Progress Report of the Max Planck Institute for Software Systems (MPI-SWS)

May 2009–April 2011

September 11, 2011

Max
Planck
Institute
for
Software Systems

Contents

1 Overview

This second progress report of the Max Planck Institute for Software Systems (MPI-SWS) covers the period from April 2009–April 2011. We begin with an overview of the institute's mission and goals, research and organizational visions, accomplishments and challenges. The subsequent sections provide individual progress reports by the institute's nine research groups.

1.1 Challenges in software systems

Software systems is the part of computer science that lays the foundation for the practical use of computers. We interpret the term broadly to include all areas of computer science that contribute to the design, analysis, implementation, and evaluation of software-based systems. Thus, we include research in the design and implementation of dependable, distributed, and embedded systems and networks; databases and information retrieval; programming languages and programming systems; software engineering and verification; security and privacy; and human-computer interaction.

With pervasive use of information technology, the field of software systems continues to see new challenges and opportunities:

- The rise of the Internet, personal mobile devices, and wireless data services have enabled the capture, transmission, aggregation, and search over vast amounts of digital information, and has placed this information at the fingertips of anyone who can afford Internet access. This trend continues to pose new challenges in scalable and fault-tolerant distributed systems and networking, data center computing and storage, and security.

- Software systems are fundamental components in many safety- and business-critical applications, such as automotive and avionic control systems, medical devices, energy management systems, and large-scale commercial infrastructure. The software is increasingly complex and heterogeneous, composed from many different components written in many different languages at many different levels of abstraction. Developing programming methodologies and verification technologies that enable cost-effective design and verification remains a key challenge for software systems research.

- The Internet and web services like online social networks have enabled new forms of expression, communication and interaction for hundreds of millions of people, but pose important challenges for users' privacy, security, and accountability in the online world. The use of software systems to support and enhance a wide range of human activity leads to challenges that must be viewed within their non-technical (i.e., sociological, cultural, economic, and legal) context, for example, by understanding usability and human-computer interfaces.

As a growing research institute in software systems, we seek to build a research environment conducive to long-term, fundamental research on these and other challenges. In particular, we continue to hire faculty who are, individually and as a group, well positioned to address broad challenges in software systems.

1.2 Situation

The MPI-SWS was founded in November 2004 and opened its doors in August 2005. The institute has two sites, one located on the campus of Saarland University (UdS), the other on the campus of the Technical University (TU) Kaiserslautern. The sites are 45-minutes apart by car, door-to-door.

Kaiserslautern and Saarbrücken are cities with about 100,000 and 180,000 inhabitants, respectively. The cities offer attractive surroundings and a low cost of living. Access to major metropolitan areas is easy via high-speed rail (two hours to Paris) and low-cost flights from the local airports (to London, Berlin, Munich, and Hamburg). Frankfurt airport, the closest international hub, is a 60 minute drive from Kaiserslautern and a 90 minute drive from Saarbrücken.

Several research organizations in related areas are located at the two sites. The computer science department at Saarland University ranks among the top five in Germany. The Max Planck Institute for Informatics (MPI-INF) in Saarbrücken focuses on algorithms, graphics and geometric modeling, bioinformatics, databases and information systems. The German Research Center for Artificial Intelligence (DFKI), an applied research lab on artificial intelligence, has locations in both Saarbrücken and Kaiserslautern. The Intel Visual Computing Institute (Intel VCI) in Saarbrücken is a collaborative effort between Intel, MPI-INF, MPI-SWS, DFKI, and UdS. MPI-SWS is part of the Cluster of Excellence on "Multimodal Computing and Interaction" and the newly established Center for IT Security, Privacy and Accountability (CISPA) at Saarland University.

The computer science department at the TU Kaiserslautern ranks in the top quartile of departments in Germany. Kaiserslautern hosts two applied research institutes, the Fraunhofer Institute for Experimental Software Engineering and the Fraunhofer Institute for Industrial Mathematics. There are also a number of information technology startups (and a few mid-sized companies) at both sites.

The MPI-SWS has a total budget of just over EUR 10M per year and 17 faculty positions (eight have been filled). Additional growth is expected through external funding. The planned new institute buildings at the two sites jointly offer space for over 200 researchers and staff. Construction of the buildings started in 2009 (Saarbrücken) and in 2010 (Kaiserslautern), and both are expected to be completed in spring of 2012.

1.3 Mission and strategic goals

The MPI-SWS mission statement reads as follows: "The Max Planck Institute for Software Systems is chartered to conduct world-class basic research in all areas related to the design, analysis, modeling, implementation and evaluation of complex software systems. Particular areas of interest include programming systems, distributed and networked systems, embedded and autonomous systems, as well as crosscutting aspects like formal modeling and analysis of software systems, security, dependability and software engineering."

As an academic institution dedicated to high-risk, long-term research, the primary goal is to have impact primarily through publications, artifacts, and people. We aim to contribute to a stronger and broader base of software systems research in Germany and Europe. In particular, we seek to attract outstanding talent from all over the world, thus broadening the pool of talent in Germany and Europe. We expect that some of our graduates and post-docs will take faculty positions at German and European universities, thereby contributing to the strength and breadth of Software Systems research and education. At the same time, we expect our graduates to be competitive for academic and research positions at top universities and laboratories worldwide.

1.4 Institute development strategy

The core principle for achieving our goals is to hire the most talented researchers (including faculty, post-docs, and students) available within our areas of interest. When hiring faculty, this principle trumps, for instance, trying to develop one specific area or another. Hiring the most talented people necessarily means recruiting from all over the world, and in particular means being able to compete with the top CS departments in the USA. This is primarily because US universities house many of the best Software Systems groups in the world.

Faculty. Senior faculty (director) positions within the MPS are unquestionably attractive. For junior and mid-career faculty, competing with the top universities means matching the conditions they offer to Assistant and Associate Professors. In particular, we need to be able to offer these faculty members internationally competitive salaries, full independence to pursue their research agenda, and career opportunities within the institute (tenure-track), while welcoming them as first-class members of the institute's academic leadership.

It is the institute's policy that all researchers above the post-doc level are independent, tenured or tenure-track faculty. Our tenure evaluation model follows those of US schools: junior faculty are evaluated for tenure during their sixth year. (An internal mid-term review is conducted during the third or fourth year.) Tenure cases are reviewed by a committee appointed by the CPT Section of the MPS and chaired by the vice-president of the MPS.

Tenure-track faculty participate, along with tenured faculty and directors, in most institute-level decisions, including faculty hires, budget allocation, institute policy and other aspects of the institute's academic governance. The only decision in which non-tenured faculty do not participate are decisions to grant tenure. We also provide an informal system of mentorship for our faculty.

Compared to top universities, we can offer our faculty some permanent base funding for post-docs, doctoral students, travel and equipment. This funding has proved quite attractive to junior faculty, as it gets them started early, reduces the pressure to solicit third-party funding, and enables them to pursue collaborations through mutual visits with colleagues at other institutions. The opportunity to teach, combined with the freedom to choose how much and what they want to teach, is also attractive to some faculty members.

The flat organization, while common in the USA, is fairly unconventional relative to other MPIs and German universities. However, the Max Planck Society (MPS) grants its individual institutes substantial flexibility in their organizational and research strategy, which has allowed us to adopt this structure. The model so far has worked wonderfully. While the hiring of senior researchers Peter Druschel, Paul Francis, and Rupak Majumdar were notable successes, the reputation of MPI-SWS has grown significantly through the strength and depth of its tenure-track faculty: Umut Acar, Derek Dreyer, Krishna Gummadi, Rodrigo Rodrigues, and Viktor Vafeiadis. Each of them has the potential to grow into a star within their respective research communities. The appointment of Michael Backes (UdS, Security) as a Max Planck Fellow and Robert Harper (CMU, Programming Languages) as an external scientific member have provided additional visibility.

Our faculty recruiting timeline is aligned with that of US schools. We take applications for faculty candidates in December and January. An internal committee screens the applications and invites candidates for individual interview visits. We interview faculty candidates in the February through April timeframe. The faculty then nominates their selected candidates to an external appointment committee led by the responsible MPS vice-president, which issues a recommendation to the MPS president, who extends the official offers by early May.

Notably, the strength of the existing junior faculty is making it easier to recruit directors. MPI-SWS is no longer an unknown entity—it is now seen as an institute with a strong foundation, solid momentum, and the right trajectory. Director candidates can see that MPI-SWS is a place where they can have outstanding peers within their area. For instance, this was an important consideration for both Paul Francis and Rupak Majumdar. While the main emphasis is on junior hires, we continue to be on the lookout for senior hires.

MPI-SWS has already built a strong international reputation in the broad area of distributed systems and networks (Peter Druschel, Paul Francis, Krishna Gummadi, and Rodrigo Rodrigues) as well as programming languages and verification (Umut Acar, Derek Dreyer, Rupak Majumdar, and Viktor Vafeiadis). Although we continue to interview outstanding candidates in these areas, the top priority now is to expand into other areas. Specifically, we have interviewed candidates in the areas of security, embedded and real-time systems, information systems and databases, and human-computer interaction. This spring, we have had a very strong set of applicants, and we have made offers to a number of outstanding young individuals. [August 2011 Update. Three new faculty have accepted our offer and will join us this fall: Björn Brandenburg (UNC, Real-time systems), Deepak Garg (CMU, Security), and Ruzica Piskac (EPFL, Verification).]

To increase our visibility (especially in communities where we do not yet have a presence), and to start new collaborations in areas bordering our own, we continue to invite senior researchers from areas of interest as part of our distinguished lecture series, and we continue to host guests for long- and short-term research and sabbatical visits. For instance, Johannes

Gehrke (Cornell, Databases) is currently spending a year with us as visiting faculty.

Post-docs. We have been able to attract strong post-docs from around the world. In this reporting period, the following post-docs have joined our institute: Arthur Charguéraud (INRIA), Ruichuan Chen (Peking University), Bin Cheng (HIST), Allen Clement (UT Austin), Joshua Dunfield (McGill, CMU), Bryan Ford (MIT), Saikat Guha (Cornell), Chung-Kil Hur (CNRS, Cambridge), Aniket Kate (University of Waterloo), Mike Rainey (University of Chicago), Stevens Le-Blond (INRIA), and K.C. Shashidhar (KU Leuven, FIESE).

Graduate students. Our initial set of graduate students, who came with Peter Druschel from Rice University, have all graduated and have won strong positions in academia and industrial research. (For more details, see Section 1.7.) Our current student body includes a few students who followed their advisors (one of Rupak's students from UCLA and two of Umut's students from Chicago joined the move to MPI-SWS), but mostly consists of students that have been recruited directly into MPI-SWS.

Overall, the quality of students who are applying and accepting graduate positions is increasing steadily. Our applicant pool and student body is highly international, and we are in the process of establishing pipelines with top undergraduate institutions in India, China and the Middle East. We have recently been able to win students who had competing offers from Boston University, CMU, EPFL, Penn State, Rice, and UPenn. That said, students have also chosen schools such as University of Washington and UC San Diego over us. Increasing the number of outstanding graduate students applications, as well as our ability to compete with institutions in the US for top applicants, remains a priority.

To recruit the best students, we align our PhD student recruiting timetable with the US system. Understandably, early on we had to do significant outreach to find good students. Part of our challenge is that internationally, Germany is not normally on student's radar as a place for systems research, so we are working to educate students and faculty, both domestic and international, about research and study opportunities at MPI-SWS.

An important way to do outreach, of course, is to publish at top conferences. This puts us on the radar of good students as well as professors who advise students where to apply. We have taught at various summer/winter schools in Orleans and La Plagne, Brussels, Tarragona, and Pune. We have an active undergraduate internship program, which has included many interns from IITs, and had over 400 applicants in 2011. We also attract students and post-docs through international collaborations at the Max Planck Society level, such as the Indo-German Max-Planck Center for Computer Science (IMPECS). We also send posters advertising our graduate program to colleagues at many of the leading undergraduate institutions worldwide.

Research software development engineers. The institute is planning to build a team of Research Software Development Engineers (RSDE). These would be highly qualified developers who can understand research and can work with faculty and students on the development of research projects.

