

Jana Dunfield

School of Computing
Queen's University
Goodwin Hall Room 557
Kingston, ON K7L 3N6, Canada

jd169@queensu.ca
613-533-6000 x78432
<http://research.cs.queensu.ca/~jana>

- Research interests** Programming languages: type inference and type-based software verification; incremental computation; functional programming; automated deduction.
- Teaching experience** Functional programming (Carnegie Mellon; McGill; Queen's); formal logic and logic programming (Queen's); programming languages (Carnegie Mellon; McGill; UBC; Queen's); software quality assurance (Queen's).
- Personal** US and Canadian citizen.
- Positions** Assistant Professor, School of Computing, Queen's University (Kingston, Ontario), Aug. 2017–present.
Sessional Lecturer, Department of Computer Science, University of British Columbia (Vancouver), Sept.–Dec. 2016, Sept.–Dec. 2015.
Research Associate (non-tenure-track faculty), Department of Computer Science, University of British Columbia (Vancouver), May 2014–May 2017.
Postdoctoral Researcher, Max Planck Institute for Software Systems (Kaiserslautern and Saarbrücken, Germany), Sept. 2010–May 2014.
Postdoctoral Fellow, School of Computer Science, McGill University, Sept. 2007–Aug. 2010.
Course Lecturer, School of Computer Science, McGill University, Jan.–April 2010.
Teaching Assistant, School of Computer Science, McGill University, Jan.–March 2008.
- Education** Ph.D. Computer Science, Carnegie Mellon University (Pittsburgh, PA, USA), August 2007.
Dissertation title: *A Unified System of Type Refinements*.
Committee: Frank Pfenning (chair), Jonathan Aldrich, Robert Harper, Benjamin Pierce (University of Pennsylvania).
US National Science Foundation Graduate Research Fellowship, 2000–2003.
B.S. (high honors) Computer Science, Portland State University (Portland, Oregon, USA), August 2000.
- Grants and awards** Queen's University Research Initiation Grant, C\$60,000 (total; awarded 2017; closed 2023).
Programming languages for scalable incremental computation and advanced gradual typing.
NSERC Discovery Grant, C\$243,500 (total, 2018–2026; 2019–'20 instalment deferred; includes C\$12,500 Discovery Launch Supplement, 1-year ECR extension, and 1-year COVID-19 extension).¹
NSERC COVID-19 Supplement, C\$5,280 (total; 2020–2021).
PPDP 2008 “Most Influential Paper” (10-year test of time) award (with Brigitte Pientka) for “Programming with proofs and explicit contexts”, received September 2018.

¹The Launch Supplement and the 2018–2019 instalment appear in the NSERC Awards Database under my previous first name.

**Journal and
conference
papers**

Dimitrios J. Economou, Neel Krishnaswami, and Jana Dunfield. Focusing on refinement typing.
ACM Transactions on Programming Languages and Systems 45(4), 2023.

Jana Dunfield and Neel Krishnaswami. Bidirectional typing.
ACM Computing Surveys 54(5), May 2021.

Jana Dunfield and Neelakantan R. Krishnaswami. Sound and complete bidirectional typechecking for higher-rank polymorphism with existentials and indexed types.
arXiv:1601.05106 [cs.PL]. *46th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL '19)*. (77/267 \approx 28% accepted.)

Jana Dunfield. Extensible datasort refinements.
In *European Symposium on Programming (ESOP '17)*, pp. 476–503, April 2017.
(36/112 \approx 32% accepted.)

Khurram A. Jafery and Jana Dunfield. Sums of uncertainty: refinements go gradual.
In *44th ACM SIGPLAN Symposium on Principles of Programming Languages (POPL '17)*, pp. 804–817, 2017. (64/282 \approx 23% accepted.)

Matthew A. Hammer, Jana Dunfield, Kyle Headley, Nicholas Labich, Jeffrey S. Foster, Michael Hicks and David Van Horn. Incremental computation with names.
In *ACM SIGPLAN Int'l Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA)*, 2015. (53/210 \approx 25% accepted.)
Implementation accepted by the OOPSLA AEC (Artifact Evaluation Committee).

Jana Dunfield. Elaborating evaluation-order polymorphism.
In *20th ACM SIGPLAN Int'l Conference on Functional Programming (ICFP '15)*, 2015. (35/119 \approx 29% accepted.)

Jana Dunfield. Elaborating intersection and union types.
Journal of Functional Programming 24(2–3), pp. 133–165, 2014 (Cambridge Univ. Press).

