time and space. In *POPL '06: Conference Record of the 33rd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (Charleston, SC, 2006)*, page 168–179, New York, 2006. Association for Computing Machinery. doi:10.1145/1111037.1111053.

Norman Danner and Chris Pollett. The weak pigeonhole principle for function classes in S^1_2. *Mathematical Logic Quarterly*, 52(6):575–584, 2006. doi:10.1002/malq.200610015.

Chris Pollett and Norman Danner. Circuit principles and weak pigeonhole variants. In M. Atkinson and F. Dehne, editors, *Computing: The Australasian Theory Symposium (Newcastle, Australia, 2005)*, volume 41 of *Conferences in Research and Practice in Information Technology*, page 31–40, Sydney, 2005. Australian Computer Society.

Norman Danner and Chris Pollett. Minimization and NP multifunctions. *Theoretical Computer Science*, 318(1–2):105–119, 2004. doi:10.1016/j.tcs.2003.10.020.

Norman Danner. Ramified recurrence with dependent types. In *Typed Lambda Calculi and Applications (Kraków, 2001)*, volume 2044 of *Lecture Notes in Computer Science*, page 91–105. Springer, Berlin, 2001.

Norman Danner. Ordinals and ordinal functions representable in the simply typed λ-calculus. *Annals of Pure and Applied Logic*, 97(1–3):179–201, 1999.

Norman Danner and Daniel Leivant. Stratified polymorphism and primitive recursion. *Mathematical Structures in Computer Science*, 9(4):507–522, 1999. URL `http://journals.cambridge.org/article_S0960129599002868`.

Lawrence S. Moss and Norman Danner. On the foundations of corecursion. *Logic Journal of the IGPL*, 5(2):231–257, 1997. doi:10.1093/jigpal/5.2.231.

Eric Hammer and Norman Danner. Towards a model theory of diagrams. *Journal of Philosophical Logic*, 25(5):463–482, 1996. doi:10.1007/BF00257381. Reprinted as "Towards a model theory of Venn diagrams" in G. Allwein and J. Barwise (eds.), *Logical Reasoning with Diagrams*, pp. 109–127, Oxford University Press, 1996).

Non-refereed publications and manuscripts

Norman Danner and Daniel R. Licata. Denotational semantics as a foundation for cost recurrence extraction for functional languages. In preparation. URL `https://arxiv.org/abs/2002.07262`.

Other publications

Norman Danner. *Ordinal Notations in Typed λ-Calculi.* PhD thesis, Indiana University, Bloomington, 1999.

Norman Danner. Book review of *Derivation and Computation: Taking the Curry-Howard Correspondence Seriously* (by H. Simmons). *Bulletin of Symbolic Logic*, 7(3):380–383, 2001.

Conference and Workshop Presentations

NII Shonan Seminar on Higher Order Complexity Theory and its Applications, 2019, Japan. *Cost recurrence extraction for higher-order languages.*

Dagstuhl Seminar on Resource Bounded Analysis, 2017, Germany. *Formalizing extract-and-solve for complexity analysis.*

Yale Programming Languages Day 2015. *Provably correct extraction of cost recurrences from higher-order programs.*

International Conference on Functional Programming 2015, Vancouver. *Denotational cost semantics for functional languages with inductive types.*

Workshop on Higher Order Computation: Types, Complexity, Applications 2014, Paris. (invited workshop) *Ramified structural recursion and corecursion.*

Workshop on Programming Languages meets Program Verification 2013, Rome. *A static cost analysis for a higher-order language.*

Hot Topics in Privacy Enhancing Technologies 2011. *Simulating circuit creation in Tor: Preliminary results.*

Technical Symposium on Computer Science Education 2008, Portland. *Teaching and building humanitarian open source software.*

Workshop on Implicit Computational Complexity 2008, Villetaneuse. *Affine tiered recursion: semantics and pragmatics.*

Computability in Europe 2007, Siena. *Two algorithms in search of a type system.*

Typed λ-Calculi and Applications 2001, Kraków. *Ramified recurrence and dependent types.*

Association for Symbolic Logic Annual Meeting 1999, San Diego. *Stratified polymorphism and primitive recursion.*

Association for Symbolic Logic Annual Meeting 1998, Toronto. *Ordinal notations in typed λ-calculi.*

External Funding

National Science Foundation CCF/SHF. *Certified cost recurrences for higher-order functional programs* (PI with Daniel R. Licata, Wesleyan University). This grant funds our research into extracting certified cost information from program source code. June 2016–August 2020. Amount: $461,830.