1.5 Institute structure

In the first reporting period (August 2005-April 2009), we had focused on the fundamentals of building the organization and establishing a strong research program, particularly targeting the Saarbrücken site. In this reporting period (April 2009-2011), we have shifted our focus to the Kaiserslautern site, which started in January 2009 with the arrival of Paul Francis as the institute's second director. In 2010, Rupak Majumdar joined the institute in Kaiserslautern as the third director, and Umut Acar and Viktor Vafeiadis joined as tenure-track faculty. Having reached balance with four faculty in each site, we are hiring for Saarbrücken this year.

Moving forward, our strategy for placing new faculty will be to hire for the site with fewer faculty, or if both sites have the same number, to alternate sites. As a general rule, we do not give new faculty a choice of site unless there is a two-body situation that can only be solved at one site, or the new faculty has strong personal reasons.

Note in particular that we neither try to co-locate nor try to separate researchers who work in similar research areas. There are pros and cons to either approach, and by making the choice of site "algorithmic", we avoid complex and ultimately unresolvable discussions about the matter. Our current approach of letting the chips fall where they may, combined with persistent efforts to bridge the gap between the two sites, seems to be working out well. For example, we have several cross-site collaborations, both within and across disciplines.

To maintain the collaborative environment that is engendered by MPI-SWS's flat structure, it is critical that we effectively bridge the distance gap between the two sites. In the period that Kaiserslautern has been open, we have used a combination of face-to-face visits and telecommunication. During the first year of its existence (2009), the Kaiserslautern-site researchers visited Saarbrücken twice a month, while Saarbrücken visited Kaiserslautern once a month. Starting mid-year 2010, when Kaiserslautern approached the size of Saarbrücken, we balanced the visits, with each site visiting the other every other week, resulting cumulatively in one visit per week.

The bus ride takes roughly one hour. People tend to have spontaneous as well as planned meetings during the trip, so the time is spent productively. We feel that it may even improve communication among the researchers of one site.

The visits result in weekly face-to-face interactions between members of each site. Indeed, we have found that they tend to foster interaction, both because we feel motivated to exploit the opportunity, and because the visitor is less distracted by the demands of his or her office. Faculty conduct lunchtime meetings during visit days, where institute matters can be discussed face-to-face. Our graduate students have independently come up with ideas to foster interaction amongst themselves: they hold weekly update meetings where they present their progress on what they have been working on and take comments and questions from fellow students. This event was motivated by a desire for students to make the most out of their visit days away from their office. We also tend to schedule seminars and general-audience talks during visit days, including general-audience talks arranged jointly with the Max-Planck Institute for Informatics. These seminars aim to familiarize institute researchers with one another's work.

Although visit days partially make up for the lack of contact between institute members at the two sites, they often don't suffice. Connecting the two sites in real time requires state-of-the-art electronic communications equipment. As a step in that direction, we have set up videocast facilities at both sites, which we use to transmit talks. The setup supports separate channels for the speaker view, his or her presentation slides, and views of both audiences. The quality of these videocasts is quite good and we are finding this to be an adequate solution. For one-on-one or small group settings, we have found Skype video to be a convenient and reasonable approach. Looking forward, we plan to setup a high-quality video wall between the two medium-sized meeting rooms in the two sites. This will be used for larger group meetings as well as for seminars and the like.

The visit days, the availability of institute automobiles, and the electronic communications equipment have gone a long way to bridge the gap between the two sites. Researchers at the two sites have established several active collaborations, e.g., collaborations include those between the groups of Derek Dreyer and Viktor Vafeiadis, Umut Acar and Rodrigo Rodrigues, and Rupak Majumdar and Rodrigo Rodrigues. In addition, there are weekly "systems" seminars, where students and postdocs from Druschel's, Francis's, Gummadi's, and Rodrigues's groups all meet together to discuss current research in areas that span the areas of all four groups. Similarly, there is a weekly "programming-languages and verification" seminar where students and post-docs from Acar's, Dreyer's, Majumdar's, and Vafeiadis's group meet.

1.6 Relationship with local institutions

MPI-SWS maintains close relationships with the universities in Saarbrücken and Kaiserslautern. Faculty at MPI-SWS have developed and taught the following courses:

- Operating Systems, Saarland University, 2009, 2011
- Type Systems for Modules, Saarland University, 2010-11
- Distributed Systems, Saarland University, 2010
- Recent Advances in Computer Systems, Saarland University, 2010-11
- Concurrent Program Logics, Saarland University, 2011
- Verification of Reactive Systems, TU Kaiserslautern, 2011

The Concurrent Program Logics course taught by Derek Dreyer and Viktor Vafeiadis is currently being conducted at both locations (via video-conferencing equipment). It is already registered at Saarland University and is currently in the process of being registered as a graduate course at TU Kaiserslautern.

MPI-SWS faculty participate in the Cluster of Excellence for Multimedia Computing and Communication (MMCI) at Saarland University, the Saarbrücken Graduate School for Computer Science, the Intel Visual Computing Institute (Intel VCI) in Saarbrücken, the new Center for IT Security, Privacy and Accountability (CISPA) at Saarland University, and the Kaiserslautern Science Alliance.

1.7 Accomplishments

Publications and talks. MPI-SWS has produced 84 peer-reviewed publications during the reporting period. We have published in several top conferences in software systems, including POPL, PLDI, CCS, S&P, OSDI, NSDI, LICS, ICFP, SOSP, SIGCOMM, and EMSOFT. 16 (19%) of the these publications have authors from two or more groups. Our publications are listed in the per-group sections (2–8).

MPI-SWS faculty gave 11 invited talks at conferences. We have collectively served as program chairs or co-chairs for 8 conferences. We have served on the PCs of at least 55 conferences and workshops. The details are given in the individual group sections.

Awards and honors. Dreyer was nominated for a 2011 Microsoft Research Faculty Fellowship (MSRFF), and was one of only 16 finalists for the award (out of around 200 applicants, at most one per institution). His ICFP 2010 paper [80] was nominated by ACM SIGPLAN for a CACM Research Highlight.

Majumdar received a Sloan Foundation Fellowship in 2010 and a SIGBED best paper award at EMSOFT 2010.

Backes received the MIT TR35 Award and an ERC Starting Grant in 2009.

Alumni. We are happy to report that we graduated our first PhD candidates during the past two years: Alan Mislove, Andreas Haeberlen, Atul Singh, Animesh Nandi, Ansley Post, and Marcel Dischinger. At a time when many US universities were not even hiring because of the economic situation, Alan and Andreas secured interviews at several top US universities. Alan accepted a faculty position at Northeastern University and Andreas accepted a faculty position at the University of Pennsylvania. Both Andreas and Alan won NSF Career Awards their first year as Assistant Professors at UPenn and Northeastern, respectively. Atul Singh interviewed at research labs in the US and Europe, and accepted a researcher position at NEC Labs, Princeton. Animesh Nandi interviewed at research labs in India, and accepted a researcher position at Bell Labs, India. Ansley Post accepted a position at Google, Zurich. Finally, Marcel Dischinger, who is the most recent graduate and first student to complete his entire doctoral program with MPI-SWS, accepted a research position at Barracuda Networks, Austria.

In addition, several of our postdocs were well placed. Saikat Guha went to MSR India, Meeyoung Cha took a faculty position at KAIST, Bryan Ford accepted a faculty position at Yale University, Boris Köpf accepted a faculty position at IMDEA-Madrid, and Andreas Rossberg took a position at Google, Munich.

2 The Programming Languages and Systems Group

The group was established in February 2010, when Umut joined the institute; this report therefore covers the period from February 2010–April 2011.

2.1 Overview

The group members research the design and implementation of rich programming languages and their applications in a broad range of problems. One of the most interesting aspect of our work is its breadth, which often spans several areas of computer science. I attribute this to the foundational nature of the problems that we study and to the effectiveness of the solutions that we propose.

Personnel. Umut Acar, who joined in February 2010, leads the research group. Three post-docs, Arthur Charguéraud, Joshua Dunfield, and Mike Rainey, joined the group from INRIA, CMU/McGill University, and the University of Chicago, arriving in January 2011, August 2010, and November 2010 (respectively). Two of Umut's junior students from the US, where he had a position before, Matthew Hammer and Yan Chen, also moved to Germany while the more senior students, Ozgur Sumer (University of Chicago), Duru Turkoglu (University of Chicago), and Ruy Ley-Wild[1] (Carnegie Mellon), remained in the US to complete their theses. Ruy Ley-Wild completed his thesis in Fall 2010 and started a post-doc position at IMDEA Software in Spain. Sumer and Turkoglu are defending in June 2011 and finishing this summer.

Publications. We distinguish between papers that are published and those that are forthcoming (accepted but not yet published), within the period of this report that overlap the group's existence. We also include the papers in submission. Our accepted and published papers include five papers in top conferences in four areas of computer science (programming languages, computational geometry, machine learning, and parallelism) and three journal publications. We have eight papers currently under review at several conferences and journals.

- **Published.** Two top conference papers: one paper in PLDI 2010 [5] (a top programming languages conference), one paper in SCG [8] (a top

[1] Ruy did his thesis under my mentorship while he formally remained at CMU.

computational-geometry conference). Several workshop papers: one paper [2] in TaPP (a workshop on provenance), one paper in FCG [11] (a workshop on computational geometry), and one paper by Joshua in ITRS [83] (a workshop on intersection types). Finally, Ruy Ley-Wild completed his Ph.D. and published his thesis [138].

- **Forthcoming.** We have three accepted papers to appear in top conferences: one paper in SCG [12] (a top computational-geometry conference), one paper in SPAA [10] (a top parallelism conference), and one paper [192] in AAAI (a top AI conference). We have three journal papers accepted for publication. Umut and co-authors have a paper [66] in MSCS (Mathematical Structures in Computer Science). Arthur has paper in the Journal of Automated Reasoning [62]. Mike and co-authors have one paper in the Journal of Functional programming [86], while another [52] is invited for publication in the Journal of Functional Programming special issue for ICFP 2010. In addition, we have one paper accepted for publication in the HOTCloud Workshop [53]; this joint work with Rodrigo Rodrigues's group.

- **Under review.** We currently have six conference papers and two journal papers under review: three papers on advances in self-adjusting computation [118, 64, 139] (ICFP, ICFP, and OOPSLA 2011), a paper on a parallel scheduling technique [7] (OOPSLA 2011), a paper on a calculus for provenance [3] (ICFP 2011), and a paper on an incremental/dynamic map-reduce framework [54] (SOCC 2011). Our journal submissions include a paper on adaptive statistical inference [16] (Journal of Machine Learning, JMLR), and a paper on Dynamic Well-Spaced Point Sets [9] (Computational Geometry: Theory and Applications).

Presentations. In addition to conference presentations delivered by the members of the group, Umut was invited as a speaker for a Department Seminar at the University of Bonn in Fall 2010, and for an invited talk at ADS 2011 (the fifth Bertinoro workshop on Algorithms and Data Structures).

Software. We actively develop two major pieces of software that extend the C and ML languages (projects CEAL and Delta ML respectively) to support self-adjusting computation.

Collaborations. Externally, we collaborate with researchers at the University of Edinburgh (James Cheney), Indiana University (Amal Ahmed), Rochester Institute of Technology (Matthew Fluet), University of California at Irvine (Alex Ihler), AutoDesk Inc. (Benoit Hudson), University of Birmingham (Paul Levy and Roly Perera), University of Minho (João Saraiva) with whom we have a grant from the Portuguese Science foundation for collaboration. We recently started collaborations with the research groups of Tom Ball at Microsoft Research (Redmond), and Camil Demetrescu and Irene Finocchi at University of Rome. These collaborations are relatively new and have not yet yielded concrete results such as publications. We continue collaborating closely with students at the University of Chicago (Ozgur Sumer and Duru Turkoglu).

Within MPI-SWS, we collaborate with the research group members of Derek Dreyer, Rodrigo Rodrigues, and Viktor Vafeiadis.

Service. Members of the group were invited to perform relatively significant service work.

- **Editorial board.** Umut serves as a Guest Editor for the Journal of Functional Programming Special Issue devoted to papers from the 2010 International Conference on Functional Programming.