Yan Chen, Jana Dunfield, Matthew A. Hammer and Umut A. Acar.
Implicit self-adjusting computation for purely functional programs.
Journal of Functional Programming 24(1), pp. 56–112, 2014 (Cambridge Univ. Press).

Jana Dunfield and Neelakantan R. Krishnaswami.
Complete and easy bidirectional typechecking for higher-rank polymorphism.
In *18th ACM SIGPLAN Int'l Conference on Functional Programming (ICFP '13)*, pp. 429–441, Sept. 2013. (40/133 \approx 30% accepted.)

Jana Dunfield. Elaborating intersection and union types.
In *17th ACM SIGPLAN Int'l Conference on Functional Programming (ICFP '12)*, pp. 17–28, Sept. 2012. (32/88 \approx 36% accepted.)
Selected for an invitation to submit an extended version to *J. Functional Programming*.

Yan Chen, Jana Dunfield and Umut A. Acar. Type-directed automatic incrementalization.
In *33rd ACM SIGPLAN Conf. on Programming Language Design and Implementation (PLDI '12)*, pp. 299–310, June 2012. (48/255 \approx 19% accepted.)

Yan Chen, Jana Dunfield, Matthew A. Hammer and Umut A. Acar.
Implicit self-adjusting computation for purely functional programs.
In *16th ACM SIGPLAN Int'l Conference on Functional Programming (ICFP '11)*, pp. 129–141, September 2011. (33/92 \approx 36% accepted.)

Brigitte Pientka and Jana Dunfield.
Beluga: a framework for programming and reasoning with deductive systems
(System Description).
In *Int'l Joint Conference on Automated Reasoning (IJCAR '10)*, pp. 15–21,
2010. (40/89 \approx 45% accepted.)

- Journal and conference papers, continued**
- Brigitte Pientka and Jana Dunfield. Programming with proofs and explicit contexts. In *10th Int'l ACM Symposium on Principles and Practice of Declarative Programming (PPDP '08)*, pp. 163–173, 2008. (24/51 \approx 47% accepted.) “Most Influential Paper” award, received at PPDP 2018.
- Jana Dunfield and Frank Pfenning. Tridirectional typechecking. In *31st ACM Symposium on Principles of Programming Languages (POPL '04)*, pp. 281–292, 2004. (29/176 \approx 16% accepted.)
- Jana Dunfield and Frank Pfenning. Type assignment for intersections and unions in call-by-value languages. In *6th Int'l Conf. on Foundations of Software Science and Computation Structures (FoSSaCS '03)*, pp. 250–266, 2003. (26/94 \approx 28% accepted.)
- Workshop papers (refereed)**
- Jana Dunfield. Annotations for intersection typechecking. In pre-proceedings of *Workshop on Intersection Types and Related Systems (ITRS '12)*, June 2012; post-proceedings in *EPTCS 121*, 2013, pp. 35–47.
- Jana Dunfield. Untangling typechecking of intersections and unions. In: pre-proceedings of *Workshop on Intersection Types and Related Systems (ITRS '10)*, pp. 59–70, 2010; post-proceedings in *EPTCS 45*, 2011.
- Jana Dunfield. Greedy bidirectional polymorphism. In *ACM SIGPLAN Workshop on ML (ML '09)*, pp. 15–26, 2009.
- Jana Dunfield and Brigitte Pientka. Case analysis of higher-order data. In *Int'l Workshop on Logical Frameworks and Meta-languages: Theory and Practice (LFMTP '08)*, pp. 69–84, 2008. Elsevier *ENTCS 228*.
- Jana Dunfield. Refined typechecking with Stardust. In *Workshop on Programming Languages meets Program Verification (PLPV '07)*, pp. 21–32, 2007.
- Dissertation**
- Jana Dunfield. *A Unified System of Type Refinements*. Carnegie Mellon University, August 2007. Technical Report CMU-CS-07-129.
- Technical reports**
- Jana Dunfield and Frank Pfenning. Tridirectional typechecking. Carnegie Mellon University technical report CMU-CS-04-117, 2004. (Extended version of POPL '04 paper.)
- Jana Dunfield. Combining two forms of type refinements. Carnegie Mellon University technical report CMU-CS-02-182, 2002.
- Unpublished work**
- Jana Dunfield. Flattening subtyping by eta expansion. arXiv:2412.19053 [cs.PL]. December 2024.
- Selected talks**
- Uncertainty is hope: towards a unified foundation of gradual typing (Logic and Semantics Seminar, University of Cambridge, April 2019).
- Uncertainty is Hope: logic and gradual typing (invited tutorial, MFPS '18).
- Gradual typing for refinements (Queen's University, Kingston, May 2017; Brock University, St. Catharines, June 2017).
- Elaborating evaluation-order polymorphism (ICFP '15).
- Complete and easy bidirectional typechecking for higher-rank polymorphism (ICFP '13).
- Elaborating intersection and union types (ICFP '12).
- Verifying functional programs with type refinements (IMDEA Software Institute, Madrid, Feb. 2010; Max Planck Institute for Software Systems, Saarbrücken, Mar. 2010).
- Greedy bidirectional polymorphism (ML Workshop '09).
- Programming with proofs and explicit contexts (PPDP '08).
- Case analysis of higher-order data (LFMTP '08).
- Refined typechecking with Stardust (PLPV '07).
- Tridirectional typechecking (POPL '04).
- Type assignment for unions and intersections in call-by-value languages (FoSSaCS '03).

