

Contents

1	General Information	2
1.1	Personal	2
1.2	Brief Biography	2
1.3	Education	3
1.4	Professional Appointments	4
1.5	Honors	4
1.6	Professional and Scholarly Associations	4
2	Research	5
2.1	Research Summary	5
2.2	Publications	9
2.3	Patents and Software Systems	19
2.4	Talks, Tutorials, and Summer School Lectures	19
2.5	Funding	24
3	Teaching	26
3.1	Vision for Teaching	26
3.2	Teaching at UCLA	26
3.3	Teaching at Purdue University	27
3.4	Teaching at other Institutions	31
3.5	Current Students	31
3.6	Former Students	31
4	Service	33
4.1	Service to Purdue University	33
4.2	Service to UCLA	34
4.3	Journal Editorial Boards	34
4.4	Guest Editor	35
4.5	External Review Committees	35
4.6	U.S. National Science Foundation Review Panels	35
4.7	France’s National Research Agency Evaluation Committees	35
4.8	Conference Program Committees	36
4.9	Conference Steering Committees	39
4.10	Conference Organizing Committees	39
4.11	Service to Professional Societies	40
4.12	Other Professional Services	40

1 General Information

1.1 Personal

Professor of Computer Science
UCLA Computer Science Department
4531K Boelter Hall
Los Angeles, CA 90095-1596

Web: <http://www.cs.ucla.edu/~palsberg>
Email: palsberg@ucla.edu
Phone: +1 310-825-6320
Fax: +1 310-794-5057

Born November 7, 1964 in Denmark. Permanent resident of the United States.

1.2 Brief Biography

Jens Palsberg is a Professor of Computer Science at UCLA. He received a Ph.D. in Computer Science from University of Aarhus, Denmark in 1992. In 1992–1996 he was a visiting scientist at various institutions, including MIT. In 1996–2002 he was an Associate Professor and, in 2002–2003, Professor of Computer Science at Purdue University. His research interests span the areas of compilers, embedded systems, programming languages, software engineering, and information security. He has authored over 80 technical papers, co-authored the book *Object-Oriented Type Systems*, and co-authored the 2002 revision of Appel’s textbook on *Modern Compiler Implementation in Java*. He is the recipient of National Science Foundation CAREER and ITR awards, a Purdue University Faculty Scholar award, an IBM Faculty Award, and an Okawa Foundation research award. Dr. Palsberg’s research has also been supported by DARPA, Intel, and British Telecom. Dr. Palsberg is an associate editor of *ACM Transactions of Programming Languages and Systems*, a member of the editorial board of *Information and Computation*, and a former member of the editorial board of *IEEE Transactions on Software Engineering*. He is serving as the vice chair of ACM SIGBED, Special Interest Group on Embedded Systems, and he has served as vice chair of computer science at UCLA, as associate head of computer science at Purdue University, as the general chair of the ACM Symposium on Principles of Programming Languages (POPL) and International Workshop on Model Checking of Software (SPIN), as conference chair of the IEEE Symposium on Logic in Computer Science (LICS), as a program chair for ACM Symposium on Principles of Programming Languages (POPL), the Static Analysis Symposium (SAS), Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS), Conference on Embedded Systems Software (EMSOFT), Conference on Formal Methods and Programming Models for Co-Design (MEMOCODE), Symposium on Requirements Engineering for Information Security (SREIS), and ACM Workshop on Program Analysis for Software Tools and Engineering (PASTE), and he has been a member of more than 50 other conference program committees.

1.3 Education

Ph.D. Computer Science, University of Aarhus, Denmark, 1992.

Thesis title: “Provably correct compiler generation”. Advisor: Peter Mosses.

M.Sc. Computer Science and Mathematics, University of Aarhus, Denmark, 1988.

1.4 Professional Appointments

7/03–present Professor of Computer Science, UCLA, USA.

7/05–present Graduate Vice Chair, Computer Science, UCLA, USA.

7/03–5/06 Adjunct Professor of Computer Science, Purdue University, USA.

8/02–7/03 Professor and Associate Head of Computer Science, Purdue University, USA.

5/02–8/02 Visiting Scholar, University of California, San Diego, USA.

8/96–7/02 Associate Professor of Computer Science, Purdue University, USA.

6/95–8/96 Visiting Scientist, Massachusetts Institute of Technology, USA.

7/94–6/95 Research Assistant Professor, University of Aarhus, Denmark.

9/93–6/94 Visiting Assistant Professor, Northeastern University, USA.

8/91–8/93 Research Associate, University of Aarhus, Denmark.

1.5 Honors

- National Science Foundation CAREER Award, 1998.
- Purdue University Faculty Scholar, 1999–2004, in recognition of outstanding academic distinction.
- One of the Ten Best Teachers of Undergraduates in the School of Science, Purdue University, for 2001, as selected by junior and senior science students.
- Okawa Foundation Research Award, 2003.
- IBM Faculty Award, 2005.
- ACM Distinguished speaker 2006–present.

1.6 Professional and Scholarly Associations

Association for Computing Machinery (ACM), ACM SIGPLAN, ACM SIGBED.

2 Research

In May 2003, Palsberg was listed as the 1055th most cited author in computer science (of 659,481 authors) by the NECI Scientific Literature Digital Library, available at

<http://citeseer.nj.nec.com/allcited.html>.

2.1 Research Summary

Palsberg's research spans the areas of compilers, embedded systems, programming languages, software engineering, and information security. The goal of most of his research is the discovery of principles and techniques that enable easier writing and understanding of programs, more reliable reasoning about the correctness and safety of programs, and faster and more portable implementations of programs. To show the viability of research results, researchers often demonstrate their ideas on toy languages. From time to time, some of the ideas are picked up by people who design languages for serious programming. A recent example is the Java language where well-tested ideas such as objects, static typing, interfaces, and garbage collection are key facets of the design.

Palsberg's research interests include:

1. resource-aware compilation,
2. type inference and static analysis for object-oriented software,
3. software evolution,
4. information security, and
5. partial evaluation.

The following is a survey of some of his accomplishments and plans.

Resource-Aware Compilation

Real-time, reactive and embedded systems are widely and increasingly used throughout society (e.g., flight control, railway signaling, vehicle management systems, medical devices). This trend is likely to continue, as applications that would have been unthinkable only a few short years ago come into the reach of ever more complex processors. Many such applications are long lived, interact with their environment continuously, and are under important real-time constraints. As these reactive systems permeate our lives, bringing us everything from intelligent pace-makers to tiny freshness-tracking devices in groceries, the need for cost-effective, confidence-inspiring software engineering techniques and tools grows proportionately.

While powerful processors are increasingly being used for small tasks, the demand for cost-effective computation is also forcing the use of smaller, resource-constrained devices in even greater numbers. For developers, this means that the use of large, complex, "fool-proof" real-time operating systems is often not an option. Much of the software in reactive and embedded systems is programmed from scratch, in low level languages like C and assembly,

with hardware interrupt handling and no high-level process model. At present, the only way to evaluate such systems for safety and correctness is through extensive testing or simulation. Such component and integration testing is intensely time-consuming and expensive, but often less than comprehensive. Furthermore, in a time when rapid market cycles in all areas of electronics and communications virtually guarantee design changes in any meaningfully complex project, even the slightest change can invalidate months of comprehensive testing.

A key difficulty in building reactive real-time systems is the complicated timing constraints that such systems must meet. For example, the avionic controls of a high performance fighter jet must be able to process input from the pilot and dozens of instruments several hundred times per second in order to react properly. The disparate tasks of such a reactive system may have complex deadlines and intricate interactions, making it difficult to know whether a given system design can meet all of its expected deadlines. Once the system is actually built, it is still difficult to tell whether or not it satisfies the designer's timing constraints under all normal operating conditions.

While it is unlikely that the need for full-scale testing will ever be completely supplanted by any other methodology, there is great potential for tools to substantially decrease the cost of building and testing reactive systems, both in time and effort. For example, static analysis of timing properties in reactive systems could eliminate whole classes of errors prior to testing. Much work has gone into the theoretical foundations of designing and verifying real-time systems, and several model checkers for real-time logics are now available. At the end of the rainbow in this research area lies *correct by construction* technology, which would allow software to be proven "correct" in a wide range of aspects before actual testing even begins.

Our goal is to design and implement tools for building reactive real-time systems that go beyond "valid via testing", toward "correct by construction" technology.

Such tools may lead to faster software development and fewer surprises after deployment.

We currently focus on the challenges faced by compilers for embedded software. For embedded systems, predictability and resource awareness are of greater importance than execution efficiency. For example, if a compiler is not aware of the size of the instruction store for generated code, opportunities to squeeze code into the available space can be lost. A programmer can often hand-optimize generated code, but this is tedious and error-prone, and does not scale to large systems. This project is aimed towards a new generation of resource-aware compilers with solid foundations. Next-generation compilers will lead to increased confidence in a wide variety of embedded systems, including sensor networks, medical implants, engine control, and fly-by-wire/drive-by-wire systems.

We are investigating resource-awareness in several dimensions, including language design, type systems, static analysis, model checking, code generation, and reverse engineering. We are focusing on real-time, interrupt-driven systems and network processors. The resources that we currently consider are code size, stack size, real-time deadlines, bus width, and network-processor packet engines.

Palsberg's work in this area is supported by an NSF ITR award and by Intel.

Among Palsberg's research accomplishments in this area are:

✓ A tool for static checking of interrupt-driven assembly code [63].

- ✓ A code-size-aware compiler that uses integer constraints for register allocation and instruction selection [41].
- ✓ A typed interrupt calculus, a proof of stack boundedness, a prototype implementation of a type checker, and an algorithm for type inference [65, 43].
- ✓ An almost-automatic reverse-engineering tool for real-time Z86 assembly code [98].

Type Inference and Flow Analysis for Object-Oriented Languages

Palsberg is developing type inference algorithms and static analyses for object-oriented software which are practically useful for tool developers and language designers. Abadi (Digital Systems Research Center) and Cardelli (Microsoft Research) wrote in their book, with reference to Palsberg's paper [25]:

“Thus the absence of minimum typings poses practical problems for type inference. Palsberg has described an ingenious algorithm for type inference that surmounts these problems.” [On p.97 of “A Theory of Objects”, by Martín Abadi and Luca Cardelli, Springer-Verlag, New York, 1996.]

An example application is tool support for transforming a dynamically-typed prototype application into a statically-typed product, say, from Smalltalk to Java. Palsberg is also investigating type-based analysis which integrates types and flow information. In simple settings, the known benefits of such an integration include: easy correctness proofs, faster flow analysis without sacrificing precision, a definition of a both sound and complete flow analysis, and a simplified compiler structure. His goal is to obtain these benefits for object-oriented languages. Aiken (University of California, Berkeley) and Heintze (Bell Laboratories, Lucent Technologies) wrote in their tutorial notes a section about a commonly used flow analysis of object-oriented programs:

“This analysis was discovered by Palsberg and Schwartzbach . . . very efficient in practice.” [On p.16,27 of the tutorial notes by Alex Aiken and Nevin Heintze, distributed at their 1995 tutorial on constraint-based program analysis given at the ACM Symposium on Principles of Programming Languages, available from <http://www.cs.berkeley.edu/~aiken>]

Palsberg's work in this area has been supported by an NSF CAREER award.