- **General chair.** Umut has been invited to serve as the general chair for DAMP 2012 (ACM SIGPLAN Workshop on Declarative Aspects of Multicore Programming).

- **Program committees.** Umut has been invited to serve on the program committees for the International Conference on Functional Programming (ICFP) 2010, the European Symposium on Algorithms (ESA) 2011, and the Workshop on Theory and Practice of Provenance (TaPP) 2011, as well as on the external program committee for Principles of Programming Languages (POPL) 2012. Joshua Dunfield has been invited to serve on the program committee for the 2011 ACM Workshop on ML.

- **External reviews.** We have reviewed a number of papers externally for journals and conferences such as JFP, POPL, VLDB, PEPM, VINO.

- **Service at MPI-SWS.** Members of the group actively participate in various organizational activities at MPI-SWS:

- Post-doc members of the group, led by Arthur Charguéraud, organize a weekly Seminar on "Programming Languages and Verification," bringing together researchers from the Programming Languages and Verification areas.
- Umut serves on the graduate-program committee, which is responsible for admissions, recruiting, and running the daily operations of the MPI-SWS graduate program. As part of this role, Umut has increased the visibility of MPI-SWS in the best undergraduate and graduate schools in Turkey, attracting a number of excellent students to the graduate program.
- Umut serves as the MPI-SWS representative to Science Alliance, a organization consisting of representatives of local universities and research laboratories.
- Umut performed several concrete "pure service" items, including an assessment of housing conditions for visitors and for institute members in Kaiserslautern and Saarbruecken.
- Umut led the development of this (bi-annual progress) report.

2.2 Research agenda

2.2.1 Research Overview

Our research covers a relatively broad range of issues in the design, implementation, and applications of programming languages and systems. Umut's earlier work includes the development of language features and type systems for rich programming languages [56, 147], and parallelism [4, 208]. More recent work, which is discussed in this report, includes programming languages for data provenance, parallelism, and self-adjusting computation and its applications in computational geometry, machine learning, and distributed computations. In this overview I briefly outline the background for the concrete topics that we have studied in the past 14 months here at MPI-SWS.

Our work on provenance is motivated by the realization that in most of computer science, we view computation as a black box, allowing the programmer or the user to peek into the box only barely if at all. This black-box view of the computation is often unsatisfactory, especially when we require more detailed information about various aspects of computation, e.g., how some data is computed, whether it uses private or public information. We have been working on this problem; the goal is make computation more transparent by allowing the programmer or users to "query" it. The problem naturally connects with the areas of databases, privacy and security, as well as program analysis and compilation techniques such as slicing.

Our work on self-adjusting computation aims to provide language-based techniques for developing highly dynamic software. Within the past several years, there have been some exciting developments particularly in two dimensions: language support and applications, which primarily came out of the work of our Ph.D. students. Ruy Ley-Wild and Yan Chen have developed direct programming-language and compilation support for compiling self-adjusting programs for high-level, functional languages. Matthew Hammer has developed direct compiler support for a low-level C-like language for self-adjusting computation. Duru Turkoglu and Ozgur Sumer of University of Chicago have applied self-adjusting computation to problems for computational geometry and artificial intelligence (and/or machine learning) respectively, making important progress on several open problems and establishing lines of research, which have each been well-received in the respective communities. These students have also led the exploration of interactions between parallelism and self-adjusting computation, developing the first simultaneously dynamic (self-adjusting) and parallel solutions to some important problems.

Our work on applications demonstrates an interesting power of self-adjusting computation techniques: they enable reasoning about problems at an algorithmically higher level. Specifically, the code generated by the compiler of a self-adjusting language can be faster by an asymptotically linear factor, which can result in vast efficiency improvements in practice if the algorithm is carefully designed to make this possible. This allows researchers and programmers to design creative, novel algorithms, while permitting the compiler to improve dramatically the asymptotic complexity without requiring the programmer to make major changes to their code. Several researchers at Microsoft Research (Redmond) have recently shared with us a similar experience with a prototype self-adjusting-computation library that they have had developed. Their experiments pleasantly surprised them: they were able to speedup their computation by many factors by making relatively simple changes to their code base. These researchers are now interested in several research and development projects involving self-adjusting computation. (They have recently submitted a paper on their work to OOPSLA).

Our current work on self-adjusting computation continues to build on the previous work while branching out in several direction. In the past year, we have invented techniques for automatically translating conventional programs into self-adjusting programs; this result, which I consider a breakthrough, should lead to the development of programming languages that require little or no additional programming effort to reap the benefits of self-adjusting computation. In a related line of work we have been developing techniques for compiling low-level languages such as C directly into self-adjusting programs. We have recently completed a prototype of a compiler from C to self-adjusting executable. In a joint project with Rodrigo Rodrigues's group involving several Ph.D. students, we have been researching application of self-adjusting computation techniques to large-scale data processing, e.g., with the MapReduce framework. Our preliminary results to date show that supported by careful engineering, the technique can improve efficiency of many data processing tasks without the programmer changing a single like of work. We achieve this effectiveness by adapting self-adjusting computation to the particular problem domain by taking advantage of our knowledge of the structure of the computation. In a sense, this can be viewed as a domain-specific language.

In a different line of work, we develop scheduling techniques for efficient parallel computation on modern multicore computers. Our approach to this problem combines theoretical analyses that take into account constant factors and practical implementations. Our starting point is the observation that scheduling techniques that rely on traditional complexity analysis that ignore constant factors tied to factors such as closure creation, cache misses cannot be successful in practice because it is exactly these costs that prevent parallelism from being effective in practice.

In the near future, we will continue our work on provenance, parallelism, and self-adjusting computation, and also start researching the interactions between parallel and self-adjusting computations. I view this as a very important direction, because the two techniques benefit from each other in an important way: parallel computation reduces the runtime of a program without reducing the total work performed, while self-adjusting computation reduces the total work. In other words, the techniques act as duals, complementing each other in the work and time relationships. If they can be combined, the techniques can thus reduce the total work and running time simultaneously. Such a development can result in major improvements in practical efficiency in various resources (e.g., time, energy, and hardware).

2.2.2 Traceable data types for self-adjusting computation

This work [5] proposes techniques to enable combining the benefits of self-adjusting computation with programmer-designed efficient dynamic data structures (e.g., queues and stacks) to achieve the benefits of both. The approach overcomes certain limitations of self-adjusting computation's generic propagation techniques by enabling the programmer to use problem-specific solutions soundly within a self-adjusting program. This result has effectively eliminated any penalty that a user would have to incur by using self-adjusting computation (compared to using a hand-crafted algorithm), and has led to several important results in applications areas.

2.2.3 CEAL: Sound self-adjusting computation in C

For the past several years, we have been developing language and compilation techniques for self-adjusting computation suitable for low-level, stack-based languages such as C. These languages are challenging, because without the expressiveness and type safety guarantees of functional languages, ensuring correctness and usability becomes difficult. Our initial work developed a library-based approach that mimics those in functional languages [116, 117]. This approach, however, could not guarantee a crucial soundness property: that the self-adjusting programs would respond to modifications correctly.

To address this problem, we have developed techniques for sound self-adjusting computation in the context of low-level languages. To this end, we consider semantics that explicitly model the low-level structures (such as the run-time stack) on which low-level languages rely on. Unlike the previous "big-step" semantics, we use small step semantics that reflects accurately the underlying, imperative stack-based execution model. Although out formulation is relatively clean, proving it turned out to be major challenge, e.g., a technical problem caused us to withdraw an earlier submission. In addition to proving the semantics sound, we have also developed compilation techniques for realizing this semantics and implemented them. We have recently submitted a paper on this work to OOPSLA 2011 [118]. This recent results take an important step in developing sound techniques for low-level self-adjusting computation. In the future, we plan to develop static analysis and compiler optimization for improving efficiency. This work is the topic of Matt Hammer's Ph.D. thesis.

2.2.4 Implicit self-adjusting computation

We have been developing direct language and compiler support for self-adjusting computation in type-safe language SML [141, 140]. Even though, in the last couple of years, we have made significant progress, the techniques still have still required relatively substantial programmer annotations. For example, it would be quite difficult to explain to an undergraduate student how s/he might write a self-adjusting program using existing approaches. In some ways, the state of the art in writing self-adjusting program was analogous to writing parallel programs with threads, as opposed to writing a parallel program in a more high-level language (e.g., with nested-parallelism or fork-join constructs such as in Cilk, Haskell, NESL, and PML).

Motivated by this challenge, we have been researching the question of whether it would be possible to translate conventional programs automatically into self-adjusting programs, with little or no annotations. Continuing on the analogy to parallelism, this can be viewed as something similar to implicit parallelism, where we can write parallel programs by using high-level language primitives.

This turned out to be a challenging problem; I have picked this problem up at numerous times only to give up after some failed attempts. Finally, committed to solving this problem or proving it impossible, my student Yan Chen and I have tried numerous approaches for over a year, until finally we devised what would form the first iteration of a type-directed translation algorithm that can rewrite purely functional programs to self-adjusting programs. Somewhat surprisingly (though in hindsight this seems more plausible), we have found that the type system guiding the translation is closely related to information flow type systems [181, 169]. The translation is quite tricky, but we have proved some relevant soundness and correctness properties. We have also implemented a prototype version of the system within our Delta ML language and compiler. Our early experimental results are encouraging showing that on the small programs that we have tried, we could come close to hand-written code. A paper on our findings is currently under submission in ICFP 2011 [64]. This research, which I expect to lead to numerous explorations on the design of compilers and optimization techniques for self-adjusting computation, will be the topic of Yan Chen's Ph.D. thesis.

2.2.5 Dynamic computational geometry

Many real-world applications—such as robotics, geographic information systems, circuit design, and computer-aided design—require computational geometry algorithms. In many of these applications, the underlying geometry evolves over time, as certain geometric elements are inserted/deleted and/or the existing input elements move. In this line of research, we develop solutions to dynamic geometric problems. Our motivation for considering these problems is simple: they are hard—many of these problems remain open because they do require developing detailed. As a result, if we can make our techniques work on these problems, then they will likely be effective for many others. In the last year, we have had some breakthrough results and solved two long standing open problems: the problems of computation of meshes of dynamically changing and continuously moving sets of points [8, 9, 12, 10]. We have also showed that the approach generalizes to support parallel computation [10].

Our approach to dynamic computational geometry is to design an algorithm for solving a static instance of the problem where data remains the same. We then translate the static algorithm to a dynamic one by using self-adjusting computation techniques. In previous work similar techniques were applied to simpler problems [6], but these are the first result where we solved sophisticated, open problems that resisted traditional approaches. What has enabled us to make progress on these problems is the higher level of algorithmic abstraction that self-adjusting computation enables. This topic is Duru Turkoglu's Ph.D. thesis.

2.2.6 Machine learning

Many machine learning applications involve repeatedly performing inference on a statistical model that slowly changes over time. Traditionally such changes, however small, require performing a complete inference on the changed model. In the past several years, we have been developing techniques for dynamic inference, where inference can be performed while permitting changes to the underlying statistical model. As a simple example, consider computing the marginal distribution of a leaf node in a Markov chain with n variables. Using standard techniques, upon a change to the conditional probability distribution at one end of the chain, we must perform $\Omega(n)$ computation to compute the marginal distribution of the node at the other end of the chain. Using adaptive inference, such a change can be incorporated to the result in $O(\log n)$ time, i.e., in logarithmic, rather than linear, time.

As in the computational-geometry work, our approach to dynamic inference exploits self-adjusting computation: we first design a static algorithm for adaptive inference that assumes that the model remains unchanged. We then translate the static algorithm into a dynamic one by using self-adjusting computation. Although the problem of adaptive inference has been identified in early nineties, many interesting instance of the problem have remained open until our work. We have published several papers in top conferences on this topic [13, 14, 15]. In the past year, we have prepared a journal version of these papers with some new theoretical and practical results [16], and generalized our results to support parallel computation [192]. This topic is the subject of Ozgur Sumer's Ph.D. thesis.