Supervision Graduate:

- Graduate supervision (School of Computing, Queen's University)
 - December Stuart (MSc), supervised since 2022
- Alumni:
 - Dimitrios Economou (PhD, graduated 2024)
 - Nicholas Mertin (MSc, graduated 2025), co-supervised (with Karen Rudie, Electrical and Computer Engineering)
- PhD supervisory committee (School of Computing, Queen's University)
 - Lama Moukahal (2018–, graduated 2021)
 - Majid Babaei (2018–, graduated 2021)
- Examinations and defences (School of Computing, Queen's University)
 - Amin Marwan Fakhereldine (head's representative, PhD defence, 2024)
 - Ahmed Awad (chair, MSc defence, 2024)
 - Ella Morgan (chair, MSc defence, 2023)
 - Annabelle Sauve (internal examiner, MSc defence, 2023)
 - Dimitrios Economou (examiner, PhD comprehensive examination, 2022)
 - Armstrong Foundjem (head's representative, PhD defence, 2022)
 - Sindhura Thirumal (chair, MSc defence, 2021)
 - Hunter Orr (chair, MSc defence, 2021)
 - Lama Moukahal (examiner, PhD comprehensive examination, 2020)
 - Majid Babaei (examiner, PhD comprehensive examination, 2020)
 - Nafiseh Kahani (head's representative, PhD defence, 2020)
 - Benjamin Cecchetto (head's representative, PhD defence, 2019)
 - Marwa Elsayed (head's representative, PhD defence, 2018)
 - Heng Li (examiner, PhD comprehensive examination, 2018)
 - Mojtaba Bagherzadeh (examiner, PhD comprehensive examination, 2017)
 - Mohamed Sami Rakha (internal examiner, PhD defence, 2017)
- PhD RPE (Research Proficiency Evaluation) committee member (Computer Science, University of British Columbia)
 - Joey Eremondi (2017); Felipe Bañados Schwerter (2015)

Undergraduate:

- 3-student CISC 499 project: supervisor, January–April 2022 (School of Computing, Queen's University)
- 3-student CISC 499 project: co-supervisor, January–April 2021 (School of Computing, Queen's University)
- 1-student CISC 499 project: supervisor, January–April 2020 (School of Computing, Queen's University)
- Undergraduate research supervisor, May–July 2019 (School of Computing, Queen's University)
- Two 1-student CISC 499 projects: supervisor, January–April 2019 (School of Computing, Queen's University)
- 5-student CISC 498 project: supervisor, Sept. 2018–April 2019 (School of Computing, Queen's University)
- 1-student CPSC 448 Directed Studies supervisor, January–May 2017 (Computer Science, University of British Columbia)

Service **Service to the research community:**

- **Conference programme committee member:**
 - ESOP 2025; OOPSLA 2023; ESOP 2022; ESOP 2016; ICFP 2016; PPDP 2014
- **Conference external review committee member:** ICFP 2018
- **Workshop programme committee member:**
 - IC 2019; TFP 2019; ITRS 2010, 2016, 2018; MLW 2011
- **Journal reviewer:** *Information and Computation; ACM Computing Surveys; Computer Languages, Systems & Structures; ACM Transactions on Programming Languages and Systems; Journal of Functional Programming*
- **Conference reviewer:** CSL, ESOP, ICFP, LICS, PLDI, POPL, SAC
- **Workshop reviewer:** CPP, LFMT, PLPV, RTA, WFLP
- **Grant reviewing:** External reviewer for an NSERC Discovery grant (2021–2022)
- Mentor, SIGPLAN-M (SIGPLAN Long-Term Mentoring Committee), 2023–present
- Panellist, *Academic Career Planning During a Crisis*, affiliated with ICFP 2020
- Member, Name Change Policy Working Group (<https://ncpwg.org>)