Among Palsberg's research accomplishments in this area are:

- ✓ The first polynomial-time type inference algorithm for an object-oriented language with subtyping and recursive types [25].
- ✓ The first polynomial-time type inference algorithm for an object-oriented language with record concatenation and subtyping [40].
- ✓ The first control-flow analysis algorithm for an object-oriented language with dynamic and multiple inheritance [23].
- ✓ The first polynomial-time type inference algorithm for a type system with finite types and non-structural subtyping [15].
- ✓ The first polynomial-time algorithm for deciding the subtype relation between recursive types [17].
- ✓ The first formal relationship between a type system and a control-flow analysis [22].

- ✓ The first formal relationship between traditional types and constrained types [27].
- ✓ The first type inference algorithm that handles simple thistypes (also known as self-types) [31].
- ✓ The first proof that 0-CFA style flow analysis is sound with respect to arbitrary beta-reduction [20].
- ✓ A denotational semantics of inheritance and a proof that it is equivalent to the standard method lookup algorithm of Smalltalk [13].

Software Evolution

Palsberg is developing techniques for supporting software evolution. The work is proceeding in three different directions: design and implementation of new language constructs based on Lieberherr's notion of adaptive programming; tool support for the use of design patterns in software systems; and experiments with large-scale genetic programming. For example, Palsberg has with Kevin Tao developed the Java Tree Builder, which is a freely available frontend for The Java Compiler Compiler from Sun Microsystems. It supports interaction with syntax trees using the Visitor design pattern. Sankar (formerly Sun Microsystems, now President and CEO of Metamata), the principal developer of the Java Compiler Compiler, wrote in an open letter:

“JTB has clearly shown its impact on the user community.” [Sriram Sankar, open letter, May 14, 1998, <http://www.cs.purdue.edu/homes/taokr/jtb/sankar.txt>.]

Palsberg's work on software evolution has been supported by DARPA and British Telecom.

Among Palsberg's research accomplishments in this area are:

- ✓ An efficient implementation of adaptive software [21, 32].
- ✓ An extension of object-oriented languages which makes it easy to program many common design patterns [34].

Information Security

Palsberg is developing techniques for statically enforcing security policies and for protecting the intellectual property contained in software. Palsberg's work on information security is supported by CERIAS, Center for Education and Research in Information Assurance and Security at Purdue University.

Among his research accomplishments in this area is:

- ✓ A framework of trusted types for detection of security holes in software [33].

Trusted types enable a trust analysis which ensures that untrusted data cannot reach an operation which assumes that the data is trusted. For example, a HTTP server handles requests to execute CGI scripts, and these requests should initially be treated as untrusted. The requested script should only be run if it can be verified that the requesting browser has adequate authorization. Palsberg and Orbaek's trust analysis [33] ensures that checks are performed on all data paths from the reception of the URL request to the command that executes the requested script.

One of the main conferences in the area of programming languages is the ACM Symposium on Principles of Programming Languages. At the 1998 meeting, there were three papers

on information security, and all of them referenced Palsberg’s paper with Orbaek [33]. For example, Heintze and Riecke (Bell Laboratories, Lucent Technologies) wrote in their paper:

“Type systems have been also used for the related problem of reasoning about trustworthiness of data. For instance, [Palsberg’s paper with Oerbaek] introduces a calculus in which one can explicitly annotate expressions as trusted or distrusted and check their trust/distrust status; this system enforces consistent use of these annotations.” [On p.373 of “The SLam Calculus: Programming with Secrecy and Integrity”, by Nevin Heintze and Jon Riecke, in Proceedings of POPL’98, 25th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, pp.365–377, ACM, New York.]

✓ A software watermarking system for Java [62].

Software Watermarking is an approach to software ownership protection. There are at least four U.S. patents on software watermarking, and an idea for further advancing the state of the art was presented in 1999 by Collberg and Thomborsen. The new idea is to embed a watermark in dynamic data structures, thereby protecting against many program-transformation attacks. We have implemented and experimented with a watermarking system for Java based on the ideas of Collberg and Thomborsen.

Partial Evaluation

Palsberg is interested in making partial evaluation more easily applicable. It is well known that source-program modifications can make a partial evaluator yield dramatically better results. For example, eta-redexes can preserve static data flow by acting as an interface between values and contexts. Together with Danvy and Malmkjær, Palsberg has presented a type-based explanation of what eta-expansion achieves, why it works, and how it can be automated. This leads to a unified view of various source-code improvements, including a popular transformation called “The Trick.”

Among Palsberg’s research accomplishments in this area are:

✓ The first framework and algorithm for improving partial evaluators by the use of eta-redexes [19, 28].

✓ An automatically generated compiler generator, obtained by partial evaluation of an interpreter for the semantic metalanguage [26].

✓ A proof of correctness for flow-based binding-time analysis [12].

In 1998, Palsberg gave lectures at the Summer School on *Partial Evaluation: Practice and Theory*, Copenhagen, Denmark.

2.2 Publications

Books

- [1] “Object-oriented type systems”, with Michael I. Schwartzbach. John Wiley & Sons, 1994. ISBN 0 471 94128 X. 180 pp.
- [2] “Modern compiler implementation in Java”. Cambridge University Press, 2002. Main author: Andrew W. Appel; co-author: Jens Palsberg.

Editor of Conference Proceedings

- [3] “Static analysis”, Springer-Verlag (*LNCS* 1824), 2000. Proceedings of SAS’00, 7th International Static Analysis Symposium.
- [4] “Program analysis for software tools and engineering”, with Matthew B. Dwyer. ACM Press, 2002. Proceedings of PASTE’02, ACM SIGPLAN-SIGSOFT Workshop.
- [5] “Proceedings of POPL’05, ACM SIGPLAN-SIGACT symposium on principles of programming languages”. ACM Press, 2005. General chair: Jens Palsberg. Program chair: Martín Abadi.
- [6] with Holger Hermanns. Springer-Verlag, 2006. Proceedings of TACAS’06, International Conference on Tools and Algorithms for the Construction and Analysis of Systems.
- [7] with James Hoe. IEEE Press, 2006. Proceedings of MEMOCODE’06, ACM-IEEE Conference on Formal Methods and Programming Models for Co-Design.
- [8] with Klaus Havelund and Rupak Majumdar. Springer-Verlag, 2008. Proceedings of SPIN’08, Model Checking Software, 15th International SPIN Workshop.
- [9] with Luca de Alfaro. ACM Press, 2008. Proceedings of EMSOFT’08, International Conference on Embedded Software.

Refereed Journal Articles

- [10] “Safety analysis versus type inference for partial types”, with Michael I. Schwartzbach. *Information Processing Letters*, 43:175–180, 1992.
- [11] “Normal forms have partial types”. *Information Processing Letters*, 45:1–3, 1993.
- [12] “Correctness of binding-time analysis”. *Journal of Functional Programming*, 3(3):347–363, 1993.
- [13] “A denotational semantics of inheritance and its correctness”, with William Cook. *Information and Computation*, 114(2):329–350, 1994.
[Preliminary version in Proceedings of OOPSLA’89, ACM SIGPLAN Fourth Annual Conference on Object-Oriented Programming Systems, Languages and Applications, pages 433–443, New Orleans, Louisiana, October 1989].
- [14] “Static typing for object-oriented programming”, with Michael I. Schwartzbach. *Science of Computer Programming*, 23(1):19–53, 1994.
- [15] “Efficient inference of partial types”, with Dexter Kozen and Michael I. Schwartzbach. *Journal of Computer and System Sciences*, 49(2):306–324, 1994.
[Preliminary version in Proceedings of FOCS’92, 33rd IEEE Symposium on Foundations of Computer Science, pages 363–371, Pittsburgh, Pennsylvania, October 1992].

- [16] “Safety analysis versus type inference”, with Michael I. Schwartzbach. *Information and Computation*, 118(1):128–141, 1995.
- [17] “Efficient recursive subtyping”, with Dexter Kozen and Michael I. Schwartzbach. *Mathematical Structures in Computer Science*, 5(1):113–125, 1995.
[Preliminary version in Proceedings of POPL’93, Twentieth Annual SIGPLAN–SIGACT Symposium on Principles of Programming Languages, pages 419–428, Charleston, South Carolina, January 1993].
- [18] “Complexity results for 1-safe nets”, with Allan Cheng and Javier Esparza. *Theoretical Computer Science*, 147(1–2):117–136, 1995.
[Preliminary version in Proceedings of FST&TCS 13, Thirteenth Conference on the Foundations of Software Technology & Theoretical Computer Science, Springer-Verlag (LNCS 761), pages 326–337, Bombay, India, December 1993].
- [19] “The essence of eta-expansion in partial evaluation”, with Olivier Danvy and Karoline Malmkjær. *Lisp and Symbolic Computation*, 8(3):209–227, 1995.
[Preliminary version in Proceedings of PEPM’94, ACM SIGPLAN Workshop on Partial Evaluation and Semantics-Based Program Manipulation, pages 11–20, Orlando, Florida, June 1994].
- [20] “Closure analysis in constraint form”. *ACM Transactions on Programming Languages and Systems*, 17(1):47–62, January 1995.
[Preliminary version in Proceedings of CAAP’94, Colloquium on Trees in Algebra and Programming, Springer-Verlag (LNCS 787), pages 276–290, Edinburgh, Scotland, April 1994].
- [21] “Efficient implementation of adaptive software”, with Cun Xiao and Karl Lieberherr. *ACM Transactions on Programming Languages and Systems*, 17(2):264–292, March 1995.
- [22] “A type system equivalent to flow analysis”, with Patrick M. O’Keefe. *ACM Transactions on Programming Languages and Systems*, 17(4):576–599, July 1995.
[Preliminary version in Proceedings of POPL’95, 22nd Annual SIGPLAN–SIGACT Symposium on Principles of Programming Languages, pages 367–378, San Francisco, California, January 1995].
- [23] “Type inference of Self: Analysis of objects with dynamic and multiple inheritance”, with Ole Agesen and Michael I. Schwartzbach. *Software – Practice & Experience*, 25(9):975–995, September 1995.
[Preliminary version in Proceedings of ECOOP’93, Seventh European Conference on Object-Oriented Programming, Springer-Verlag (LNCS 707), pages 247–267, Kaiserslautern, Germany, July 1993].
- [24] “Strong normalization with non-structural subtyping”, with Mitchell Wand and Patrick M. O’Keefe. *Mathematical Structures in Computer Science*, 5(3):419–430, 1995.