2.2.7 Programming languages for provenance

The term "provenance" refers to information about how some piece of data may have been created. In many data intensive computations, the origin, the authenticity, and the process by which data are created are of critical important. Provenance problem has become especially important, as our lives rely increasingly on data generated and manipulated by computer systems, ranging from sensors and agents that collect raw data, to end-user applications that analyse and present data for human consumption. Many data-processing systems derive data from multiple sources by complex transformations. An error anywhere in this process can have significant consequences.

Unfortunately, existing programming languages and software systems for tracking and querying provenance are relatively limited and spectacular examples of errors abound. For example, in 2006, a protein crystallographer was obliged to retract five articles, including three published in *Science*, after some of his results could not be reproduced. The problem was that "a homemade data-analysis program had flipped two columns of data, inverting the electron-density map from which his team had derived the final protein structure" [152]. Such problems emphasize the need for *transparent computation*: nontechnical users, programmers, and software agents often need answers to queries about how a computation has derived its results. For example, it should be possible to determine which inputs contributed to what parts of the result, how this would be affected from changes to the input data.

This topic is an active area of study in the database and scientific computing communities; there are literally hundreds of papers on this topic (for surveys, see [57, 67, 188]). Existing approaches, however, consider only certain restricted database languages, which are generally not Turing complete, and support only limited forms of queries concerning where data might be copied from (called "where provenance"), why it has the value it does (called "why provenance"), and what source data it depends on (called"dependency provenance").

We have been developing techniques for answering broad provenance queries in general-purpose languages. Our earlier work [65, 2] focused on establishing some formal ground for this work. In recent work we have proposed a general-purpose approach to allow computations to generate "traces" which can then be used to answer any and all provenance queries. We present techniques for slicing traces to allow their specialization by throwing away parts of the trace that do not influence the provenance query of interest. A key insight that led to this general-purpose approach was the observation that traces themselves can be written in the same language as the programs themselves are written. This formulation makes trace and trace slices "readable" (by the programmer or the user), as well as "executable" so that provenance information can update itself even as the source data that it may depend on changes. We have also presented a prototype implementation that allows us to experiment with some of these ideas and illustrate the benefits of the approach. A paper on this work has recently been submitted to ICFP 2011 [3].

2.2.8 Oracle scheduling

Implicit parallel programming languages, such as Cilk, Haskell, NESL, and PML, provide the user with a high-level interface to specify parallelism while they automate the distribution of work across processors by using a scheduler. The scheduler determines the order in which tasks (i.e., lightweight, independent threads of control) execute and the assignments from tasks to processors. Although there is an extensive literature on task scheduling that addresses how to simultaneously balance work across processors and minimize scheduling costs, the practical efficiency of parallel computation is still limited by heavy scheduling costs.

We aim to improve the state of the art and make parallel schedulers effective in practice To this end, we have gone way back in the literature and started looking into some key scheduling results that parallelism relies on. One such theorem by Brent (a.k.a., the work-time principle) states that if you use P processors, then you will get a speedup of factor P. We know, however, that this does not happen in practice. We believe that the missing link is that this and related theorems ignore an important point: constant factors. Ignoring constant factors in asymptotic analysis of sequential algorithms in terms of the problem size is justified because problems sizes are often large and they often dwarf whatever the constant factors may be. But this is not the case in parallelism: the number of processors are often small constants.

Starting with this perspective, we have generalized Brent's theorem to take into account parallel-task-creation costs, conventionally ignored as constant overheads, and presented a cost semantics for a parallel language that takes these costs into account. These results show that task-creation costs could overwhelm the scheduler and prevent speedups in practice, which confirms experience with parallelism in practice. We then developed techniques for reducing these overheads by relying on an oracle that can help us identify the running time of parallel task. We call this approach *oracle scheduling*, because it can be combined with any scheduler and because it relies on an oracle to estimate the size of a parallel task, before it is run.

We realize an instance of oracle scheduling by combining user annotations indicating the asymptotic-complexity of parallel tasks with judicious use of dynamic profiling to estimate the actual run-time of these tasks. We have implemented a tool that parses a Caml program annotated with complexity annotations (also written in Caml syntax) and generates an instrumented source code that implements our scheduling policy (written in PML, the source language of Manticore). Every function gets compiled in two versions: one parallel version, that evaluates costs and fork tasks only when appropriate, and one purely-sequential version, that ignores all the forking instructions. The scheduling policy relies on the two following rules: (1) If a task too small, then it is executed sequentially. (2) If the parallel version of the code involves a fork instruction $(t_1||t_2)$, then the fork instruction is executed only if both t_1 and t_2 correspond to tasks that are large. Note that this approach is independent of the parallel scheduling algorithms. Using our implementation, we have performed an extensive set of experiments and compared ourselves to the work-stealing scheduler. Our results show that the proposed approach can reduce scheduling overheads down to 5% of the sequential run time and improve the efficiency of parallel programs scheduled anywhere between 20%-400%, sometimes delivering perfect speedups. We recently submitted a paper on this work to OOPSLA 2011 [7].

2.2.9 Incremental map reduce

This is an active collaboration with Rodrigo Rodrigues's group. Our motivation comes from the fact that data sets are growing exponentially, while being modified often by small amounts at a time. In such scenarios, incremental updating of data-intensive computation is very attractive because the complexity of update can be sublinear in the size of the data, taking the sting out of exponential growth. Our initial experiments using our incrementalized version of the MapReduce framework are promising. Our first paper on the topic has appeared in a workshop [53]; we also currently have a more comprehensive paper submitted to SOCC 2011 [54].

2.2.10 Parallel self-adjusting computation

There are a number of interesting, deep relationships between parallel and self-adjusting computations. For example, parallel programs are amenable to efficient incremental updates via self-adjusting computation because they naturally divide the computation into independent pieces (this allows self-adjusting computations generic update mechanism to isolate the effects of dynamic changes to data and perform efficient updates). Due to these and other interactions, combining these two techniques could lead to very interesting outcomes, e.g., efficient parallel updates time while performing minimal total work (determined by the size of the update instead of total size). As special cases of this approach, in papers appearing in SPAA 2011 and AAAI 2011 we have parallelized dynamic (self-adjusting) algorithms for geometric and machine learning problems [10]. Our work on incremental map reduce (briefly described above) also takes advantage of similar principles. A research group and Microsoft Research has also started working on this problem and have developed a parallel self-adjusting computation library in C#. Exploration of this direction will be an important focus for our group in the upcoming years.

3 The Information Security and Cryptography Group

3.1 Overview

The report covers the period from May 2009 – April 2011. The group's research interests are in theoretical foundations and applied aspects of information security and cryptography. Major research topics have been the design and verification of security protocols and implementations, linking formal methods and cryptography, privacy and anonymity, and investigating novel side-channel attacks.

Personnel. The group is led by Max Planck Fellow Michael Backes, and currently has one post-doc (Aniket Kate). Boris Köpf was a post-doc in the group until mid 2009, when he accepted a faculty position at IMDEA, Spain. Michael Backes additionally has the chair for information security and cryptography at Saarland University. The works done in the MPI-SWS group are usually conducted jointly with the university group, which currently has six graduate students (Matthias Berg, Oana Ciobotaru, Sebastian Gerling, Sebastian Meiser, Esfandiar Mohammadi, and Raphael Reischuk).

Collaborations. The group has joint publications with the distributed systems group (led by Peter Druschel) and the verification systems group (formerly led by Andrey Rybalchenko). Externally, the group has collaborated with researchers at Stanford, Penn, ETHZ, Cornell, CMU, Austin, Microsoft Research Cambridge, IBM Research Zurich, CWI Amsterdam, Max-Planck Institute for Informatics, Karlsruhe, Darmstadt, and Venice.

Publications. The group has published regularly in the top conferences and journals of its field. During the reporting period (2009-2011), group members have co-authored four CSF [32, 131, 133, 132], two S&P [27, 38], two CCS [35, 46], two ESORICS [28, 29], one Usenix Security [31], two NDSS [30, 44], two PODC [40, 43], one PETS [39], one FSTTCS [41], one AsiaCCS [25], two TOSCA [45, 36], one OTM [34], two journal [49, 26], and several workshop [33, 37, 48, 47, 42, 47] publications.

Teaching and invited talks. Michael Backes taught the undergraduate Cryptography course and the graduate Advanced Cryptography course at Saarland University in 2009, and a graduate course on Practical Aspects of

Security and the undergraduate Security course in 2010. Additionally, he held four graduate student seminars in the last two years. He received the 2009 teaching award of the CS department for teaching the undergraduate Cryptography course, and the 2010 teaching award for teaching the graduate security course.

Michael was an invited speaker at ETAPS 2011. Michael also lectured at the Spring School on Computationally Sound Proofs (CoSyProofs) as well as the ARTIST Summer School in 2009.

Service. Michael Backes is an Associate Editor of Springer's International Journal on Information Security and Cryptography. He currently serves on the steering committee of CSF and ESORICS, and as the steering committee chair of FMSE. Michael is the program co-chair of CSF 2011, CSF 2010, and ESORICS 2009. Michael served on the program committees of ACM CCS 2011, IEEE S&P 2011, IEEE CSF 2011, ESORICS 2011, Crypto 2011, NDSS 2011, ACM AsiaCCS 2011, FCC 2011, IEEE S&P 2010, IEEE CSF 2010, TCC 2010, ESORICS 2010, PETS 2010, ACM AsiaCCS 2010, FAST2010, IEEE S&P 2009, IEEE CSF 2009, PETS 2009, ARSPA-WITS 2009, ESORICS 2009, FCC 2009, and LPAR 2009.

Awards. In addition to the two teaching awards mentioned above, Michael Backes received the MIT TR35 Award and an ERC Starting Grant in 2009.

3.2 Research agenda

The group's research interests are in theoretical foundations and applied aspects of information security and cryptography. Major research topics have been the design and verification of security protocols and implementations, linking formal methods and cryptography, privacy and anonymity, and investigating novel side-channel attacks.

The group's vision is to develop methodologies and tools for addressing key steps of achieving end-to-end security – from high-level specifications of the desired security requirements for a given task, to a specification of a security protocol that relies on innovative cryptographic primitives, to a secure, executable program. The research interest of the group particularly concentrates on developing a general methodology for automatically devising security protocols and programs based on high-level specifications of selected security requirements and protocol tasks. This methodology also includes novel verification techniques that complement all design phases along with a theory which propagates verification results from phase to phase with the ultimate goal of certified end-to-end security of the constructed solution.

The following sections describe three of our main research thrusts that we have pursued in the last two years. Here we intentionally focus on those works that have been primarily conducted at MPI-SWS as part of the fellowship, in contrast to works that have primarily been conducted at Saarland University. A strict separation of these works does not exist though.

3.2.1 Design and analysis of security protocols and programs

Security proofs of cryptographic protocols and programs are known to be difficult, and work towards the automation of such proofs has started soon after the first protocols were developed. Here we highlight three of our main contributions to this research field in the report period.

First, a major focus of our research in this area is to develop novel abstractions and analysis techniques for dealing with protocols that involve modern cryptographic primitives, such as zero-knowledge (ZK) proofs and secure multi-party computation. In earlier works, we have developed an abstraction of non-interactive zero-knowledge proofs in the applied π-calculus and exemplified the abstraction's applicability to real-world protocols by automatically verifying the secrecy, authenticity, and anonymity properties of the Direct Anonymous Attestation (DAA) protocol using ProVerif (Backes et al., S&P 2008) as well as a novel type system (Backes et al., CCS 2008). The analysis using the type system is modular and compositional, and provides security proofs for an unbounded number of protocol executions. In a similar spirit, we have developed a symbolic abstraction of secure multiparty computation (SMPC) along with an automated verification procedure based on a novel type system [41]. In this paper, we have furthermore used our abstraction and the mechanized verification procedure to analyze the SIMAP protocol – an SMPC-based protocol for sugar-beet auctions that was widely used in Denmark.

Second, a central challenge in the design of security protocols for modern applications is devising protocols that satisfy strong security properties. Ideally, the designer should only have to consider restricted security threats (e.g., honest-but-curious participants); automated tools should then strengthen the original protocols so that they withstand stronger attacks (e.g., malicious participants). We have proposed a general technique for automatically strengthening protocols so that they stay secure even in the presence of compromised participants [32]. The central idea is to automatically transform the original cryptographic protocols by adding non-interactive zero-knowledge proofs and let each participant prove that the messages sent to the other participants are generated in accordance to the protocol. We have used our aforementioned type system for zero-knowledge proofs to automatically verify the security of the transformed protocols. In order to facilitate the devision of executable programs from high-level security specifications, a tool chain was developed for compiling abstract protocol specifications in the applied pi-calculus down to ML and to Java executable code.