National Science Foundation CCF/SHF. *Complexity and feasibility for programs over coinductively-defined data* (PI with James Royer, Syracuse Univesrity). This NSF grant funds my research with James Royer in implicit complexity for coinductive data, and includes funding

for two undergraduate summer research internships. July 2013–June 2016. Amount: $122,340 (Wesleyan); $309,550 (total).

National Science Foundation CPATH II. *Building a Community to Incorporate Humanitarian Free and Open Source Software into Undergraduate Computing Education.* (PI with Ozgur Izmirli [Connecticut College], Danny Krizanc [Wesleyan University], Ralph Morelli [Trinity College], Gary Parker [Connecticut College]). This grant continues our CPATH CB grant and focuses on exporting the model we have developed for incorporating HFOSS development into the undergraduate curriculum. Fall 2009–Summer 2011. Amount: $155,500 (Wesleyan); $800,000 (total).

National Science Foundation CPATH CB Collaborative Grant. *Can humanitarian open-source development help revitalize undergraduate computing education?* (Co-PI with Ozgur Izmirli [Connecticut College], Danny Krizanc [Wesleyan University], Ralph Morelli [Trinity College], Gary Parker [Connecticut College]) The activities funded by this grant center around using open-source software development, with a focus on software for humanitarian and community needs, as a tool for attracting, motivating, and retaining undergraduate students in Computer Science. Fall 2007–Summer 2009. Amount: $70,825 (Wesleyan); $496,429 (Total).

Mellon Foundation. (Wesleyan steering committee representative) This joint grant (with Connecticut College and Trinity College) funds pedagogical and research initiatives in Computer Science departments at small liberal arts institutions. Fall 2005–Summer 2010. Amount: $800,000 (Total).

Professional Activities

Workshop chair: *Logic and Computational Complexity (LCC) 2017* (co-chair).

Steering committee: *Logic and Computational Complexity (LCC)*, 2018-present.

Program committee: *Developments in Implicit Computational Complexity (DICE) 2015.*

Panel reviewer: National Science Foundation CISE 2017.

Referee: *Annals of Pure and Applied Logic*, 2003, 2006; *Computer Science Logic*, 2010, 2011; *Information and Computation*, 1997, 2002, 2012, 2015; *International Conference on Automata, Languages, and Programming*, 2009; *Journal of Functional Programming*, 2002; *Journal of Logic and Computation*, 2011; *Logic in Computer Science*, 1999, 1996. *Logic of Programming and Automated Reasoning*, 2000; *Logical Methods in Computer Science*, 2008, 2019; *Mathematical Foundations of Computer Science*, 2010; *Proof Theory and Computer Science*, 2001; *SIAM Journal on Computing*, 2004; *Transactions on Computational Logic*, 2014. *TYPES Conference*, 2003; *Workshop on Functional and Logic Programming*, 2010.

Reviewer: *Mathematical Reviews*, 2000–2003; *Bulletin of Symbolic Logic*, 2001–2002.

Other professional service: Doctoral consortium reviewer, ACM Richard Tapia Celebration of Diversity in Computing 2015–2017.

Teaching Experience

Wesleyan University, Department of Computer Science. Undergraduate: Cryptography (first-year seminar); Privacy and security (first-year seminar); Introduction to Programming; How to Talk to Machines; Computer Science I; Data Structures/Computer Science II; Design and Analysis of Algorithms; Design of Programming Languages; Programming Language Implementation; Programming Methods/Software Development; Computer Graphics; Data Analysis. Graduate: Logic and Discrete Mathematics; Proof Theory; Topics in Theoretical Computer Science.

University of California, Los Angeles, Program in Computing. Undergraduate: Introduction to computers; C++ programming (first- and second-quarter); Java programming (first- and second-quarter); Computability and complexity theory.

August 1994–May 1999. Associate Instructor (Graduate Student Instructor), Indiana University, Bloomington, Mathematics Department. Responsible for assisting large lecture courses, as well as teaching *College Algebra*, *Finite Mathematics*, *Brief Calculus*, *Mathematics for Elementary Education Majors*, and *Introduction to Mathematical Problem Solving*.