- [25] “Efficient inference of object types”. *Information and Computation*, 123(2):198–209, 1995.
[Preliminary version in Proceedings of LICS’94, Ninth Annual IEEE Symposium on Logic in Computer Science, pages 186–195, Paris, France, July 1994].
- [26] “Generating action compilers by partial evaluation”, with Anders Bondorf. *Journal of Functional Programming*, 6(2):269–298, 1996.
[Preliminary version in Proceedings of FPCA’93, Sixth ACM Conference on Functional Programming Languages and Computer Architecture, pages 308–317, Copenhagen, Denmark, June 1993].
- [27] “Constrained types and their expressiveness”, with Scott Smith. *ACM Transactions on Programming Languages and Systems*, 18(5):519–527, September 1996.
- [28] “Eta-expansion does the Trick”, with Olivier Danvy and Karoline Malmkjær. *ACM Transactions on Programming Languages and Systems*, 18(6):730–751, November 1996.
- [29] “Type inference with non-structural subtyping”, with Mitchell Wand and Patrick M. O’Keefe. *Formal Aspects of Computing*, 9:49–67, 1997.
- [30] “Class-graph inference for adaptive programs”. *Theory and Practice of Object Systems*, 3(2):75–85, 1997.
- [31] “Type inference with simple selftypes is NP-complete”, with Trevor Jim. *Nordic Journal of Computing*, 4(3):259–286, Fall 1997.
- [32] “A new approach to compiling adaptive programs”, with Boaz Patt-Shamir and Karl Lieberherr. *Science of Computer Programming*, 29(3):303–326, September 1997.
[Preliminary version in Proceedings of ESOP’96, European Symposium on Programming, Springer-Verlag (LNCS 1058), pages 280–295, Linköping, Sweden, April 1996.].
- [33] “Trust in the λ -calculus”, with Peter Ørbæk. *Journal of Functional Programming*, 7(6):557–591, November 1997.
[Preliminary version in Proceedings of SAS’95, International Static Analysis Symposium, Springer-Verlag (LNCS 983), pages 314–330, Glasgow, Scotland, September 1995].
- [34] “Evolution of object behavior using context relations”, with Linda Seiter and Karl Lieberherr. *IEEE Transactions on Software Engineering*, 24(1):79–92, 1998.
[Preliminary version in Proceedings of ACM FSE’96, Fourth Symposium on the Foundations of Software Engineering, pages 46–57, San Francisco, California, October 1996].
- [35] “Equality-based flow analysis versus recursive types”. *ACM Transactions on Programming Languages and Systems*, 20(6):1251–1264, 1998.

- [36] “Optimal representations of polymorphic types with subtyping”, with Alexander Aiken and Edward L. Wimmers. *Higher-Order and Symbolic Computation*, 12(3):1–46, October 1999.
[Preliminary version in Proceedings of TACS’97, International Symposium on Theoretical Aspects of Computer Software, Springer-Verlag (*LNCS* 1281), pages 47–76, Sendai, Japan, September 1997].
- [37] “From polyvariant flow information to intersection and union types”, with Christina Pavlopoulou. *Journal of Functional Programming*, 11(3):263–317, May 2001.
[Preliminary version in Proceedings of POPL’98, 25th Annual SIGPLAN–SIGACT Symposium on Principles of Programming Languages, pages 197–208, San Diego, California, January 1998].
- [38] “Efficient and flexible matching of recursive types”, with Tian Zhao. *Information and Computation*, 171:364–387, 2001.
[Preliminary version in Proceedings of LICS’00, Fifteenth Annual IEEE Symposium on Logic in Computer Science, pages 388–398, Santa Barbara, California, June 2000].
- [39] “CPS transformation of flow information”, with Mitchell Wand. *Journal of Functional Programming*, 13(5):905–923, 2003.
- [40] “Type inference for record concatenation and subtyping”, with Tian Zhao. *Information and Computation*, 189:54–86, 2004.
[Preliminary version in Proceedings of LICS’02, IEEE Symposium on Logic in Computer Science, pages 125–136, Copenhagen, Denmark, July 2002].
- [41] “Compiling with code-size constraints”, with Mayur Naik. *ACM Transactions on Embedded Computing Systems*, 3(1):163–181, 2004.
[Preliminary version in Proceedings of LCTES’02, Languages, Compilers, and Tools for Embedded Systems joint with SCOPES’02, Software and Compilers for Embedded Systems, pages 120–129, Berlin, Germany, June 2002].
- [42] “Type-safe method inlining”, with Neal Glew. *Science of Computer Programming*, 52:281–306, 2004.
[Preliminary version in Proceedings of ECOOP’02, European Conference on Object-Oriented Programming, pages 525–544, Springer-Verlag (*LNCS* 2374), Malaga, Spain, June 2002].
- [43] “Stack size analysis of interrupt driven software”, with Krishnendu Chatterjee, Di Ma, Rupak Majumdar and Tian Zhao and Thomas A. Henzinger. *Information and Computation*, 194(2):144–174, 2004. Special issue dedicated to Paris Kanellakis.
[Preliminary version in Proceedings of SAS’03, International Static Analysis Symposium, Springer-Verlag (*LNCS* 2694), pages 109–126, San Diego, June 2003].
- [44] “Deadline analysis of interrupt-driven software”, with Dennis Brylow. *IEEE Transactions on Software Engineering*, 30(10):634–655, 2004.

- [Preliminary version in Proceedings of FSE'03, ACM SIGSOFT International Symposium on the Foundations of Software Engineering joint with ESEC'03, European Software Engineering Conference, 198–207, Helsinki, Finland, September, 2003].
- [45] “Automatic discovery of covariant read-only fields”, with Tian Zhao and Trevor Jim. *ACM Transactions on Programming Languages and Systems*, 27(1):126–162, January 2005.
[Preliminary version in Informal Proceedings of FOOL'02, Ninth International Workshop on Foundations of Object-Oriented Languages, Portland, Oregon, 2002].
- [46] “Method inlining, dynamic class loading, and type soundness”, with Neal Glew. *Journal of Object Technology*, 4(8):33–53, 2005.
[Preliminary version in Sixth Workshop on Formal Techniques for Java-like Programs, Oslo, Norway, June 2004].
- [47] “Type-based confinement”, with Tian Zhao and Jan Vitek. *Journal of Functional Programming*, 16(1):83–128, 2006.
[Preliminary version, entitled “Lightweight confinement for Featherweight Java”, in Proceedings of OOPSLA'03, ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages and Applications, pages 135–148, Anaheim, California, October 2003].
- [48] “Encapsulating objects with confined types”, with Christian Grothoff and Jan Vitek. *ACM Transactions on Programming Languages and Systems*, 29(6), 2007.
[Preliminary version in Proceedings of OOPSLA'01, ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages and Applications, pages 241–253, Tampa Bay, Florida, October 2001].
- [49] “A type system equivalent to a model checker”, with Mayur Naik. *ACM Transactions on Programming Languages and Systems*, 30(5), 2008.
[Preliminary version in Proceedings of ESOP'05, European Symposium on Programming, pages 374–388, Edinburgh, Scotland, April 2005].
- [50] “Aliased register allocation for straight-line programs is NP-complete”, with Jonathan K. Lee and Fernando Magno Quintão Pereira. *Theoretical Computer Science*, 407:258–273, 2008.
[Preliminary version in Proceedings of ICALP'07, 34th International Colloquium on Automata, Languages and Programming, pages 680–691, Wroclaw, Poland, July 2007].

Refereed Conference Papers

The following papers are *not* preliminary versions of the journal articles listed above.

- [51] “Type substitution for object-oriented programming”, with Michael I. Schwartzbach. In *Proceedings of OOPSLA/ECOOP'90, ACM SIGPLAN Fifth Annual Conference on*

- Object-Oriented Programming Systems, Languages and Applications; European Conference on Object-Oriented Programming*, pages 151–160, Ottawa, Canada, October 1990.
- [52] “What is type-safe code reuse?”, with Michael I. Schwartzbach. In *Proceedings of ECOOP’91, Fifth European Conference on Object-Oriented Programming*, pages 325–341. Springer-Verlag (LNCS 512), Geneva, Switzerland, July 1991.
- [53] “Object-oriented type inference”, with Michael I. Schwartzbach. In *Proceedings of OOPSLA ’91, ACM SIGPLAN Sixth Annual Conference on Object-Oriented Programming Systems, Languages and Applications*, pages 146–161, Phoenix, Arizona, October 1991.
- [54] “A provably correct compiler generator”. In *Proceedings of ESOP’92, European Symposium on Programming*, pages 418–434. Springer-Verlag (LNCS 582), Rennes, France, February 1992.
- [55] “An automatically generated and provably correct compiler for a subset of Ada”. In *Proceedings of ICCL’92, Fourth IEEE International Conference on Computer Languages*, pages 117–126, Oakland, California, April 1992.
- [56] “Making type inference practical”, with Nicholas Oxhøj and Michael I. Schwartzbach. In *Proceedings of ECOOP’92, Sixth European Conference on Object-Oriented Programming*, pages 329–349. Springer-Verlag (LNCS 615), Utrecht, The Netherlands, July 1992.
- [57] “Binding-time analysis: Abstract interpretation versus type inference”, with Michael I. Schwartzbach. In *Proceedings of ICCL’94, Fifth IEEE International Conference on Computer Languages*, pages 289–298, Toulouse, France, May 1994.
- [58] “Comparing flow-based binding-time analyses”. In *Proceedings of TAPSOFT’95, Theory and Practice of Software Development*, pages 561–574. Springer-Verlag (LNCS 915), Aarhus, Denmark, May 1995.
- [59] “The essence of the visitor pattern”, with C. Barry Jay. In *Proceedings of COMP-SAC’98, 22nd Annual International Computer Software and Applications Conference*, pages 9–15, Vienna, Austria, August 1998.
- [60] “Program optimization for faster genetic programming”, with Bradley Lucier and Sudhakar Mamillapalli. In *Proceedings of GP’98, Genetic Programming*, pages 202–207, Madison, Wisconsin, July 1998.
- [61] “Scalable propagation-based call graph construction algorithms”, with Frank Tip. In *Proceedings of OOPSLA’00, ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages and Applications*, pages 281–293, Minneapolis, Minnesota, October 2000.