Third, we have presented a new type system for verifying the security of cryptographic protocol implementations (instead of symbolic protocol specifications) [36]. The type system combines prior work on refinement types, with union, intersection, and polymorphic types, and with the novel ability to reason statically about the disjointness of types. The increased expressivity enables the analysis of important protocol classes that were previously out of scope for the type-based analyses of protocol implementations. In particular, our types can statically characterize: (*i*) more usages of asymmetric cryptography, such as signatures of private data and encryptions of authenticated data; (*ii*) authenticity and integrity properties achieved by showing knowledge of secret data; (*iii*) applications based on zero-knowledge proofs. The type system comes with a mechanized proof of correctness in Coq and an efficient type-checker.

3.2.2 Computational soundness

Proofs of security protocols are known to be error-prone and, owing to the distributed-system aspects of multiple interleaved protocol runs, difficult for humans to generate. Hence, work towards the automation of such proofs started soon after the first protocols were developed. From the start, the actual cryptographic operations in such proofs were idealized into so-called symbolic or Dolev-Yao models. This idealization simplifies proofs by freeing them from cryptographic details such as computational restrictions, probabilistic behavior, and error probabilities. Unfortunately, these idealizations also abstract away the algebraic properties a cryptographic algorithm may exhibit. Therefore a symbolic analysis may overlook attacks based on these properties. To exclude this possibility, so-called computational soundness results have been established that aim at achieving the best of two worlds: The analysis can be performed (possibly automatically) using symbolic abstractions, but the final results hold with respect to the realistic security models used by cryptographers. We again highlight three of our main contributions to this research field in the report period.

First, we have developed CoSP, a *general framework for conducting computational soundness proofs* of symbolic models and for embedding these proofs into formal calculi [35]. CoSP considers arbitrary equational theories and computational implementations, and it abstracts away many details that are not crucial for proving computational soundness, such as message scheduling, corruption models, and even the internal structure of a protocol. CoSP enables soundness results, in the sense of preservation of trace properties, to be proven in a conceptually modular and generic way: proving x cryptographic primitives sound for y calculi only requires $x + y$ proofs (instead of $x \cdot y$ proofs without this framework), and the process of embedding calculi is conceptually decoupled from computational soundness proofs of cryptographic primitives. We have exemplified the usefulness of CoSP by proving the first computational soundness result for the full-fledged applied π-calculus under active attacks. Concretely, we embed the applied π-calculus into CoSP, and give a sound implementation of public-key encryption and digital signatures.

Second, we investigated the *computational soundness of protocol implementations.* Increasing attention has recently been given to the formal verification of the source code of cryptographic protocols. The standard approach is to use symbolic abstractions of cryptography that make the analysis amenable to automation. This leaves the possibility of attacks that exploit the mathematical properties of the cryptographic algorithms themselves. In this paper, we show how to conduct the protocol analysis on the source code level (F# in our case) in a computationally sound way, i.e., taking into account cryptographic security definitions. We build upon the prominent F7 verification framework (Bengtson et al., CSF 2008) which comprises a security type-checker for F# protocol implementations using symbolic idealizations and the concurrent lambda calculus RCF to model a core fragment of F#. To leverage this prior work, we give conditions under which symbolic security of RCF programs using cryptographic idealizations implies computational security of the same programs using cryptographic algorithms. Combined with F7, this yields a computationally sound, automated verification of F# code containing public-key encryptions and signatures. For the actual computational soundness proof, we use the CoSP framework (described above). We thus inherit the modularity of CoSP, which allows for easily extending our proof to other cryptographic primitives.

Third, we have established the first *computational soundness guarantees for zero-knowledge proofs* [49] and *secure multi-party computation* [41]. For the soundness of ZK proofs, we first had to identify which additional properties a cryptographic (non-interactive) zero-knowledge proof needs to fulfill in order to serve as a computationally sound implementation of symbolic zero-knowledge proofs; this leads to the novel definition of a *symbolically-sound zero-knowledge proof system.* We prove that even in the presence of arbitrary active adversaries, such proof systems constitute computationally sound implementations of symbolic zero-knowledge proofs. This yields the first computational soundness result for symbolic zero-knowledge proofs and the first such result against fully active adversaries of Dolev-Yao models that go beyond the core cryptographic operations. For the soundness of SMPC, we proceeded in two steps: We first established a connection between our symbolic abstraction of SMPC in the applied π-calculus (symbolic setting) and the notion of an ideal functionality for SMPC in the UC framework, which constitutes a low-level abstraction of SMPC that is defined based on bitstrings, Turing machines, etc. (cryptographic setting). In the second step, we have built upon existing results on the secure realization of this functionality in the UC framework in order to obtain a secure cryptographic realization of our symbolic abstraction of SMPC. This computational soundness result holds for SMPC that involve arbitrary arithmetic operations; moreover, it is compositional, since the proof is parametric over the other (non-interactive) cryptographic primitives used in the symbolic protocol and within the SMPC itself. Computational soundness holds as long as these primitives are shown to be computationally sound themselves (e.g., in the CoSP framework).

3.2.3 Automatic discovery and quantification of information leaks

Information-flow analysis keeps track of sensitive information that is processed by a program during its execution. One of the main goals of the analysis is to check whether any sensitive information is exposed to the environment. When information is leaked, the analysis needs to qualitatively and quantitatively assess the extent of the leak. We present three of our research highlights.

First, we have developed DisQuant: the first *automatic method for information-flow analysis* that discovers *what information is leaked* and computes its comprehensive *quantitative interpretation*: The original DisQuant-tool was restricted to deterministic systems [38]; we subsequently extended it to uniform probabilistic systems [132] and to non-uniform probability distributions [25]. In DisQuant, the leaked information is characterized by an equivalence relation on secret artifacts, and is represented by a logical assertion over the corresponding program variables. Our measurement procedure computes the number of discovered equivalence classes and their sizes, which provides a basis for computing complex quantitative properties, e.g., average and worst-case guessing entropy, or covert channel capacity. Our method exploits an inherent connection between qualitative information-flow and program verification techniques. We have provided an implementation of our method that builds upon existing tools for program verification and information theoretic analysis.

Second, we have aimed at *quantifying a system's resistance to so-called side-channel attacks*. Side-channel attacks against cryptographic algorithms aim at breaking cryptography by exploiting information that is revealed by the algorithm's physical execution, e.g., its running time, power consumption, and electromagnetic radiation. In an earlier work, we proposed a novel approach for quantifying a system's resistance to side-channel attacks (Backes and Köpf, ESORICS 2008). In follow-up work, we investigated the use of bucketing to achieve a security countermeasure against timing attacks, which indeed resulted in a provably secure countermeasure [131]. Finally, we have investigated the applicability of blinding techniques to render side-channel attacks infeasible, again yielding provably secure countermeasures [133].

Third, we have conducted research on *identifying novel side-channels*. First, we have shown how *reflections* in the user's eye can be exploited for spying on confidential data [27], based on our earlier work on exploiting reflections in stationary objects (Backes et al., S&P 2008). Recovering reflections in the human eye from a distance is considerably more difficult than recovering reflections from (larger) stationary objects, mainly because of the occurrence of various types of blurs such as motion blur caused by the constant movement of the eye. We have demonstrated how to use existing image deconvolution algorithms to remove such blurs, thereby improving the image quality of the reflections. We have also investigated to what extent monitor images can be reconstructed from the diffuse reflections on a wall or the user's clothes, and provide information-theoretic bounds limiting this type of attack. Moreover, we have examined the problem of *acoustic emanations of printers* [31]. We have presented a novel attack that recovers what a dot-matrix printer processing English text is printing based on a record of the sound it makes, if the microphone is close enough to the printer. In our experiments, the attack recovers up to 72 % of printed words, and up to 95 % if we assume contextual knowledge about the text, with a microphone at a distance of 10cm from the printer. After an upfront training phase, the attack is fully automated and uses a combination of machine learning, audio processing, and speech recognition techniques, including spectrum features, Hidden Markov Models and linear classification; moreover, it allows for feedback-based incremental learning. We have evaluated the effectiveness of countermeasures, and we have described how we successfully mounted the attack in-field (with appropriate privacy protections) in a doctor's practice to recover the content of medical prescriptions.

We admittedly considered the last two projects as fun projects initially. However, in addition to getting published in premium venues of information security, they have received wide attention by the media, with more than 250 newspaper articles and more than 45 appearances in TV and radio casts in Europe on these topics.

4 The Type Systems and Functional Programming Group

4.1 Overview

This report covers the period May 2009–April 2011. The group's research focuses on the design, semantics, verification and implementation of modern programming languages and systems. Major topics of study have included advanced type systems for modular programming and verification, advanced techniques for reasoning about data abstraction and local state, and compositional approaches to verified compilation.

Personnel: The group is led by Derek Dreyer, who joined the institute in January 2008. It currently includes one postdoc, Chung-Kil (Gil) Hur, and three PhD students: Georg Neis, Beta Ziliani, and Scott Kilpatrick. Gil received his PhD in 2010 from the University of Cambridge and joined us in Oct. 2010 after an 11-month postdoc at Laboratoire PPS, CNRS & Université Paris Diderot. Georg, who was formerly a graduate student at Saarland University, joined us in Nov. 2008. He completed his master's thesis with our group in 2009, and passed his qualifying exam in Dec. 2009. He is expected to pass his area exam this year. Beta entered the program in Jan. 2010 with a master's degree from the University of Buenos Aires, and Scott entered the program in Aug. 2010 with a master's degree from the University of Texas at Austin. Both Beta and Scott will be taking their qualifying exams this year.

Andreas Rossberg, who was a postdoc in the group from Aug. 2007 to Jan. 2010, is now a software engineer at Google Munich.

Publications: During the period covered by this report, our primary publication output consists of five papers in top-tier conferences in logic and programming languages (LICS 2009 [76], ICFP 2009 [161], POPL 2010 [81], ICFP 2010 [80], POPL 2011 [123]) and one paper in a leading international workshop (TLDI 2010 [177]).

In addition, Beta published a paper in the HOR 2010 workshop [150] together with his former collaborators at the University of Buenos Aires, and Georg completed his master's thesis [160], which is an extension of our ICFP 2009 [161] paper.

Forthcoming publications: We have a paper accepted to appear in the upcoming LICS 2011 conference [124], which is joint work with Viktor Vafeiadis. Extended journal versions of our LICS 2009 [76] and ICFP 2009 [161] papers were accepted to appear in special issues of *Logical Methods* *in Computer Science* [77] and the *Journal of Functional Programming* [162] devoted to selected papers from the respective conferences.

In addition, Gil has two articles (jointly with his former collaborators) that have been accepted to appear in journals: one in the *Journal of Automated Reasoning* [51], and one in *Logical Methods in Computer Science* [85].

Papers under submission and preparation: We have submitted a paper to ICFP 2011 [102]. We have also been invited to submit an extended version of our ICFP 2010 [80] paper to a special issue of the *Journal of Functional Programming* devoted to selected papers from the conference; we have accepted the invitation and are currently preparing the journal version. We are also preparing journal versions of our ICFP 2008 [82], POPL 2009 [19], TLDI 2010 [177], POPL 2011 [123], and LICS 2011 [124] papers.

In addition, Georg has submitted a paper to OOPSLA 2011 [118] together with members of Umut Acar's group.

Honors and awards: Derek was nominated by the managing director of MPI-SWS for a 2011 Microsoft Research Faculty Fellowship (MSRFF), and was one of only 16 finalists for the award (out of around 200 applicants, at most one per institution). Our ICFP 2010 paper was nominated by ACM SIGPLAN for a CACM Research Highlight (a distinction granted to only a handful of papers per year across all PL conferences).

Presentations: Our conference and workshop publications during this period were presented at the respective venues by various members of our group. Derek presented our LICS'09 and POPL'10 papers, Andreas presented our ICFP'09 and TLDI'10 papers, Georg presented our ICFP'10 paper, and Gil presented our POPL'11 paper.