Administrative service:

- **Departmental (School of Computing):**
 - Appointments Committee member and Employment Equity representative, 2019–2020
 - Renewal, Tenure and Promotion Committee member and Employment Equity representative, 2020–2021
 - Undergraduate Committee member, October 2017–present
 - Equity, Diversity and Inclusion Committee member, 2019–2023 (chair, 2022–2023)
 - Awards Committee, 2018–2019
 - NSERC USRA ranking committee, 2020
 - Department representative to the Faculty of Arts and Science:
 - * EDII Council, 2022–2023
 - * Faculty Board, 2023–2025
- **University-wide:**
 - QUAQE (Queen’s University Association for Queer Employees):
 - * Organizing Committee, 2022–present
 - * Education and Outreach Subcommittee member, 2021–2022
 - * Member Support and Advocacy Subcommittee member, 2021–2022

- Teaching** CISC 465+865 Semantics of Programming Languages, Queen’s University, Winter 2024 (465+865) Winter 2023 (465+865) Winter 2022 (465+865), Summer 2022 (465), Winter 2021 (465+865), Winter 2020 (465+865), Winter 2019 (865), Winter 2018 (865). Enrolments: **32+4, 11+6, 23+6, 1, 19+6, 12+6, 6, 7.**
- Graduate course, combined with undergraduate CISC 465 since Winter 2020.
- Lectured (modified during pandemic); designed new course, created content and lecture notes; created and marked assignments; supervised 0–1 teaching assistants; held office hours.
- CISC 204 Logic for Computing Science, Queen’s University, Fall 2019 (September–December 2019). Enrolment: **350.**
- Lectured; created course content and lecture notes; created exams; supervised 12 teaching assistants; held office hours.
- CISC 360 Programming Paradigms, Queen’s University, Fall 2024, Fall 2023, Winter 2023, Fall 2022, Winter 2022, Fall 2021, Winter 2021, Fall 2020, Fall 2019.
- Required for all Computing undergraduates. Enrolments: **279, 244, 136, 156, 123, 148, 89, 235, 57.**
- Lectured (except Fall 2020 / Winter 2021); revised CISC 260 course content and created lecture notes; created assignments and exams; supervised teaching assistants; held office hours.
- CISC 260 Programming Paradigms, Queen’s University, Fall 2018 (September–December 2018).
- Required for all Computing undergraduates. Enrolment: **73.**
- Lectured; created course content and lecture notes; created assignments and exams; marked exams; supervised two teaching assistants; held office hours.
- CISC 327 Software Quality Assurance, Queen’s University, Fall 2018 (enrolment: **120**) and Fall 2017 (enrolment: **140**). Required for Software Design undergraduates.
- Lectured; updated course content and lecture notes; created assignments and exams; marked exams; supervised four teaching assistants; held office hours.
- CPSC 311 Definition of Programming Languages, University of British Columbia, 2016 Winter 1 (September–December 2016) and 2015 Winter 1 (September–December 2015). Enrolment: **119, 108.**
- Sessional lecturer. Lectured; updated course content and wrote lecture notes; created assignments and exams; marked portions of exams; supervised five teaching assistants; held office hours.
- COMP 302 Programming Languages and Paradigms, McGill University, Winter 2010. Enrolment at end of term: **61.**
- Course lecturer. Lectured, wrote and revised lecture notes, created assignments and exams, marked portions of assignments and exams, supervised teaching assistants, and held office hours.
- COMP 523 Language-Based Security, McGill University, Winter 2008. Master’s-level graduate course. Teaching assistant for Brigitte Pientka. Marked assignments and held office hours. (Ended early due to a teaching assistant strike.)
- 15–312 Foundations of Programming Languages, Carnegie Mellon University, Fall 2002. Satisfies area requirement for undergraduate CS majors. Held weekly recitations, guest lectured, prepared and graded assignments, assisted in exam creation and grading, and held office hours.
- 15–212 Principles of Programming, Carnegie Mellon University, Spring 2001. Required for undergraduate CS majors; typically taken in the second year. Held weekly recitation, prepared and graded assignments, assisted with exam creation and grading, and held office hours.