- [62] “Experience with software watermarking”, with Sowmya Krishnaswamy, Minseok Kwon, Di Ma, Qiuyun Shao, and Yi Zhang. In *Proceedings of ACSAC’00, 16th Annual Computer Security Applications Conference*, pages 308–316, New Orleans, Louisiana, December 2000.
- [63] “Static checking of interrupt-driven software”, with Dennis Brylow and Niels Damgaard. In *Proceedings of ICSE’01, 23rd International Conference on Software Engineering*, pages 47–56, Toronto, May 2001.
- [64] “Efficient type matching”, with Somesh Jha and Tian Zhao. In *Proceedings of FOS-SACS’02, Foundations of Software Science and Computation Structures*, pages 187–204. Springer-Verlag (LNCS 2303), Grenoble, France, April 2002. An expanded version of the paper, co-authored with Fritz Henglein, is to appear in the book *Automatic Program Development: a Tribute to Robert Paige*.
- [65] “A typed interrupt calculus”, with Di Ma. In *Proceedings of FTRTFT’02, 7th International Symposium on Formal Techniques in Real-Time and Fault Tolerant Systems*, pages 291–310. Springer-Verlag (LNCS 2469), Oldenburg, Germany, September 2002.
- [66] “Efficient spill code for SDRAM”, with V. Krishna Nandivada. In *Proceedings of CASES’03, International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, pages 24–31, San Jose, California, October 2003.
- [67] “Timing analysis of TCP servers for surviving denial-of-service attacks”, with V. Krishna Nandivada. In *Proceedings of RTAS’05, 11th IEEE Real-Time and Embedded Technology and Applications Symposium*, pages 541–549, San Francisco, March 2005.
- [68] “Avrora: Scalable sensor network simulation with precise timing”, with Ben L. Titzer and Daniel K. Lee. In *Proceedings of IPSN’05, Fourth International Conference on Information Processing in Sensor Networks*, pages 477–482, Los Angeles, April 2005.
- [69] “Nonintrusive precision instrumentation of microcontroller software”, with Ben L. Titzer. In *Proceedings of LCTES’05, Conference on Languages, Compilers and Tools for Embedded Systems*, pages 59–68, Chicago, Illinois, June 2005.
- [70] “Type-safe optimisation of plugin architectures”, with Neal Glew and Christian Grothoff. In *Proceedings of SAS’05, Static Analysis Symposium*, pages 135–154. Springer-Verlag (LNCS 3672), London, UK, September 2005.
- [71] “Register allocation via coloring of chordal graphs”, with Fernando Magno Quintão Pereira. In *Proceedings of APLAS’05, Asian Symposium on Programming Languages and Systems*, pages 315–329. Springer-Verlag (LNCS 3780), Tsukuba, Japan, November 2005.
- [72] “Register allocation after classical SSA elimination is NP-complete”, with Fernando Magno Quintão Pereira. In *Proceedings of FOSSACS’06, Foundations of Software Science and Computation Structures*, pages 79–93. Springer-Verlag (LNCS 3921), Vienna, Austria, March 2006.

- [73] “Sara: Combining stack allocation and register allocation”, with V. Krishna Nandivada. In *Proceedings of CC’06, International Conference on Compiler Construction*, pages 232–246. Springer-Verlag (LNCS 3923), Vienna, Austria, March 2006.
- [74] “Inference of user-defined type qualifiers and qualifier rules”, with Brian Chin and Shane Markstrum and Todd Millstein. In *Proceedings of ESOP’06, European Symposium on Programming*, pages 264–278. Springer-Verlag (LNCS 3924), Vienna, Austria, March 2006.
- [75] “The ExoVM system for automatic VM and application reduction”, with Ben L. Titzer and Joshua Auerbach and David F. Bacon. In *Proceedings of PLDI’07, ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 352–362, San Diego, California, June 2007.
- [76] “A framework for end-to-end verification and evaluation of register allocators”, with V. Krishna Nandivada and Fernando Magno Quintão Pereira. In *Proceedings of SAS’07, International Static Analysis Symposium*, pages 153–169, Kongens Lyngby, Denmark, August 2007.
- [77] “Vertical object layout and compression for fixed heaps”, with Ben L. Titzer. In *Proceedings of CASES’07, International Conference on Compilers, Architecture, and Synthesis for Embedded Systems*, pages 170–178, Salzburg, Austria, September 2007.
- [78] “Register allocation by puzzle solving”, with Fernando Magno Quintão Pereira. In *Proceedings of PLDI’08, ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 216–226, Tucson, Arizona, June 2008.
- [79] “Constrained types for object-oriented languages”, with Vijay Saraswat and Nathaniel Nystrom, and Christian Grothoff. In *Proceedings of OOPSLA’08, ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages and Applications*, Nashville, Tennessee, October 2008.
- [80] “SSA elimination after register allocation”, with Fernando Magno Quintão Pereira. In *Proceedings of CC’09, International Conference on Compiler Construction*. Springer-Verlag (LNCS), York, UK, March 2009. To appear.
- [81] “Featherweight X10: a core calculus for async-finish parallelism”, with Jonathan K. Lee. In *Proceedings of PPOPP’10, 15th ACM SIGPLAN Annual Symposium on Principles and Practice of Parallel Programming*, Bangalore, India, January 2010.

Refereed Survey Papers

- [82] “Type inference for objects”. *ACM Computing Surveys*, 28(2):358–359, June 1996.
- [83] “Strategic directions for research on programming languages”, with Chris Hankin and Hanne Riis Nielson. *ACM Computing Surveys*, 28(4):644–652, December 1996.

- [84] “Compiler technology for object-oriented languages”. *ACM Computing Surveys*, 28A(4), December 1996. www.acm.org/pubs/citations/journals/surveys/1996-28-4es/a161-palsberg/.
- [85] “Software evolution and integration”. *ACM Computing Surveys*, 28A(4), December 1996. www.acm.org/pubs/citations/journals/surveys/1996-28-4es/a200-palsberg/.

Other Papers

- [86] “Three discussions on object-oriented typing”, with Michael I. Schwartzbach. *ACM SIGPLAN OOPS Messenger*, 3(2):31–38, 1992.
- [87] “Foundations of object-oriented languages”, with Andrew Black. *ACM SIGPLAN Notices*, 29(3):3–11, 1994.
- [88] “Eta-redexes in partial evaluation”. In John Hatcliff, Torben Æ. Mogensen, and Peter Thiemann, editors, *Partial Evaluation: Practice and Theory*, pages 356–366. Springer-Verlag, 1999.
- [89] “Type-based analysis and applications”. In *Proceedings of PASTE’01, ACM SIGPLAN/SIGSOFT Workshop on Program Analysis for Software Tools and Engineering*, pages 20–27, Snowbird, Utah, June 2001. Invited paper.
- [90] “Teaching reviewing to graduate students”, with Scott J. Baxter. *Communications of the ACM*, 45(12):22–24, December 2002.
- [91] “ILP-based resource-aware compilation”, with Mayur Naik. In Ahmed Jerraya and Wayne Wolf, editors, *Multiprocessor Systems-on-Chips*, chapter 12, pages 337–354. Elsevier, 2004.
- [92] “Programming languages”. In Richard C. Dorf, editor, *The Engineering Handbook*, chapter 145. CRC Press, 2004.
- [93] “Enabling detailed modeling and analysis of sensor networks”, with Olaf Landsiedel and Klaus Wehrle and Ben L. Titzer. *Praxis der Informationsverarbeitung und Kommunikation*, 28(2):10–15, 2005.
- [94] “Type systems: Advances and applications”, with Todd Millstein. In Y. N. Srikant and Priti Shankar, editors, *The Compiler Design Handbook*, chapter 9. CRC Press, 2007.

Unpublished Papers

- [95] “Layout construction: A case study in algorithm engineering”, with Gudmund Skovbjerg Frandsen, Erik Meineche Schmidt, and Steen Sjøgaard. Technical Report DAIMI PB-450, Computer Science Department, Aarhus University, August 1993.
- [96] “Type inference in systems of recursive types with subtyping”, with Trevor Jim. Manuscript, 1997.

- [97] “Reducing loads and stores in stack architectures”, with Thomas VanDrunen and Antony L. Hosking. Manuscript, 2000.
- [98] “Reverse engineering of real-time assembly code”, with Matthew Wallace. Manuscript, 2002.
- [99] “Visitor-oriented programming”, with Thomas VanDrunen. In *Informal Proceedings of FOOL’04, Eleventh International Workshop on Foundations of Object-Oriented Languages*, Venice, Italy, January 2004.
- [100] “Flow analysis of an intermediate language for object-oriented languages”, with Neal Glew and Ben L. Titzer. Manuscript, 2005.
- [101] “Automatic generation of flexible simulators that support updatable code”, with Ben L. Titzer and Jonathan K. Lee. Submitted for publication, 2006.

Papers Submitted for Publication

- [102] “Correctness of ILP-based register allocation”, with Mayur Naik. Manuscript, 2004.
- [103] “Efficient type-2 puzzle solving”, with Siddharth Tiwary. Submitted for publication, 2009.

2.3 Patents and Software Systems

- [104] “Automata-theoretic approach compiler for adaptive software”, with Karl Lieberherr and Boaz Patt-Shamir. US Patent 5,946,490, August 1999.
- [105] “Scalable propagation-based methods for call graph construction”, with Frank Tip. US Patent 7,003,507, February 2006.

The patent with Frank Tip is on an algorithm for static analysis of object-oriented software. It is incorporated into an IBM product, WebSphere Studio Device Developer (WSDD). WSDD has a link-time optimizer component called SmartLinker which embodies our algorithm.

“The Java Tree Builder”, a freely available frontend for The Java Compiler Compiler from Sun Microsystems, with Kevin Tao, www.cs.purdue.edu/homes/taokr/jtb/index.html. It supports interaction with syntax trees using the Visitor design pattern. A survey of user comments is available from the webpage; they received many enthusiastic comments from users both in industry and in academia. See Section 2.1 for more information.

2.4 Talks, Tutorials, and Summer School Lectures

The talks listed here are not mentioned otherwise.

Invited talks at international meetings

- The Newton Institute Euroconference on *Advances in Type Systems for Computing*, Newton Institute, Cambridge, UK (Aug 1995).
- The workshop *Logic and Semantics of Programming Languages* organized by the ESPRIT project *Categorical Logic in Computer Science*, Birmingham, UK (Sep 1996).
- PASTE, ACM SIGPLAN–SIGSOFT Workshop on *Program Analysis for Software Tools and Engineering*, Snowbird, Utah (Jun 2001).
- MFPS, Eighteenth Workshop on the *Mathematical Foundations of Programming Semantics*, New Orleans (Mar 2002).
- CASES, International Conference on *Compilers, Architectures and Synthesis for Embedded Systems* in a joint session with EMSOFT, *Workshop on Embedded Software*, Grenoble, France (Oct 2002).
- SBLP, *Brazilian Symposium on Programming Languages*, Itatiaia, Rio de Janeiro, Brazil (May 2006).
- APLAS, *Fourth Asian Symposium on Programming Languages and Systems*, Sydney, Australia (Nov 2006).
- CATS, *Computing: The Australasian Theory Symposium*, Ballarat, Australia (Jan 2007). Keynote speaker.
- VMCAI, *International Conference on Verification, Model Checking and Abstract Interpretation*, San Francisco (Jan 2008). Invited 2-hour tutorial on “verification of register allocators.”
- ICESS, *The 6th IEEE International Conference on Embedded Software and Systems*, HangZhou, Zhejiang, China (May 2009).

Other talks at international meetings

- The meeting of the ESPRIT project *Semantique*, Mont St. Michel, France (Oct 1991).
- The ESPRIT/NSF sponsored workshop *Foundations of Object-Oriented Languages*, Stanford University, California (Oct 1993).
- The ESPRIT/NSF sponsored workshop *Foundations of Object-Oriented Languages*, Paris, France (Jun 1994).
- The 6th *Nordic Workshop on Programming Theory*, Aarhus, Denmark (Oct 1994).
- The Dagstuhl-Seminar on *Object-Orientation with Parallelism and Persistence*, Schloss Dagstuhl, Germany (Apr 1995).