Besides conference talks, Derek and Andreas have presented our work at a number of international universities and labs with world-class PL groups, including Carnegie Mellon University (May 2009), Microsoft Research Cambridge (Nov. 2009, May 2010), Queen Mary, University of London (Nov. 2009), IT University of Copenhagen (Dec. 2009, Jan. 2011), IMDEA Software, Madrid (Jan. 2010), Laboratoire PPS, Paris (Feb. 2010), and the University of Texas at Austin (Apr. 2010). We have also presented our work at a Wall Street trading firm, Jane St. Capital, that builds most of their software in OCaml (Sep. 2010).

Derek, Georg and Gil were invited to attend a Dagstuhl Seminar on "Modelling, Controlling and Reasoning About State" that took place in August 2010. Derek and Gil gave presentations at the seminar.

Georg gave an extended version of his ICFP'10 talk at the 2011 European

Workshop on Computational Effects.

Collaborations: We have established successful collaborations thus far with Lars Birkedal (ITU-Copenhagen), Amal Ahmed (Indiana), Claudio Russo (MSR-Cambridge), Aleks Nanevski (IMDEA Madrid), and Georges Gonthier (MSR-Cambridge), as well as with Viktor Vafeiadis, head of the Software Analysis and Verification Group at MPI-SWS. Georg has collaborated with members of Umut Acar's group at MPI-SWS (see Umut's section of this report for discussion of that work). He also did an internship at Microsoft Research Cambridge in the fall of 2010, under the supervision of Nick Benton. He is pursuing possible research directions that were initiated during his internship.

We have recently begun research collaborations with Robert Harper (CMU), Jan Schwinghammer (Saarland University), Neel Krishnaswami (MSR-Cambridge), and Simon Peyton Jones (MSR-Cambridge), on a variety of topics described below.

Teaching: Derek taught a graduate seminar in Winter 2010-2011 on *Type Systems for Modules*. The course covered the large body of work on type systems for ML-style modular programming, beginning with work from the 1980s on existential and dependent types, and culminating in Derek's recent work on synthesizing ML-style and mixin-style modules.

This spring, Derek and Viktor Vafeiadis will co-teach a graduate seminar on *Concurrent Program Logics*. The course will cover the long line of program logics for concurrent imperative programs, beginning with the early work on Hoare logic and rely-guarantee reasoning, and culminating in Viktor's recent work on RGSep and concurrent abstract predicates.

Service: Derek served on the program committees for MLPA 2009, MLPA 2010, FOOL 2010, and POPL 2011, and was program chair for TLDI 2011. He has been asked to serve on the program committee for CC 2012 and the external review committee for POPL 2012. He has also refereed papers for the journals TCS, TOPLAS, HOSC, and JFP, as well as for the conferences ICFP 2009, APLAS 2009, POPL 2010, ESOP 2010, LICS 2010, ICFP 2010, and LICS 2011. He has served as workshops co-chair for ICFP 2010 and ICFP 2011, and as steering committee member for the ML Workshop (from 2008 to 2010) and TLDI (since 2010). Lastly, he has served as the moderator of the TYPES mailing list (one of the most widely-subscribed-to mailing lists in the PL community) since April 2009.

Internally to MPI-SWS, Derek served as the chair of the Faculty Recruiting Committee for the 2010 hiring season, and as a member of the same committee for the 2011 hiring season. Since Feb. 2010, he has also served as the elected staff representative for MPI-SWS in the Chemistry, Physics & Technology (CPT) Section of the Max Planck Society, which entails participation in CPT Section meetings three times a year.

Andreas was program chair for the 2009 ML Workshop, was a member of the committee for the ACM SIGPLAN Doctoral Dissertation Award, and refereed several papers for POPL 2011, as well as JFP. (He was also invited to be on the PC for ICFP 2010, but declined due to the constraints of his position at Google Munich.) Gil refereed papers for LICS 2011 and MFPS 2011. Georg refereed papers for ESOP 2010, CSL 2010, and MFPS 2011.

4.2 Research agenda

Modern programming languages provide support for a variety of features from a myriad of programming paradigms—functional, object-oriented, imperative, concurrent, distributed, etc.—and modern programming systems encourage interoperation of libraries written in different languages. While access to many different styles of programming offers programmers a lot of flexibility, it also makes programs hard to write and hard to reason about. Accumulation of features from different paradigms leads to language bloat; incorporating new features into a well-understood language can invalidate the reasoning principles that programmers (and compilers!) rely on; and if care is not taken, invoking libraries written in one language from code in another language can result in violations of basic properties like type safety, due to a mismatch between the languages' implicit reasoning principles. Furthermore, there are a number of real-world programming idioms—*e.g.,* the event-driven model-view-controller paradigm predominant in GUI programming—that we don't even know how to formally specify, let alone verify, because they involve a complex combination of higher-order and imperative programming.

The overall aim of our group's work is to establish foundations for the design and verification of next-generation programming languages and systems. Specifically, we are interested in taking time-tested techniques from type theory and semantics, and adapting them to work for real modern languages and systems, in which the beauty of higher-order, strongly-typed functional programming must mingle with semantically "dirtier" (yet critically useful) features like dynamic type analysis, mutable state, control operators, and concurrency, not to mention low-level (*e.g.,* assembly) code. How do we build languages and logical methods for effectively programming and verifying such heterogeneous, multi-paradigm systems?

Initially, the primary focus of our work was on language design, and in particular the design of module systems that combine the benefits of modularity mechanisms from different languages and paradigms. This line of work, which was the crux of Derek's thesis and postdoctoral research, culminated in the paper we published in ICFP 2008 [82] on MixML, a type system that synthesizes ML-style modules with the OO concept of "mixins". We have continued to study module system design, and Scott Kilpatrick has taken up the mantle of this line of work. (See Section 4.5 for more details.) However, beginning in mid-2008, the group's focus began to broaden significantly to include semantics and verification.

Most of our publications during the current review period have explored novel advances to and applications of the method of *Kripke step-indexed logical relations* (hereafter, KSLR) for establishing principles of local reasoning in multi-paradigm languages. This topic—which originated for us with the desire to prove some very basic things about the data abstraction guarantees of ML modules—has proven something of a gold mine. First, by making progress on the fundamental theory of KSLR, we have unleashed a variety of interesting problems that were previously out of reach (and we have solved some of them). Second, our exploration of this topic has called our attention to challenging problems in related areas as diverse as verified compilation, separation logic, proof mechanization, and language-based security. The KSLR method is directly applicable to some of these problems and not to others, but most importantly it has provided us with a fruitful avenue for interesting problem selection.

In the remainder of this section, we will briefly describe a number of our key contributions from this review period, as well as the new research directions we are actively exploring.

4.3 Logical relations for multi-paradigm programs

Our work on this topic began in 2008 with the goal of proving that the "generative" data abstraction afforded by the functor mechanism in Standard ML actually guaranteed encapsulation of local state, as had been claimed informally for decades. In order to demonstrate this formally, it became clear that we needed a semantic model that accounted for at least most of the key features of the ML language—recursive types, abstract data types (existential types), higher-order functions, and mutable state—and in particular we needed a *relational* model that would allow us to prove the observational equivalence of two ML programs. (Reasoning about observational equivalence is essential to establishing *representation independence*, *i.e.,* that the internal implementation of a program module can be changed without affecting the behavior of the rest of the program.) Much to our surprise, no such relational model existed, largely because the semantic modeling of general, higher-order mutable references in the presence of abstract types had long been a hard open problem.

In our POPL 2009 paper [19]—which was joint work between Amal Ahmed, Derek Dreyer, and Andreas Rossberg—we gave the first relational model for proving equivalences between ML programs with all the aforementioned features (hereafter, the ADR model). Our model employed a synthesis of previous methods, most notably Pitts and Stark's work on *Kripke logical relations* for reasoning about local state [165], and Appel and McAllester's development of *step-indexed models* for modeling recursive types [22], which Ahmed had generalized to account for higher-order state in her thesis work [17]. Our model also extended these previous methods in a critical dimension: whereas Pitts and Stark's model only enabled one to establish fixed invariants on a module's (or an object's) local state, we additionally supported the ability to enforce properties about local state that *evolve* over time, as the rest of the program interacts with the module/object that owns the state.

During the current review period, we have extended our initial work on this topic in various directions.

4.3.1 "Logical" step-indexed logical relations

The basic idea of "step-indexing" is that, if it is difficult to semantically model some feature (*e.g.,* recursive types or higher-order state) without entering into a circular construction, one can stratify the model by a "step index" (a natural number representing the number of steps of computation for which some desired property holds), and define the model inductively on the step index instead. However, a continual annoyance in working with step-indexed logical relations, as well as a stumbling block to their general acceptance, is the tedious, error-prone, and proof-obscuring reasoning about step indices that seems superficially to be an essential element of the method.

For instance, in our POPL 2009 paper, we found that our proofs of representation independence results were cluttered with step-index arithmetic, to the point that their main substance was difficult to ascertain. Thus, it seems clear that widespread acceptance of step-indexed logical relations will hinge on the development of abstract proof principles for reasoning about them. The key difficulty in developing such abstract proof principles is that, in order to reason about programs being *infinitely* logically related—*i.e.,* belonging to a step-indexed logical relation at *all* step indices, which is ultimately what one really cares about—one must reason about their presence in the logical relation at any *particular* step index, and this forces one into finite, step-specific reasoning.

In our LICS 2009 paper [76]—which is joint work between Derek Dreyer, Amal Ahmed, and Lars Birkedal—we present a solution to this dilemma in the form of a logic we call LSLR. Our solution involves a novel synthesis of ideas from two well-known pieces of prior work: (1) Plotkin and Abadi's logic for relational reasoning about parametric polymorphism [166], and (2) Appel, Melliès, Richards, and Vouillon's "very modal model of a modern, major, general type system" [23]. The former provides the basic framework for encoding a logical-relations model and logical-relations proofs *syntactically*, and the latter provides a modal "later" operator (written $\triangleright A$), which enables us to encode in the logic the notion of something being true "after one step of computation".

We show how to encode in LSLR a logical relation that is sound with respect to observational equivalence, based on a step-indexed relational model of Ahmed [18] for a language with abstract and recursive types (but no state). Compared with Ahmed's relation, ours is more high-level and abstract: proofs using it do not require any tedious step-index arithmetic. Furthermore, whereas binary step-indexed logical relations are typically asymmetric, our logic enables the derivation of simple *equational* reasoning principles as well as inequational ones. Finally, our logic-based formulation of step-indexed logical relations brings into relief their close relationship with coinductive bisimulation-based methods, such as Sumii and Pierce's [193].

The extended version of our paper, which will appear soon in LMCS [77], corrects a non-negligible (if ultimately minor) flaw in the original LICS paper and presents significantly improved results. In particular, it encodes a logical relation in LSLR that is not only sound but also complete w.r.t. observational equivalence.

In our POPL 2010 paper [81]—which is joint work between Derek Dreyer, Georg Neis, Andreas Rossberg, and Lars Birkedal—we generalize LSLR to a new logic, LADR, that supports reasoning about higher-order mutable state. Just as LSLR provides a more high-level and abstract characterization of Ahmed's logical relation [18], LADR provides a more high-level and abstract characterization of (a variant of) the ADR model from our POPL 2009 paper [19]. This involves extending the basic framework of LSLR with, among other things, an explicit *necessity* modality ($\Box A$ from S4 modal logic) as well as a fragment of relational separation logic [207]. LADR significantly clarifies the high-level reasoning principles underlying ADR proofs, and allows one to express those proofs without the tedious step-index arithmetic.

4.3.2 Non-parametric parametricity

There is a fundamental tension between type analysis, which is central to dynamically-typed programming, and type abstraction, which is central to statically-typed programming. If one can inspect the identity of an unknown type at run time, then the type is not really abstract, so any invariants concerning values of that type may be broken. Consequently, statically-typed languages that support dynamically-typed programming through a designated type `Dynamic` often prohibit programmers from casting to `Dynamic` any values whose types mention user-defined abstract types. However, this is a rather severe restriction, which effectively penalizes programmers for using type abstraction.

