# Joshua Dunfield

joshdunf@cs.ubc.ca *http://www.cs.ubc.ca/~joshdunf*

***Research interests*** Programming languages: functional programming; type inference and type-based software verification; incremental computation; automated deduction.

***Teaching experience*** Functional programming (Carnegie Mellon; McGill); programming languages (Carnegie Mellon; McGill; UBC).

***Personal*** United States citizen; Canadian permanent resident.

***Positions***

Sessional Lecturer, Department of Computer Science, University of British Columbia (Vancouver), September–December 2016, September–December 2015.

Research Associate (non-tenure-track faculty), Department of Computer Science, University of British Columbia (Vancouver), May 2014–present.

Postdoctoral Researcher, Max Planck Institute for Software Systems (Kaiserslautern and Saarbrücken, Germany), September 2010–May 2014.

Postdoctoral Fellow, School of Computer Science, McGill University (Montréal), September 2007–August 2010.

Course Lecturer, School of Computer Science, McGill University (Montréal), January–April 2010.

Teaching Assistant, School of Computer Science, McGill University (Montréal), January–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.

***Journal and conference papers***

Joshua Dunfield.
Extensible datasort refinements. To appear in *European Symposium on Programming (ESOP '17)*, April 2017.

Khurram A. Jafery and Joshua Dunfield.
Sums of uncertainty: refinements go gradual.
To appear in *44th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL '17)*, 2017.

Matthew A. Hammer, Joshua 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 ≈ 25% accepted.)
Implementation accepted by the OOPSLA AEC (Artifact Evaluation Committee).

*continued on next page*

***Journal and conference papers (cont.)***

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

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

Yan Chen, Joshua 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).

Joshua 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 ≈ 30% accepted.)

Joshua 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 ≈ 36% accepted.)
Selected for an invitation to submit an extended version to *J. Functional Programming*.

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

Yan Chen, Joshua 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 ≈ 36% accepted.)

Brigitte Pientka and Joshua 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 ≈ 45% accepted.)

Brigitte Pientka and Joshua 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 ≈ 47% accepted.)

Joshua Dunfield and Frank Pfenning. Tridirectional typechecking.
In *31st ACM Symposium on Principles of Programming Languages (POPL '04)*, pp. 281–292, 2004. (29/176 ≈ 16% accepted.)

Joshua 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 ≈ 28% accepted.)

***Drafts***

Matthew A. Hammer, Joshua Dunfield, Dimitrios J. Economou and Monal Narasimhamurthy.
Typed Adapton: Refinement types for incremental computations with precise names.
arXiv:`1610.00097 [cs.PL]`.

Joshua Dunfield and Neelakantan R. Krishnaswami. Sound and complete bidirectional typechecking for higher-rank polymorphism and indexed types.
arXiv:`1601.05106 [cs.PL]`. Draft, January 2016.

***Workshop papers (refereed)***

Joshua 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.

Joshua 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.

Joshua Dunfield. Greedy bidirectional polymorphism.
In *ACM SIGPLAN Workshop on ML (ML '09)*, pp. 15–26, 2009.

Joshua 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.

Joshua Dunfield. Refined typechecking with Stardust.
In *Workshop on Programming Languages meets Program Verification (PLPV '07)*, pp. 21–32, 2007.

***Dissertation***

Joshua Dunfield. *A Unified System of Type Refinements*.
Carnegie Mellon University, August 2007.
Published as Technical Report CMU-CS-07-129.

***Technical reports***

Joshua Dunfield and Frank Pfenning. Tridirectional typechecking.
Carnegie Mellon University technical report CMU-CS-04-117,
2004. (Extended version of POPL '04 paper.)

Joshua Dunfield. Combining two forms of type refinements.
Carnegie Mellon University technical report CMU-CS-02-182, 2002.

***Selected talks***

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).

***Service***

**Advising:**

- Felipe Bañados Schwerter: PhD RPE (Research Proficiency Evaluation) committee member, 2015 (Computer Science, University of British Columbia)

**Reviewing:**

- **Programme committee member**:
  - ITRS 2010
  - MLW 2011
  - PPDP 2014
  - ESOP 2016
  - ICFP 2016
  - ITRS 2016
- **Journal reviewer:** *Information and Computation*, *ACM Computing Surveys*.
- **Conference reviewer:** CSL, ESOP, ICFP, LICS, PLDI, POPL, SAC.
- **Workshop reviewer:** CPP, LFMTP, PLPV, RTA, WFLP.

***Teaching*** CPSC 311 Definition of Programming Languages, University of British Columbia, 2016 Winter 1 (September–December 2016).

Undergraduate elective. Enrolment at end of term: **119**.

**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.

CPSC 311 Definition of Programming Languages, University of British Columbia, 2015 Winter 1 (September–December 2015).

Undergraduate elective. Enrolment at end of term: **108**.

**Sessional Lecturer.** Lectured; updated course content and wrote lecture notes; created assignments and exams; marked portions of exams; supervised three teaching assistants; held office hours.

COMP 302 Programming Languages and Paradigms, McGill University, Winter 2010.

Required for undergraduates in CS and several other majors; typically taken in U1 or U2 (Québec system; roughly, 2nd and 3rd year). 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. Student evaluations (1–5 scale):

| | |
|---|---|
| "Overall, this is an excellent course" | mean 4.0 |
| "Overall, this instructor is an excellent teacher" | mean 4.2 |
| union of all questions | mean 4.2 |

A summary of the student evaluations is attached to my teaching statement.

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.

Teaching assistant for Frank Pfenning. Held weekly recitations, guest lectured, prepared and graded assignments, assisted in exam creation and grading, and held office hours. Student evaluations (1–4 scale):

| | min. | mean | max. |
|---|---|---|---|
| "Overall effectiveness" | 3 | 3.73 | 4 |
| union of all questions | 3 | 3.68 | 4 |

15–212 Principles of Programming, Carnegie Mellon University, Spring 2001.

Required for undergraduate CS majors; typically taken in the second year.

Teaching assistant for Michael Erdmann and Jeannette Wing. Held weekly recitation, prepared and graded assignments, assisted with exam creation and grading, and held office hours.

*(References on following page)*

***References*** Available upon request.