- The Dagstuhl-Seminar on *Abstract Interpretation*, Schloss Dagstuhl, Germany (Aug 1995).
- The *Fifth Workshop on Specification of Behavioral Semantics* held in conjunction with OOPSLA'96, ACM SIGPLAN 11th Annual Conference on Object-Oriented Programming Systems, Languages and Applications, San Jose, California (Oct 1996).
- The Dagstuhl-Seminar on *Applications of Tree Automata in Rewriting, Logic and Programming*, Schloss Dagstuhl, Germany (Oct 1997).
- The 2nd Workshop on *Distributed Object Security* held in conjunction with OOPSLA'99, ACM SIGPLAN 14th Annual Conference on Object-Oriented Programming Systems, Languages and Applications, Denver, Colorado (Nov 1999).
- The Dagstuhl-Seminar on *Effective Implementation of Object-Oriented Programming Languages*, Schloss Dagstuhl, Germany (Nov 2000).
- The University of Washington/Microsoft Research Summer Workshop on *Specifying and Checking Properties of Software*, Sleeping Lady, Washington (Aug 2001).
- NSF/SIGDA Embedded Systems Workshop, Atlanta (Nov 2001).

Tutorials and summer school lectures

- “Types for the language designer”, with Michael I. Schwartzbach. Half-day tutorial (75 participants) given at OOPSLA'92, ACM SIGPLAN Seventh Annual Conference on Object-Oriented Programming Systems, Languages and Applications, Vancouver, Canada, October 1992.
- “Object-oriented type systems”, with Michael I. Schwartzbach. Half-day tutorial (16 participants) given at ECOOP'93, European Conference on Object-Oriented Programming, Kaiserslautern, Germany, July 1993.
- “Types for the language designer”, with Michael I. Schwartzbach. Half-day tutorial (47 participants) given at OOPSLA'93, ACM SIGPLAN Eighth Annual Conference on Object-Oriented Programming Systems, Languages and Applications, Washington D.C., USA, October 1993.
- “Types for the language designer”, with Michael I. Schwartzbach. Half-day tutorial (48 participants) given at OOPSLA'95, ACM SIGPLAN Tenth Annual Conference on Object-Oriented Programming Systems, Languages and Applications, Austin, Texas, October 1995.
- “Type inference for objects”. A series of lectures (48 participants) given at the ACM State of the Art Summer School on Functional and Object-Oriented Programming, Sobótka, Poland, September 1996.

- “Types for the language designer”, with Michael I. Schwartzbach. Half-day tutorial (32 participants) given at OOPSLA’96, ACM SIGPLAN 11th Annual Conference on Object-Oriented Programming Systems, Languages and Applications, San Jose, California, October 1996.
- “The essence of eta-expansion in partial evaluation”. A series of lectures (40 participants) given at the Summer School on Partial Evaluation: Practice and Theory, Copenhagen, Denmark, July 1998.
- “Four Compiler Courses at Purdue University”. Half-day tutorial (15 participants) given at PLDI’02, ACM SIGPLAN Conference on Programming Language Design and Implementation, Berlin, Germany, June 2002.
- “Compiling with time and space constraints”. A one-hour tutorial (90 participants) given at the Summer School on Application-Specific Multi-Processor System-on-Chip, Château de Pizay, France, July 2002.
- “Programming Sensor Networks”, with Mani Srivastava. Half-day tutorial (28 participants) given at EMSOFT’05, ACM Conference on Embedded Software, Jersey City, New Jersey, September 2005.
- “Programming Sensor Networks”. Half-day tutorial (60 participants) given at Summer School in Wireless Sensor Networks, co-located with 5th International Conference on AD-HOC Networks & Wireless, Ottawa, Canada, August 2006.
- “SSA-based Register Allocation”, with Philip Brisk, Sebastian Hack, Fernando Pereira, and Fabrice Rastello. Half-day tutorial (22 participants) given at CASES’08, Embedded Systems Week 2008, Atlanta, October 2008.
- “SSA-based Register Allocation”, with Philip Brisk, Alan Darte, and Fabrice Rastello. Half-day tutorial (11 participants) given at CGO’09, International Symposium on Code Generation and Optimization 2009, Seattle, March 2009.

Distinguished talks at universities and research institutions.

- “Sensor Network Programming Tools”. A lecture in the Evans and Sutherland Distinguished Lecture Series at the University of Utah, Salt Lake City, Utah, March 2006.
- “Event Driven Software Quality”. ACM Distinguished Lecture, at University of Canterbury, Christchurch, New Zealand, Feb 2008.
- “Event Driven Software Quality”. ACM Distinguished Lecture, at University of Waikato, Hamilton, New Zealand, Feb 2008.
- “Event Driven Software Quality”. ACM Distinguished Lecture, at University of Alberta, Edmonton, Canada, Sep 2008.
- “Event Driven Software Quality”. ACM Distinguished Lecture, at University of Victoria, British Columbia, Canada, Sep 2008.

Talks at universities and research institutions. The talks listed next were given at institutions with which Palsberg was not affiliated at the time.

Queens University, Kingston, Canada (Oct 1990), University of Toronto (Oct 1990), Concordia University, Montreal (Oct 1990), Aalborg University, Denmark (Mar 1991), RWTH Aachen, Germany (Feb 1992), Stanford University (Apr 1992), Sun Microsystems, California (Apr 1992), Kansas State University (May 1992), Yale University (May 1992), Odense University, Denmark (May 1992), University of Copenhagen, Denmark (Nov 1992), AT&T Bell Labs, New Jersey (Jan 1993), University of Karlsruhe, Germany (Jul 1993), University of Passau, Germany (Jul 1993), Brown University (Nov 1993), Carnegie Mellon University (Nov 1993), University of Pennsylvania (Nov 1993), Williams College, Massachusetts (May 1994), Harvard University (May 1994), Massachusetts Institute of Technology (Jun 1994), AT&T Bell Labs, New Jersey (Jun 1994), Massachusetts Institute of Technology (Jun 1994), Yale University (Jun 1994), Aalborg University, Denmark (Nov 1994), UC Berkeley (Feb 1995), Northeastern University (Jul 1995), Princeton University (Jul 1995), Northeastern University (Nov 1995), University of Pennsylvania (Dec 1995), North Carolina State University (Feb 1996), Kansas State University (Feb 1996), Indiana University (Mar 1996), Purdue University, Indiana (Mar 1996), Johns Hopkins University, Baltimore (Apr 1996), Princeton University (Apr 1996), Bell Labs, New Jersey (Apr 1996), Harvard University (Apr 1996), University of California, Santa Barbara (May 1996), University of Aarhus (Dec 1996), University of Technology, Sydney (Jul 1997), University of Sydney (Jul 1997), University of California, Riverside (Oct 1997), University of Aarhus (May 1998), Ball State University, Indiana (Oct 1998), Boston University (Mar 1999), Princeton University (Mar 1999), University of Padova, Italy (May 1999), BT Labs, England (May 1999), University of Aarhus (May 1999), University of Copenhagen (Jun 1999), University of Illinois at Urbana-Champaign (Oct 1999), IBM T.J. Watson Research Center (Dec 1999), University of California, Riverside (Jan 2000), University of Copenhagen (Nov 2000), University of Aarhus (Nov 2000), Colorado State University (Dec 2000), Chalmers University of Technology (Jan 2001), Boston University (Mar 2001), Massachusetts Institute of Technology (Mar 2001), Northeastern University (Mar 2001), University of California, San Diego (Oct 2001), Georgia Institute of Technology (Nov 2001), Microsoft Research, Redmond (Mar 2002), UCLA (Mar 2002), U.C. Irvine (Mar 2002), École Normale Supérieure, Paris (Jul 2002), UCLA (Feb 2003), Rensselaer Polytechnic Institute, New York (May 2003), UC Berkeley (Dec 2003), U.C. Irvine (May 2004), Intel Research Laboratory, Berkeley (Jul 2004), Symantec, Santa Monica, California (Mar 2005), University Federal, Rio de Janeiro (May 2006), PUC, Rio de Janeiro (May 2006), University of Sydney (Nov 2006), University of Aarhus, Denmark (Feb 2007), Intel, Santa Clara (July 2007), University of California, Riverside (Oct 2007), University of Freiburg, Germany [four talks] (Oct 2007), Google, Mountain View, California (Dec 2007), Apple, Cupertino, California (Dec 2007), University of Aarhus, Denmark (Jun 2008), Sun Microsystems, California (Aug 2008), National University of Singapore (Sep 2008), Pomona College, Los Angeles (Oct 2008), Zhejiang University, Hangzhou, China (May 2009).

2.5 Funding

Research Grants

- 1/98–12/99, “Software Evolution”, British Telecom, \$50,000.
- 4/98–3/02, “CAREER: Type Inference for Object-Oriented Software”, U.S. National Science Foundation Faculty Early Career Development Award, CCR–9734265, \$230,000.
- 7/98–6/99, “Evolution of Software via Adaptive Programming”, DARPA via subcontract from Northeastern University, \$24,000.
- 1/99–12/99, “Type Inference for Java: Supporting the Transition From Rapid Prototype to Product”, IBM, \$35,000.
- 7/99–6/04, Purdue University Faculty Scholar Award, \$50,000.
- 7/99–6/00, “Software Security in Distributed Systems”, Lilly Endowment Inc. via subcontract from Center for Education and Research in Information Assurance and Security, Purdue University, \$50,000.
- 10/99–09/00, “Software Security in Distributed Systems”, IBM, \$35,000.
- 7/00–6/02, “Secure Assembly of Software Systems from Components”, Lilly Endowment Inc. via subcontract from Center for Education and Research in Information Assurance and Security, Purdue University, \$76,037.
- 08/00–05/03, “GAANN: Fellowship Initiative in the Development of the Next Generation Computing Infrastructure”, with Susanne Hambruch (PI) and Ananth Grama (co-PI). U.S. Department of Education, \$486,750.
- 6/01–8/04, “DCMF/NES: Dynamic Compositional Middleware Frameworks for Networked Embedded Systems”, with Jan Vitek (PI), Tony Hosking (co-PI), Douglas Lea (co-PI), William Pugh (co-PI). DARPA, \$3,274,680.
- 9/01–8/06, “ITR: Static Timing of Interrupt-Driven Software”, U.S. National Science Foundation Information Technology Research Award, CCR–0112628. \$432,900.
- 9/01–8/02, “Efficient Crypto Implementations for Low-Power Devices”, with T. N. Vijaykumar. Lilly Endowment Inc. via subcontract from Center for Education and Research in Information Assurance and Security, Purdue University, \$75,000.
- 8/02–7/03, “A Model for Surviving Denial of Service Attacks”. Lilly Endowment Inc. via subcontract from Center for Education and Research in Information Assurance and Security, Purdue University, \$40,000.
- 1/03–12/03, “Resource-Aware Compilation for IXP Network Processors”, Intel, \$30,000.
- 1/03–12/03, “A Simplified Eclipse IDE for Computer Science Freshmen”, IBM Eclipse Innovation Award, \$28,000.