Thus, a number of researchers have proposed that languages with type analysis facilities should also support *dynamic type generation* [175, 201, 176]. That is, when one defines an abstract type, one should also be able to generate at run time a "fresh" type name, which may be used as a dynamic representative of the abstract type for purposes of type analysis. The idea is that the freshness of name generation will ensure that user-defined abstract types are viewed dynamically in the same way that they are viewed statically—*i.e.,* as distinct from all other types.

The question remains: how do we know that dynamic type generation *works*? In a language with intensional type analysis—*i.e., non-parametric* polymorphism—does dynamic type generation provably provide us with the same kinds of abstraction guarantees that we get from traditional parametric polymorphism?

In our ICFP 2009 paper [161]—which is joint work between Georg Neis, Derek Dreyer, and Andreas Rossberg—we answer the above question in the affirmative. We study an extension of System F, supporting (1) a type-safe cast operator, which is essentially a variant of Girard's J operator [100], and (2) a facility for dynamic generation of fresh type names. As a practical language mechanism, the cast operator is somewhat crude in comparison to the more expressive `typecase`-style constructs proposed in the literature, but it nonetheless renders polymorphism *non-parametric.* Our main technical result is that, in a language with non-parametric polymorphism, parametricity may be provably regained via judicious use of dynamic type generation.

Our approach employs KSLR in a style generally similar to the one in our POPL 2009 paper [19]. Here, however, we employ a different, novel form of possible world, which gives relational interpretations to *dynamically* generated type names. This logical relation may be used to reason about parametricity and representation independence in a non-parametric setting.

The extended version of our paper, which will appear soon in JFP [162], includes expanded discussion of various points and presents the metatheory of our logical relations in significantly more detail than space permitted in the conference version.

4.3.3 The impact of higher-order state and control effects on local relational reasoning

The focus of the work on KSLR that we have described so far is primarily on reasoning about programs that actually *use* the interesting, semantically complex features (recursive types, higher-order state, etc.) of the languages being modeled. But of course this is only part of the story. When features are added to a language, they also enrich the expressive power of program *contexts*. Hence, programs that do *not* use those new features, and that are observationally equivalent in the absence of those features, might not be observationally equivalent in their presence. One well-known example of this is the loss of referential transparency in an impure language like ML.

For the purposes of our work, we are interested in relational reasoning about *stateful* programs, so we will be taking a language with some form of mutable state as our baseline. Nonetheless, we feel it is important not only to study the kinds of local reasoning principles that stateful programming can *enable*, but also to understand the principles that powerful features like higher-order state and control effects *disable*.

This latter topic has been broached extensively within the framework of *game semantics*. In the 1990s, Abramsky set forth a research programme (subsequently undertaken by a number of people) concerning what he called the *semantic cube* [1]. The idea was to develop fully abstract game-semantic characterizations of various axes in the design space of ML-like languages. For instance, the absence of mutable state can be modeled by restricting game strategies to be *innocent*, and the absence of control operators can be modeled by restricting game strategies to be *well-bracketed*. These restrictions are orthogonal to one another and can be composed to form fully abstract models of languages with different combinations of effects. Unfortunately, when it comes to reasoning about many actual examples, these game-semantics models do not yet supply a direct technique for proving programs equivalent, except in fairly restricted languages.

In our ICFP 2010 paper [80]—which is joint work between Derek Dreyer, Georg Neis, and Lars Birkedal—we marry the aspirations of Abramsky's semantic cube to the powerful proof method of KSLR. Specifically, we provide the first fully abstract logical relation for an ML-like language with recursive types, abstract types, general references and call/cc (hereafter, the DNB model). Then, we show how, under orthogonal restrictions to the expressive power of our language—namely, the restriction to first-order state and/or the removal of call/cc—we can enhance the proving power of our model in correspondingly orthogonal ways, and we demonstrate this proving power on a range of interesting examples.

The central contribution of this paper is to observe that the degree of freedom with which local state properties may evolve depends directly on which particular effects are present in the programming language under consideration. In order to expound this observation, we first recast the ADR model from our POPL 2009 paper [19] in the more familiar terms of *state transition systems*. The basic idea is that the "possible worlds" of the ADR model are really state transition systems, wherein each state dictates a potentially different property about the heap, and the transitions between states control how the heap properties are allowed to evolve.

Next, we explain how to extend the ADR model with support for first-class continuations via the well-studied technique of *biorthogonality* (aka $\top\top$-closure) [165]. Interestingly, nearly all of the example program equivalences proved in the ADR paper continue to hold in the presence of call/cc, and their proofs carry over easily to the DNB model.

The ADR paper also included several interesting examples that the ADR model was *unable* to reason about. The unifying theme of these examples is that they rely on the *well-bracketed* nature of computation—*i.e.,* the assumption that control flow follows a stack-like discipline—an assumption that is only valid in the *absence* of call/cc. In our paper, we consider two simple but novel enhancements to our state-transition-system model—*private transitions* and *inconsistent states*—which are only sound in the absence of call/cc and which correspondingly enable us to prove all of ADR's "well-bracketed examples". Conversely, we also consider the additional reasoning power gained by restricting the language to first-order state. We observe that this restriction enables *backtracking* within a state transition system, and we demonstrate the utility of this feature on several examples.

The above extensions to our basic state-transition-system model are orthogonal to each other, and can be used independently or in combination. One notable example of this is ADR's tricky "callback with lock" equivalence, an equivalence that holds *in the presence of either* higher-order state or call/cc but not both. Using private transitions but no backtracking, we can prove this equivalence in the presence of higher-order state but no call/cc; and using backtracking but no private transitions, we can prove it in the presence of call/cc but only first-order state. Yet another well-known example, due originally to O'Hearn, is true only *in the absence of both* higher-order state and call/cc; hence, it should come as no surprise that our novel proof of this example involves all three of our model's new features working in tandem.

4.3.4 Future work on Kripke step-indexed logical relations

We are currently investigating several directions for future work on KSLR.

First, in joint work with Lars Birkedal, we are exploring how our logical-relations models may be adapted to reason about ML-like languages that additionally support *concurrent programming*. We are also studying how our techniques may be used to reason about the correctness of state encapsulation as provided by the ST monad in Haskell [136].

Second, in joint work with Neel Krishnaswami, we are exploring the problem of how to verify higher-order imperative components under assumptions about their environment. The techniques we have developed so far are very powerful in that they enable one to prove correctness of a module when linked into an *arbitrary* program context, but we would also like to be able to verify *conditional correctness* of a module under certain semantic assumptions about the behavior of the modules on which it depends.

Third, in joint work with Bob Harper and Jan Schwinghammer, we are exploring the application of KSLR to reasoning about correctness of security protocols as modular programs in a concurrent message-passing language. The idea here is to view the invariant associated with a private key (or channel) as a state transition system governing how the set of messages encrypted on that key (or sent on that channel) may grow over time.

4.4 Equivalence between high-level and low-level programs

In the work on logical relations described thus far, we have focused on proving observational equivalences between programs written in the same language. However, one of the key benefits of relational reasoning techniques is that they generalize to support reasoning about equivalence of programs from *different* languages as well. One key problem that requires such reasoning is *compiler correctness*.

While compiler verification is an old problem, there has been remarkable progress in the last several years in proving the correctness of compilers for increasingly realistic languages with increasingly realistic runtime systems. Of particular note is Leroy's Compcert project [137], in which he used the Coq proof assistant to both program and verify a multi-pass optimizing compiler from Cminor (a C-like intermediate language) to PowerPC assembly.

That said, all the previous work on this topic focuses on the correctness of a *particular* compiler, leaving open the question of how to verify the correctness of assembly code that is hand-optimized or linked together from the output of multiple compilers. The issue is that compiler correctness results are typically established by exhibiting a fairly close simulation relation between source and target code, but code produced by another compiler may obey an entirely different simulation relation with the source program, and hand-optimized code might not closely simulate the source program at all. Thus, existing correctness proofs depend fundamentally on the "closed-world" assumption that one has control over how the whole source program is compiled.

In order to lift the closed-world assumption, Benton and Hur [50] suggest that what is needed is a more abstract, extensional notion of what it means for a low-level program to correctly implement a high-level one—a notion that is not tied to a particular compiler and that, moreover, offers as much flexibility in the low-level representation of high-level features as possible. To define such an equivalence, they give a *logical relation* between the high- and low-level languages. Benton and Hur present their work as the first step towards "compositional compiler correctness". However, the source language they consider—the simply-typed λ-calculus with recursion—is purely functional, and the target language they consider—an SECD machine—is relatively high-level.

In our POPL 2011 paper [123]—which is joint work between Chung-Kil Hur and Derek Dreyer—we study compositional equivalence of high- and low-level programs in a more realistic setting. Our high-level language is an expressive ML-like CBV λ-calculus, supporting abstract types, general recursive types, and general mutable references. Our low-level language is an (only slightly) idealized assembly language. Furthermore, our logical relation is designed to be sound in the presence of garbage collection, under some fairly abstract assumptions about the behavior of the garbage collector that are satisfied by both mark-and-sweep and copying collectors.

Our method depends critically on the use of a variant of the DNB model from our ICFP 2010 paper [80]. The possible worlds of that model are useful in enforcing invariants about low-level data structures (*e.g.,* that a heap-allocated representation of a closure is immutable). They are also helpful in encoding a variety of runtime system invariants, such as the convention concerning callee-save registers and the notion of data liveness. Last but not least, possible worlds enable us to reason quite flexibly about assembly code that uses *local* state in a different manner than the high-level code it implements. An interesting example of this is *self-modifying* assembly code, whose correctness proof involves reasoning about low-level internal state changes—specifically, changes to the code itself—that clearly have no high-level counterpart. This is the essence of what we mean when we say that our relation is *extensional*.

Our work on this problem spawned a related problem of how to verify correctness of low-level programs that interface to a garbage collector. Our initial explorations into this problem resulted in a paper in the upcoming LICS 2011 conference [124], which is joint work between Chung-Kil Hur, Derek Dreyer, and Viktor Vafeiadis. For further details, please see Viktor's Section 10.2 of this report.

For future work, also in collaboration with Viktor Vafeiadis, we are adapting our work on compositional compiler correctness to the setting of a C compiler. Specifically, we are building a compositional version of the CompCertTSO optimizing compiler for Clight TSO—a variant of C supporting multi-threading with a relaxed memory model—that was developed recently at the University of Cambridge by Viktor and his colleagues [187].

4.5 F-ing modules and beyond

Modularity is essential to the development and maintenance of large programs. Although most modern languages support modular programming and code reuse in one form or another, the languages in the ML family employ a particularly expressive style of module system. The key features shared by all the dialects of the ML module system are their support for hierarchical namespace management (via *structures*), a fine-grained variety of interfaces (via *translucent signatures*), client-side data abstraction (via *functors*), and implementor-side data abstraction (via *sealing*).

Unfortunately, while the utility of ML modules is not in dispute, they have nonetheless acquired a reputation for being "complex". Simon Peyton Jones, in an oft-cited POPL 2003 keynote address, likened ML modules to a Porsche, due to their "high power, but poor power/cost ratio". (In contrast, he likened Haskell—extended with various "sexy" type system extensions—to a Ford Cortina with alloy wheels.) Although we disagree with Peyton Jones' amusing analogy, it seems, based on conversations with many others in the field, that the view that ML modules are too complex for mere mortals to understand is sadly predominant.

In our TLDI 2010 paper [177]—which is joint work between Andreas Rossberg, Claudio Russo, and Derek Dreyer—we address the "complexity" complaint head-on by showing once and for all that, contrary to popular belief, the semantics of ML modules is immediately accessible to anyone familiar with System F_ω (the higher-order polymorphic λ-calculus).

How do we achieve this goal? Instead of defining the semantics of modules via a "direct" static and dynamic semantics, we employ an *elaboration* semantics in which the meaning of module expressions is defined by a simple, compositional translation into vanilla F_ω. Our approach thus synthesizes elements of the two alternative definitions of Standard ML modules given by Harper and Stone [120] and Russo [179]. Like the former, we define our semantics by elaboration; but whereas Harper and Stone elaborate ML modules into yet another module type system (a variant of Harper-Lillibridge [119]), we elaborate them into F_ω, which is a significantly simpler system. Like the latter, we classify ML modules using F_ω types; our elaboration effectively provides an *evidence translation* for a variant of Russo's semantics, which lacked a dynamic semantics and type soundness proof.