- 9/03–8/06, “Foundations of ILP-based Static Analysis”, U.S. National Science Foundation, CCR–0306401, \$270,000.
- 08/03–07/06, “GAANN: Development of the Next-Generation Computing Infrastructure”, with Greg Frederickson (PI) and Ananth Grama (co-PI). U.S. Department of Education, \$793,344.
- 08/03, “Resource-Aware Compilation”, Okawa Research Foundation, \$10,000.
- 07/04, “Curriculum Development in Software Engineering”, Lockheed Martin, \$5,000.
- 09/04–08/07, “ITR: Event Driven Software Quality”, with Edward Kohler, Rupak Majumdar, and Todd Millstein. U.S. National Science Foundation, CCF–0427202, \$1040,000.
- 08/05–6/08, “Aurora: Sensor Network Simulation”, U.S. National Science Foundation via subcontract from Center for Embedded Networked Sensing, UCLA, \$75,000.
- 12/05, IBM Faculty Award, \$30,000.
- 8/06, IBM Equipment Award, \$220,000.
- 7/07–06/10, “SoD: An Electronic Design Automation Approach to Embedded Networked Software”, U.S. National Science Foundation, with Todd Millstein (PI), Jason Cong (co-PI), and Ramesh Govindan (co-PI). CNS–0725354, \$800,000.
- 07/08–06/10, “Virgil: towards certified sensor nodes”, U.S. National Science Foundation via subcontract from Center for Embedded Networked Sensing, UCLA, \$60,000.
- 9/08–8/11, “Certification of Medical Device Software”, U.S. National Science Foundation, with Majid Sarrafzadeh. \$700,000.
- 9/09–8/14, “Customizable Domain-Specific Computing”, with Jason Cong (PI), Vivek Sarkar, Denise Aberle, Richard Baraniuk, Alex Bui, M. C. Frank Chang, Tim Cheng, Miodrag Potkonjak, Glenn Reinman, Saday Sadayappan, Luminita Vese; U.S. National Science Foundation, \$10,000,000.

Stipends

Competitive Selection based on Merit.

- 7/94–6/95, Danish Natural Science Research Council’s Post Doctoral Stipend, 11–1225, DKR 335,075 (approx. \$55,000).
- 7/89–6/91, Danish Natural Science Research Council’s Graduate Stipend, 11–7859, DKR 490,000 (approx. \$81,000).
- 7/88–6/89, Danish Research Academy’s Introductory Stipend, 1988–218/5–20, DKR 230,000 (approx. \$38,000).

3 Teaching

3.1 Vision for Teaching

My professors were excited about the field, and I wanted to have a piece of the excitement. Now I have the chance to pass on the excitement by being a good role model for my students. When I teach a course, I try to show the students that I care about the *topic*, that I care about the *course*, and that I care about *them*. Caring about the topic leads to a lecturing style where I want to radiate a sense of pride that says: “this topic is important, I want you to learn it, and now is your chance to learn it from an expert; ask me questions, and ask me again.” Good lecture planning includes planning for questions. Caring about the course leads to a good course structure, interesting homework, and a well-organized effort by the teaching assistants. A badly organized course can introduce doubt, uncertainty, frustration, and sometimes extra work for the students. The efforts of the students should be to learn the material and not to find their way through a maze created by lack of information, constant corrections to information already given, etc. Caring about the students themselves leads to a generous open-door policy. In every course I have taught, I have told the students: stop by any time. In practice, this means: day and night! While most students come to me during the day, it happens on a regular basis that the students come to me late evening, or 2 am, or whenever they see I am here. If I am in the office, then my door is open, and my students know that they can come in any time. In my courses, I usually have a two-hour midterm exam, and I insist that the students come to my office to pick up the graded exam if they want to know their score. This gives me opportunity to chat with all of the students over a few days and find out how things are going. Such time is well spent and it helps me get a snapshot of what the students think about the course while it is going on, rather than afterwards when I receive the written course evaluations.

The key to good teaching is to understand that a good course can be many, many times better than “learn it on your own.” The professor is the role model, both in the way of thinking and talking about the topic, in the way of handling the course, and in the way of interacting with the students. Paraphrasing a current advertising campaign: “textbook: \$50; pencil and paper: \$5; a professor who is a good role model: priceless.”

3.2 Teaching at UCLA

Compiler Construction (undergraduate course)

Quarter	Year	Overall Rating of the Instructor	Overall Rating of the Course	Number of Students
Fall	2003	7.04	6.91	42

Programming Languages (graduate course)

Quarter	Year	Overall Rating of the Instructor	Overall Rating of the Course	Number of Students
Winter	2004	8.14	7.67	7

3.3 Teaching at Purdue University

Programming Languages (graduate course)

Palsberg has several times taught the graduate course *CS 565 Programming Languages*, 3 credit hours. From the students' course evaluations of CS 565, we have the following average numbers, where 5 is best and 1 is worst.

Semester	Year	Overall Rating of the Instructor	Overall Rating of the Course	Number of Students
Fall	1996	4.5	4.1	41
Spring	1997	4.1	3.9	37
Fall	1997	3.9	3.6	27
Spring	1998	4.3	3.8	28
Fall	1998	4.9	4.6	14
Spring	1999	4.6	4.1	47
Fall	1999	4.5	3.9	26
Spring	2000	4.3	4.0	35
Spring	2001	4.8	4.4	24
Spring	2002	4.7	4.7	24

The graduate course *CS 565 Programming Languages* is a required course, that is, all M.S. and Ph.D. students in the Department of Computer Science must take this course. Palsberg completely revised the course when he started teaching it in 1996. The current syllabus for CS 565 was part of his successful NSF CAREER proposal. He presented an outline of the course (including homework) at a teaching session of a workshop in 1997, and he got many enthusiastic comments.

The course is a hands-on course on programming languages. So far, the two example languages are Java and Scheme.

Syllabus: Interpreters, operational semantics, type systems, type soundness, decision procedures for subtyping, type inference, principal types, typed assembly languages, continuation-passing-style transformation, closure conversion, flow analysis, method inlining, secure information flow, software obfuscation, software watermarking.

The homework includes the implementation of

1. an interpreter for a subset of Java called MiniJava,
2. a decision procedure for a flexible subtype ordering on Java interfaces,
3. a translation of MiniJava to continuation passing style, that is, a form where all method calls are in tail position,
4. a type inference algorithm for a subset of Scheme,
5. a flow-directed inlining algorithm for MiniJava, and

6. a security checker for Java bytecode.

Currently, all programming is in Java. The homeworks can be completed by writing about 10,000 lines of Java.

The course (1) relies on undergraduate-level knowledge of data structures and compilers, (2) covers a fair mix of implementation techniques, algorithms, and theorems, and (3) touches on recent research. Palsberg has successfully taught this course to people with diverse backgrounds, and it seems to be a good preparation for taking other graduate courses in computer science. The course can also stand alone as a course on object-oriented and functional programming, and it seems to serve well as preparation for advanced courses and research on programming languages.

Formal Compiling Methods (graduate course)

Palsberg has also taught the graduate course *CS 661 Formal Compiling Methods*, 3 credit hours. From the students' course evaluations of CS 661, we have the following average numbers, where 5 is best and 1 is worst.

Semester	Year	Overall Rating of the Instructor	Overall Rating of the Course	Number of Students
Fall	2001	4.7	4.5	17

In Fall 2001, the course was a seminar course in which 22 recent papers were presented. Each student presented one paper, and four of the authors of the papers (Matthew Dwyer (Kansas State University), Kathleen Fisher (AT&T Labs-Research), Neal Glew (Intertrust Technologies), and Jakob Rehof (Microsoft Research)) visited Purdue and presented five of their papers. Each week every student wrote a review of a paper, for a total of 11 reviews. The students were asked to write the reviews in the style of a review of a submission to ACM Transactions on Programming Languages and Systems. In the beginning of the course, I gave an introduction to reviewing. The teaching assistant for the course was a Ph.D. student from Purdue's Department of English. Every week, the students would send him a draft of their review, and then he would give them feedback on the writing style, sentence structure, etc. After that, the students could revise their reviews before sending them to me. I would then give them comments on the technical aspects of their reviews.

At the end of the semester, we had a mock program-committee meeting at which we selected those papers we liked best. During the final exam week, the students wrote a summary review of one of those top papers, in the style of the reviews in ACM Computing Reviews.

A report on the experience with teaching this course has appeared in Communications of the ACM [90].

Compilers (undergraduate course)

Palsberg has taught the third-year undergraduate course *CS 352 Compilers: Principles and Practice*, 3 credit hours. From the students' course evaluations of CS 352, we have the following average numbers, where 5 is best and 1 is worst.

Semester	Year	Overall Rating of the Instructor	Overall Rating of the Course	Number of Students
Fall	2000	4.2	3.8	110

In Fall 2000, I used the book *Modern Compiler Implementation in Java* by Andrew Appel. Together with a teaching assistant I designed a 10-week project on compiling a subset of Java, called MiniJava, to MIPS. Each student did the project individually; the project could be completed by writing about 8,000 lines of Java. There were three intermediate languages, called Piglet, Spiglet, and Kanga, each with a grammar, a language specification, and an interpreter. The project had five two-week subprojects: (1) write a type checker for MiniJava, (2) translate from MiniJava to Piglet, (3) translate from Piglet to Spiglet, (4) translate from Spiglet to Kanga, and (5) translate from Kanga to MIPS. For example, the fourth project was about register allocation by graph coloring. Dividing the project into subprojects turned out to have two major advantages: (i) students could do one or more of the subprojects without completing the whole compiler, and (ii) it was easier to grade each subproject than it usually is to grade a whole compiler. Teaching this course in Fall 2000 resulted in that I was chosen as one of the Ten Best Teachers of Undergraduates in the School of Science, Purdue University, for 2001, as selected by junior and senior science students. Based on the experience, I have co-authored the new edition of the textbook with Andrew Appel (Princeton University) [2].

Here are a few quotes from the anonymous evaluations written by the students in Fall 2000:

- Best CS class ever.
- Good job. Professor Palsberg is DA MAN!
- Dr. Palsberg was the best CS professor. He showed concern for the students and was available at all hours. Jens was the best professor, and I hope I have him for some other class.
- This course was, by far, the best CS course I have taken. Jens was able to explain difficult subject matter in a way easy to understand. Jens is the best CS prof. Give this man a raise.
- Jens Palsberg was the best CS prof so far and CS 352 was the first well-organized CS class I've taken. In fact I wrote him a haiku:

You are the coolest
Compilers are the coolest
Good Good, Good Good Good.

- Even though my grade in this class is fairly low (average) I feel that I've learned so much this semester. I wish I could redo the early programs! I would not want to take this class without Palsberg teaching it.
- This class was the most organized CS class, I have been in.

- Jens Palsberg is a name that to me is more than a name. Not many men can bring to the CS "table" what Jens Palsberg can. His incredible style is unmatched by any previous or future CS professor, including Monster Mav or Smooth Sammy Wagstaff. In closing, I would like to say that the CS 352 website was a remarkable one. Thank you for your time, and blessed be this man we call Jens. Good bless Denmark, and her true son, Jens Palsberg.
- Dr. Palsberg is the freaking best professor in this school! I have never seen a professor care more about his students than Jens. My own *mother* shows less concern for my well being than Jens. The projects in this course were *smoothly* designed, quite possibly making this *best* CS class I have *ever* taken. Give this man a raise! He utterly destroys every other professor I've ever had.
- It has been obvious that you and the TA's did a lot of work on the projects. They were well organized. The TA's were really great. They were always available to help us. They always answered questions promptly, even if there were a whole lot. The projects were reasonable, and clearly designed to help us understand the material. It as a hard course but it is probably the best I've taken so far. I'd never want to take this class with a different set of TA's and professor. Great job!
- Professor Palsberg was an extremely good instructor for this course. From what I have heard from past compilers classes, this one seems to have been handled much better than the course has been in the past.
- You rock :-) The TA's and you both seemed very concerned that we learn the concepts. Rock on, homeslice :-)
- This is the best course I ever took at Purdue. The instructor is very good in the sense that he cares the student's learning, sets up a very good (best) TA team, and is very responsible. The TA's are the best I ever met. Very responsible and well prepared. The course materials are well prepared, well designed. The loads for course projects are reasonable; not too low and not too high either.
- Overall, this course is excellent. I'm more than 100% satisfied.

An Introduction to Computer Science (undergraduate course)

Palsberg has been the course administrator for the first-year undergraduate course *CS 180 An Introduction to Computer Science*, 4 credit hours. From the students' course evaluations of CS 180, we have the following average numbers, where 5 is best and 1 is worst.

Semester	Year	Overall Rating of the Course	Number of Students
Fall	2002	4.1	172

3.4 Teaching at other Institutions

Under-graduate courses. *Compiler Construction*, University of Aarhus, Fall 1992, Spring 1993. *Analysis of Programming Languages*, Northeastern University, Spring 1994.

Graduate courses. *Object-Oriented Type Systems*, University of Aarhus, Spring 1993, Spring 1995. *Compiler Design*, Northeastern University, Winter 1994. *Principles of Programming Languages*, Northeastern University, Winter 1994. *Topics in Programming Languages*, Northeastern University, Spring 1994.

Of these courses, Palsberg introduced the *Compiler Construction* course at University of Aarhus. The *Object-Oriented Type Systems* at University of Aarhus course lead to his book with Schwartzbach, and the *Topics in Programming Languages* course at Northeastern University was an advanced graduate course.

3.5 Current Students

Ph.D. students, past qualifiers: Fernando Pereira (UCLA).

Ph.D. students, not past qualifiers: Mahdi Eslamimehr (UCLA), Kannan Goundan (UCLA), Shu-Yu Guo (UCLA). Stephen Kou (UCLA), Jonathan Lee (UCLA), Sean Soria (UCLA).

Member, Ph.D. thesis committees: Michael Emmi (UCLA), Simon Han (UCLA), Eitan Mendelowitz (UCLA), Maryam Moazeni (UCLA), Owen Sizemore (UCLA), Alireza Vahdatpour (UCLA), Wenyao Xu (UCLA), Michael Youssef (UCLA).

3.6 Former Students

Ph.D. students:

- Tian Zhao (Purdue University, 2002). “Type matching and type inference for object-oriented systems”. Now an associate professor at University of Wisconsin, Milwaukee.
- Dennis Brylow (Purdue University, 2003). “Static checking of interrupt-driven software”. Now an assistant professor at Marquette University, Milwaukee.
- Ma, Di (Purdue University, 2004). “Bounding the stack size of interrupt-driven programs”. Now at Synopsys.
- Krishna Nandivada (UCLA, 2005). “Combining Stack Location Allocation with Register Allocation”. Now a researcher at IBM India Research Laboratory, Delhi.
- Christian Grothoff (UCLA, 2006), “Expressive Type Systems for Object-Oriented Languages”. Now a researcher at Technical University Munich, Germany.

- Benjamin Titzer (UCLA, 2007), “Objects to Bits: Efficient Implementation of Object-oriented Languages on Very Small Devices”. Now a researcher at Sun Microsystems Laboratories, Menlo Park, California.
- Fernando Pereira (UCLA, 2008), “Register Allocation by Puzzle Solving”. Now a researcher at UFMG, Brazil.

M.S. thesis students: Nicholas Oxhøj (University of Aarhus, 1992), Carsten Pedersen (University of Aarhus, 1996), Mayur Naik (Purdue University, 2003).

External Member, habilitation committee: Franz Puntigam (Technical University of Vienna, 2001).

External Member, Ph.D. thesis committees: Francois Pottier (Université Paris 7, 1998), Ole Hougaard (University of Aarhus, 1998), William Harrison (University of Illinois, Urbana-Champaign, 2000), Igor Siveroni (Northeastern University, 2002), Galen Williamson (Northeastern University, 2004), Francesco Logozzo (Ecole Polytechnique, Paris, 2004), Florin Craciun (National University of Singapore, 2008), Florent Bouchez (ENS Lyon, 2009).

Opponent, Ph.D. thesis: Johan Agat (Chalmers University of Technology, 2001).

Member, Ph.D. thesis committees: Ignacio Silva-Lepe (Northeastern University, 1994), Paul Bergstein (Northeastern University, 1994), Paul Steckler (Northeastern University, 1994), Cun Xiao (Northeastern University, 1994), Walter Hürsch (Northeastern University, 1995), Linda Seiter (Northeastern University, 1996), Neelam Gupta (Purdue University, 1999), Ladislau-Lehel Bölöni (Purdue University, 2000), Yung-Pin Cheng (Purdue University, 2000), Sudipto Ghosh (Purdue University, 2000), Matthew Knepley (Purdue University, 2000), Yonghong Song (Purdue University, 2000), Kevin Du (Purdue University, 2001), Diego Zamboni (Purdue University, 2001), Joao Cangussu (Purdue University, 2002), Tom Daniels (Purdue University, 2002), Christopher Telfer (Purdue University, 2003), Hoi Chang (Purdue University, 2003), Thomas VanDrunen (Purdue University, 2004), Radu Sion (Purdue University, 2004), Benjamin Kuperman (Purdue University, 2004), Alan Fern (Purdue University, 2004), Bo-Kyung Choi (UCLA, 2004), Krzysztof Palacz (Purdue University, 2004), Yonghua Ding (Purdue University, 2004), Florian Buchholz (Purdue University, 2005), Rong Xu (Purdue University, 2005), Benjamin Greenstein (UCLA, 2006), Anahita Shayesteh (UCLA, 2006), SungWook Yoon (Purdue University, 2006), Philip Brisk (UCLA, 2006), Yizhou Lin (UCLA, 2006), Ramkumar Rengaswamy (UCLA, 2007), Guoling Han (UCLA, 2007), Foad Dabiri (UCLA, 2008), Jeffrey Fischer (UCLA, 2008), Petros Efstathopoulos (UCLA, 2008), Alessandro Warth (UCLA, 2008), Tammara Massey (UCLA, 2009), Shane Markstrum (UCLA, 2009), Ru-Gang Xu (UCLA, 2009), Macneil Shonle (UCSD, 2009), Steve VanDeBogart (UCLA, 2009), Brian Chin (UCLA, 2009).

4 Service

4.1 Service to Purdue University

Department of Computer Science

- Associate Head, Department of Computer Science, Purdue University, 2002–2003.
- Chair, The Faculty Search Committee, Department of Computer Science, Purdue University, 2002–2003. The effort led to the hiring of one full professor, Elisa Bertino, and four assistant professors, Daniel Aliaga, Ninghui Li, Cristina Nita-Rotaru, and Daisuke Kihara.
- Chair, Admission Steering Committee, Department of Computer Science, Purdue University, 2001–2002.
- Chair, Lower Division Curriculum Committee, Department of Computer Science, Purdue University, 2000. The committee designed a revision of the undergraduate curriculum, leading to the introduction of two new courses, CS 182 Foundations of Computer Science and CS 240 C Programming Laboratory, and the deletion of an existing course.
- Member, Advisory Committee to the Department Head, Department of Computer Science, Purdue University, 1997–2001.
- Member, The Graduate Committee, Department of Computer Science, Purdue University, 1997–2000.
- Co-author, Strategic Plan, Department of Computer Science, Purdue University, 1999, 2002.
- Member, The Faculty Search Committee, Department of Computer Science, Purdue University, 1999–2002.
- Member, Admission Steering Committee, Department of Computer Science, Purdue University, 2000–2001.
- Member, Building Campaign Committee, Department of Computer Science, Purdue University, 2001–2003.

School of Science

- Co-chair, Strategic Planning Committee consisting of the University Faculty Scholars, School of Science, Purdue University, 2002–2003.
- Member, School of Science committee to select University Faculty Scholars, Purdue University, 1999, 2002.
- Member, Computer Science Head Search Committee, School of Science, Purdue University, 2001–2002.

- Member, School of Science Faculty Council, Purdue University, 2002.
- Member, Educational Policy and Curriculum Committee, School of Science, Purdue University, 2002.
- Member, Promotions Committee, School of Science, Purdue University, 2002.

Other Services to Purdue University

- Member, Internal Advisory Board, Center for Education and Research in Information Assurance and Security, Purdue University, 1998–2002.

4.2 Service to UCLA

Department of Computer Science

- Chair, Faculty Search Committee, Department of Computer Science, UCLA, 2003–2004, 2007–2008.
- Member, Academic Policy Committee, Department of Computer Science, UCLA, 2003–2004.
- Member, Undergraduate Program Review Committee, Department of Computer Science, UCLA, 2003–2005.
- Chair, Software Systems Field Committee, Department of Computer Science, UCLA, 2004–.
- Chair, Graduate Admissions/Fellowships/TA Committee Department of Computer Science, UCLA, 2004–2007.
- Graduate Vice Chair, Department of Computer Science, UCLA, 2005–2007.
- Member, Policy and Planning Committee, Department of Computer Science, UCLA, 2005–
- Member, Faculty Search Committee, Department of Computer Science, UCLA, 2008–2009.
- Chair, Awards Committee, Department of Computer Science, UCLA, 2008–2009.

4.3 Journal Editorial Boards

- ACM Transactions on Programming Languages and Systems, 2003–.
- IEEE Transactions on Software Engineering, 1997–2001.
- Information and Computation, 2003–.
- Theory and Practice of Object Systems, John Wiley & Sons, 1994–1999.

4.4 Guest Editor

- Guest editor, with Martín Abadi, of ACM Transactions on Programming Languages and Systems, volume 29:3, a special issue on POPL 2005, ACM Press, 2007.
- Guest editor, with Michael Schwartzbach, of Theory and Practice of Object Systems, volume 1:3, a special issue on type systems, John Wiley & Sons, 1995,

4.5 External Review Committees

- Member, first-year external review committee: NSF Large ITR Project, hosted in the Center for Hybrid and Embedded Software and Systems (CHESS) at the University of California, Berkeley, and in the Institute for Software Integrated Systems (ISIS) at Vanderbilt University, 2003.
- Member, five-year external review committee: Institute for Software Research (ISR) at UC Irvine, 2004.
- Member, second-year external review committee: NSF Large ITR Project, hosted in the Center for Hybrid and Embedded Software and Systems (CHESS) at the University of California, Berkeley, and in the Institute for Software Integrated Systems (ISIS) at Vanderbilt University, 2004.
- Member, external review committee: University of California, Riverside, Computer Science Department, Graduate Program, 2007.

4.6 U.S. National Science Foundation Review Panels

In the Computer and Information Science and Engineering directorate:

- Embedded and Hybrid Systems: 2001 (CAREER), 2003, 2007.
- Information Technology Research: 2000.
- Postdoctoral Research Grants: 1997.
- Professional Opportunities for Women in Research and Education: 1998, 1999, 2000.
- Science of Design: 2004.
- Software Engineering and Languages: 1996, 1999, 2001, 2005.
- Theory of Computing: 2001.

4.7 France's National Research Agency Evaluation Committees

France's National Research Agency is in French called *Agence Nationale pour la Recherche (ANR)*.

Member

- Security, Embedded Systems and Ambient Intelligence (*securite, systemes embarques et intelligence ambiante*): 2005.

4.8 Conference Program Committees

Chair

- EMSOFT, International Conference on Embedded Software: 2008 (Atlanta) [co-chair with Luca de Alfaro].
- MEMOCODE, ACM-IEEE Conference on Formal Methods and Programming Models for Co-Design: 2006 (Napa Valley, California) [co-chair with James Hoe].
- PASTE, ACM Workshop on Program Analysis for Software Tools and Engineering: 2002 (Charleston, South Carolina) [co-chair with Matthew Dwyer].
- POPL, ACM Symposium on Principles of Programming Languages: 2010 (Madrid).
- SAS, Static Analysis Symposium: 2000 (Santa Barbara), 2009 (Los Angeles).
- SREIS, Symposium on Requirements Engineering for Information Security: 2002 (Raleigh, North Carolina).
- TACAS, International Conference on Tools and Algorithms for the Construction and Analysis of Systems: 2006 (Vienna, Austria) [co-chair with Holger Hermanns].

Subcommittee Chair

- CASES, International Conference on Compilers, Architectures and Synthesis for Embedded Systems: 2002 (Grenoble, France) [chair of the Compilers and Operating Systems subcommittee].
- DAC, Design Automation Conference: 2005 (San Diego) [chair of the Reconfigurable Computing subcommittee], 2006 (San Francisco) [chair of the FPGA Design Tools and Applications subcommittee],

Member

- ACSD, International Conference on Application of Concurrency to System Design: 2009 (Augsburg, Germany).
- AIOOL, Workshop on Abstract Interpretation for Object Oriented Languages: 2005 (Paris).
- AOSD, International Conference on Aspect-Oriented Software Development: 2002 (Enschede, The Netherlands).

- APLAS, Asian Symposium on Programming Languages and Systems: 2007 (Singapore).
- CATS, Computing: The Australasian Theory Symposium: 2006 (Hobart, Australia), 2008 (Wollongong, Australia).
- CC, International Conference on Compiler Construction: 2008 (Budapest, Hungary).
- CGO, International Symposium on Code Generation and Optimization: 2008 (Boston).
- CPS, International Workshop on Cyber-Physical Systems: 2008 (Beijing).
- DAC, Design Automation Conference: 2003 (Anaheim, California), 2004 (San Diego).
- ECOOP, European Conference on Object-Oriented Programming: 1994 (Bologna, Italy), 1995 (Aarhus, Denmark), 1996 (Linz, Austria), 1997 (Jyväskylä, Finland), 1998 (Brussels, Belgium), 1999 (Lisboa, Portugal), 2000 (Cannes, France), 2001 (Budapest, Hungary), 2003 (Darmstadt, Germany).
- EMSOFT, International Conference on Embedded Software: 2003 (Philadelphia), 2005 (Jersey City), 2007 (Salzburg, Austria).
- ETX, Eclipse Technology eXchange Workshop: 2004 (Barcelona, Spain).
- FOAL, Workshop on Foundations of Aspect-Oriented Languages: 2002 (Enschede, The Netherlands), 2003 (Boston).
- FOOL, ACM Workshop on Foundations of Object-Oriented Languages: 2001 (London).
- FORMATS, Conference on Formal Modeling and Analysis of Timed Systems: 2005 (Uppsala, Sweden).
- FORMATS+FTRTFT, joint conference on Formal Modeling and Analysis of Timed Systems and Formal Techniques in Real-Time and Fault Tolerant Systems: 2004 (Grenoble, France).
- FOSSACS, Foundations of Software Science and Computation Structures: 2004 (Barcelona, Spain), 2010 (Paphos, Cyprus).
- FSE/ESEC, ACM Symposium on the Foundations of Software Engineering, joint with European Software Engineering Conference: 1997 (Zurich, Switzerland).
- GPCE, International Conference on Generative Programming and Component Engineering: 2007 (Salzburg, Austria).
- ICCL, IEEE International Conference on Computer Languages: 1998 (Chicago).
- ICESS, International Conference on Embedded Software and Systems: 2005 (Xi'an, P. R. China).

- ICSE, International Conference on Software Engineering: 2000 (Limerick, Ireland), 2005 (St. Louis, Missouri).
- ICYCS, International Conference for Young Computer Scientists: 1999 (Nanjing, China).
- IEHSC, International Conference on Embedded and Hybrid Systems: 2005 (Singapore).
- ISOTAS, International Symposium on Object Technologies for Advanced Software: 1996 (Kanazawa, Japan).
- ITRS, Workshop on Intersection Types and Related Systems: 2000 (Geneva, Switzerland).
- IWAOOS, Intercontinental Workshop on Aliasing in Object-Oriented Systems: 1999 (Lisbon, Portugal).
- JJT, Conference on Java/Jini Technologies: 2001 (Denver, Colorado), 2002 (Boston).
- JMLC, Joint Modular Languages Conference: 2003 (Klagenfurt, Austria), 2006 (Oxford, England).
- JTRES, Workshop on Java Technologies for Real-time and Embedded Systems: 2005 (San Diego).
- LICS, IEEE Symposium on Logic in Computer Science: 1997 (Warsaw, Poland).
- MEMOCODE, ACM-IEEE Conference on Formal Methods and Programming Models for Co-Design: 2004 (San Diego), 2005 (Verona, Italy), 2007 (Nice, France).
- MobiVirt, Workshop on Mobile Computing and Virtualization: 2008 (Breckenridge, Colorado).
- OOIS, International Conference on Object-Oriented Information Systems: 1997 (Brisbane, Australia).
- OOPSLA, ACM Conference on Object-Oriented Programming, Languages, and Systems: 2002 (Seattle, Washington), 2004 (Vancouver), 2006 (Portland, Oregon).
- PEPM, ACM Conference on Partial Evaluation and Semantics-Based Program Manipulation: 1997 (Amsterdam, The Netherlands).
- PLDI, ACM Conference on Programming Language Design and Implementation: 2003 (San Diego), 2009 (Dublin).
- POPL, ACM Symposium on Principles of Programming Languages: 2000 (Boston), 2009 (Savannah, Georgia).
- QA, Workshop on Quantitative Analysis of Software: 2009 (Grenoble, France).

- SAS, Static Analysis Symposium: 1996 (Aachen, Germany), 1997 (Paris, France), 1999 (Venice, Italy), 2001 (Paris, France), 2006 (Seoul, Korea).
- SBPL, Brazilian Symposium on Programming Languages: 2006 (Itatiaia, Brazil), 2007 (Natal, Brazil).
- SREIS, Symposium on Requirements Engineering for Information Security: 2001 (Indianapolis, Indiana), 2005 (Paris, France).
- TACAS, International Conference on Tools and Algorithms for the Construction and Analysis of Systems: 1998 (Lisbon, Portugal), 2005 (Edinburgh, Scotland).
- TLCA, International Conference on Typed Lambda Calculi and Applications: 2003 (Valencia, Spain).
- VMCAI, Workshop on Verification, Model Checking and Abstract Interpretation: 1998 (Pisa, Italy), 2006 (Charleston, South Carolina).

4.9 Conference Steering Committees

Chair

- POPL, ACM Symposium on Principles of Programming Languages: 2005.

Member

- PASTE, ACM Workshop on Program Analysis for Software Tools and Engineering: 2002–2005.
- POPL, ACM Symposium on Principles of Programming Languages: 2004–present.
- ETAPS, The European Joint Conferences on Theory and Practice of Software: 2005–2006.

4.10 Conference Organizing Committees

General Chair

- LICS, IEEE Symposium on Logic in Computer Science: 2009 (Los Angeles).
- POPL, ACM Symposium on Principles of Programming Languages: 2005 (Long Beach, California).
- SPIN, Workshop on Model Checking Software: 2008 (Los Angeles).

Organizer

- SoCal, Southern California Workshop on Parallel and Distributed Processing and Architecture: 2004 (Los Angeles).
- PDM, Symposium in Honor of Peter Mosses: 2009 (Udine, Italy)

Co-Organizer

- Types, Inheritance and Assignments: organizer, with Michael I. Schwartzbach, workshop held at ECOOP'91 in Geneva, Switzerland, July 1991. The collection of position papers is available from Computer Science Department, University of Aarhus as PB-357. A summary of the discussions at the workshop has been published as [86].
- Programming Languages: organizer, with Chris Hankin and Hanne Riis Nielson, working group at the ACM Workshop on Strategic Directions in Computing Research, MIT, June 1996.
- The Semantic Challenge of Object-Oriented Programming: organizer, with Luca Cardelli, Achim Jung, and Peter O'Hearn, Dagstuhl-Seminar, Schloss Dagstuhl, Germany, June 1998.
- Static Single-Assignment Form Seminar: organizer, with Christian Bertin, Alain Darté, Sebastian Hack, Alan Mycroft, and Fabrice Rastello; Autrans, France, April 2009.

Member of the Organizing Committee

- ECOOP, European Conference on Object-Oriented Programming: member of the executive committee, Kaiserslautern, Germany, 1993.
- International Workshop on Set Constraints and Constraint-based Program Analysis: member of the organizing committee, 1997 (Schloss Hagenberg, Austria), 1998 (Pisa, Italy).
- LCTES, Languages, Compilers, and Tools for Embedded Systems: web chair, with Mayur Naik, 2003 (San Diego).
- LICS, IEEE Symposium on Logic in Computer Science: 2009–present.

4.11 Service to Professional Societies

- Secretary/treasurer of ACM SIGBED, Special Interest Group on Embedded Systems, 2005–2007.
- Vice-chair of ACM SIGBED, Special Interest Group on Embedded Systems, 2007–2009.
- Member of the SIGPLAN CACM Nomination Committee, 2008–2010.

4.12 Other Professional Services

- Member, the Danish national board of external examiners in computer science, 1993–.
- Moderator, the mailing list `objecttypes@daimi.aau.dk` for discussion of type systems for object-oriented programming, 1991–1999.

- Faculty Advisor, Purdue University Beta Chapter of Upsilon Pi Epsilon, International Honor Society for the Computing Sciences, 2002–2003.
- Member, Scientific committee, Summer School on Generative and Transformational Techniques in Software Engineering, July 2005 (Braga, Portugal), July 2007 (Braga, Portugal).