The main task of the elaboration translation is to insert introduction and elimination forms for existential types and universal types in the appropriate places, as well as to infer coercions between various F_ω types. Thus, our approach substantiates the slogan that *ML modules are just a particular mode of use of System* F_ω. While other researchers have given translations from various dialects of ML modules into System F_ω before, we are the first to define the semantics of ML modules *directly* in terms of F_ω.

Finally, as a way of corroborating the simplicity of our approach (and also as an excuse to learn Coq), we mechanized the soundness of our elaboration translation in Coq using the "locally nameless" approach of Aydemir *et al.* [24]. In the paper, we report on our mechanization experience, which, while ultimately successful, was not as pleasant as we had hoped.

The response to the paper in the functional-programming community has been auspicious, and we are currently preparing a journal version that will account for several module features left out of the workshop version for space reasons. Scott Kilpatrick is working on applying the "F-ing modules" approach to a new module language combining ideas from Derek's previous work on *higher-order modules* [78] and *modular type classes* [79].

In a separate project, Scott and Derek have also begun collaborating with Simon Peyton Jones and colleagues at MSR-Cambridge on incorporating a new and more expressive module system into GHC, the leading Haskell compiler. The idea is to use Derek and Andreas's MixML type system [82] as a starting point, for two reasons. First, MixML makes a distinction between *incomplete modules* (modules with missing pieces) and *functors* (functions from modules to modules), which traditional ML module systems do not. (For various implementation reasons, it appears to be relatively straightforward to extend GHC with support for incomplete modules but not functors.) Second, MixML supports *recursive* linking of incomplete modules, which traditional ML modules do not.

4.6 How to make ad hoc proof automation less ad hoc

The work on logical relations described in previous sections has focused on building the theoretical foundations for verification of multi-paradigm languages. But to actually carry out such verification will require the mechanization and at least semi-automation of our techniques. The project described in this section marks our first steps toward this general goal.

In recent years, interactive theorem proving has been successfully applied to the verification of important mathematical theorems and substantial software code bases. Some of the most significant examples are the proof of the Four Color Theorem [101] (in Coq), the verification of the optimizing compiler CompCert [137] (also in Coq), and the verification of the operating system microkernel seL4 [130] (in Isabelle). The abovementioned proof assistants employ higher-order logics and type systems in order to maximize expressiveness and generality, but also to facilitate modularity and reuse of verification effort.

However, despite the expressiveness of these theorem provers, effective solutions to some verification problems can often only be achieved by writing custom automated procedures in a separate *tactic* language outside of the provers' base logics. While tactics have been deployed successfully (and with impressive dexterity) in a variety of scenarios, they are beset by certain fundamental limitations.

The primary drawback of tactics is that they lack the precise typing of the theorem prover's base logic (and in the case of Coq, they are essentially untyped). This can make them much more difficult to maintain than lemmas, as changes in basic definitions do not necessarily raise type errors in the code of the tactics affected by the changes. Rather, type checking is performed on the goals obtained during tactic execution, resulting in potentially obscure error messages and unpredictable proof states in the case of failure. Moreover, the behavior of a tactic typically cannot be specified, nor can it be verified against a specification. Ideally, we would prefer to provide support for custom automation while remaining within the strongly-typed world of lemmas.

In our recent submission to ICFP 2011 [102]—which is joint work between Georges Gonthier, Beta Ziliani, Aleks Nanevski, and Derek Dreyer—we propose a novel and powerful approach to proof automation in Coq, which avoids the aforementioned problems with tactics by allowing one to program custom automated routines within the expressive dependent type system of Coq itself. Our approach involves a sophisticated application of Coq's *canonical structures*, which have existed in Coq for quite some time [182], but with sparse documentation and (perhaps as a consequence) relatively little use. At a high level, canonical structures may be viewed as a generalization of Haskell's *type classes* [202, 115], in the sense that they provide a principled way to construct default dictionaries of values and methods—and hence support overloading and implicit program construction—as part of the type inference process.

However, unlike in Haskell, where the construction of canonical instances is keyed solely on the *type* belonging to a certain type class, instance construction in Coq may be keyed on *terms* as well. This, together with Coq's support for backtracking during type inference, enables a very flexible style of dependently-typed logic programming. Furthermore, since canonical structures can embed *proofs* of interesting invariants about the instance fields being computed, one can use them to implement custom algorithms (in logic-programming style) *together with* proofs of their (partial) correctness. Thus, just as Haskell type classes are used to infer the canonical implementation of an overloaded term at a given type, canonical structures can be used to infer the canonical *proof* of an overloaded *lemma* for a given instantiation of its parameters. We feel this constitutes a beautiful application of the Curry-Howard correspondence between proofs and programs in Coq.

Perhaps the greatest challenge in making our approach fly is in developing effective and reliable ways of circumventing certain inherent restrictions of Coq's canonical structures, which were not designed with our ambitious application in mind. In particular, in order to implement various useful forms of automation using canonical structures, it is critically important to be able to write *overlapping instances*, but also to control the order in which they are considered and the order in which unification subproblems are solved. None of these features are supported primitively, but they are encodable using a series of simple "design patterns". We illustrate these patterns through several realistic examples involving reasoning about heaps, pointers and aliasing, drawn from recent work on Hoare Type Theory (HTT) [159].

5 The Distributed Systems Group

5.1 Overview

During the reporting period (May 2009–April 2011), the group's research has focused on accountability in a broad sense: accountable virtual machines, reliable accounting in hybrid content distribution networks, accountability for third-party storage systems, and packet attestation for the Internet. Moreover, the group is in the early stages of starting a new project that looks at the tension between accountability and privacy in the online world.

Personnel: The group is led by Peter Druschel and currently has three graduate students (Paarijaat Aditya, Amit Roy, Anjo Vahldiek), with a fourth student (Eslam Elnikety) joining in the Fall. Jeff Hoye continues to work for the Distributed Systems group as a research support staff member on a freelance basis. Prof. Johannes Gehrke (Cornell University) is visiting the group (and the institute) during the 2010-2011 academic year. He is supported by a Humbold Research Award.

Five doctoral students graduated from the group during the reporting period. (Since they came from Rice University with Peter Druschel, they received their degrees from Rice University; however, they spent the last 3-4 years of their doctoral studies at the institute). Alan Mislove joined Northeastern University and Andreas Haeberlen joined the University of Pennsylvania as Assistant Professors in the Fall of 2009. Atul Singh joined the research staff of NEC Labs in Princeton, and Animesh Nandi joined Bell Laboratories in Bangalore in 2009. Ansley Post joined Google Zürich in 2010. Bryan Ford spent one year as a post-doc with the group before joining Yale University as an Assistant Professor.

Collaborations: The group has collaborated closely with the networked systems group (led by Krishna Gummadi) and the dependable systems group (led by Rodrigo Rodrigues). There are also some joint activities with the information security and cryptography group (led by Michael Backes) and the rigorous software engineering group (led by Rupak Majumdar).

Externally, the group has collaborated with researchers at Duke, UPenn, Akamai, Maryland, Princeton, Rice, Northeastern, Intel Research Berkeley, EPFL, ETHZ, Saarland University, Google, TU Berlin/Deutsche Telekom Labs, and TU Munich.

Publications: Group members have co-authored papers that appeared in the ACM/IEEE Transactions on Networks (ToN), Communications of the ACM (CACM), NSDI, OSDI, SIGCOMM, WSDM, Usenix ATC, OPODIS, and several workshops [112, 114, 111, 168, 200, 157, 167, 199, 90, 191]. Bryan Ford co-authored two draft RFCs [105, 190].

Patents: Ansley Post, Rodrigo Rodrigues and Peter Druschel filed a provisional patent on "Protecting Data Integrity with Storage Leases" in October 2010.

Teaching and invited talks: Peter Druschel co-taught (with Rodrigo Rodrigues) a new graduate level course on Distributed Systems and an undergraduate Operating Systems course at Saarland University in 2010 and 2011, respectively.

Peter gave invited talks at the Workshop on Theoretical Aspects of Dynamic Distributed Systems (TADDS), Elche, Spain, in 2009, and at the First Mysore Park Workshop on Cloud Computing, Mysore, India, in 2010.

Service: Peter Druschel is on the editorial board of the ACM Transactions on Computer Systems and the Communications of the ACM. He serves on the technical steering committee of ACM SIGCOMM, and the steering committees of EuroSys, the ACM SIGOPS Asia-Pacific Workshop on Systems (APSys) and the EuroSys/INRIA Winter School on Hot Topics in Distributed Systems (HTDC).

Peter is a member of the technical advisory board (TAB) of Microsoft Research, Cambridge, and he serves on the scientific advisory boards of the SICS Center for Networked Systems, Stockholm, and the Tata Consultancy Services TECS Week, Pune. He is also a member of the Core Panel of the ATTRACT program of the Luxembourg National Research Fund (FNR), and he serves on the expert group on Privacy, Accountability and Trust of the European Network and Information Security Agency (ENISA).

Peter is chairing the SOSP 2011 program committee and served on the program committees of ACM PODC 2009 and Usenix NetEcon 2010. He continues to serve on the ACM Sigops Hall of Fame Award Committee and chairs this committee starting in 2011.

Within the Max Planck Society (MPS), Peter Druschel serves on the strategy committee (Perspektivenkommission) of the Chemical, Physical, and Technology Section (CPTS), and he serves on the information technology board (BAR). He has also served on a committee that advises the MPS during the process of creating the new Max Planck Institute for Intelligent Systems (formerly Autonomous Systems) in Tübingen and Stuttgart.

Peter served as the institute's Managing Director (a position that rotates among the senior faculty) until Dec 31, 2009.

Awards: Former doctoral students Andreas Haeberlen and Alan Mislove both won NSF Career Awards during their first year as Assistant Professors at UPenn and Northeastern, respectively.

Michael Backes, Peter Druschel, Paul Francis, Krishna Gummadi and colleagues at Saarland University and the German Research Centre for Artificial Intelligence (DFKI) were awarded a Center of Excellence grant entitled "Center for IT Security, Privacy and Accountability (CISPA)" by the German Federal Government. The initial award is for approximately EUR 5M over four years.

5.2 Research agenda

The group's research seeks to understand, design and build innovative computer systems, including distributed systems, operating systems and networks. The research spans the spectrum from principles, analysis and design to the implementation and deployment of novel systems.

Recently, the group's main research thrust has been in developing techniques to provide accountability in distributed systems. More specifically, we investigate techniques to ensure that deviations from the expected behavior (as defined by a protocol or specification) can be detected reliably and linked irrefutably to the responsible party. Accountability provides fault detection, ensures transparency, and creates incentives for participants to comply.

During the reporting period, the group

- developed a practical and efficient implementation of accountable virtual machines (AVM) based on the commercial VMware Workstation product, for which we obtained an educational source code license;
- developed and evaluated a system that provides packet attestation for the Internet;
- developed mechanisms to provide accountability in a commercial content distribution network based on a hybrid peer-to-peer and infrastructure-based architecture.

In the following subsections, we describe selected research efforts in more detail.

5.3 Practical accountable virtual machines

This effort was led by Paarijaat Aditya, with contributions from Andreas Haeberlen (now UPenn), Rodrigo Rodrigues and Peter Druschel.

An *accountable virtual machine (AVM)* provides users with the capability to audit the execution of a software system by obtaining a log of the execution, and comparing it to a known-good execution. This capability is particularly useful when users rely on software and services running on machines owned or operated by third parties. Auditing works for any binary image that executes inside the AVM and does not require that the user trust either the hardware or the accountable virtual machine monitor on which the image executes. Several classes of systems exemplify scenarios where AVMs are useful:

- in a competitive system, such as an online game or an auction, users may wish to verify that other players do not cheat, and that the provider of the service implements the stated rules faithfully;
- nodes in peer-to-peer and federated systems may wish to verify that others follow the protocol and contribute their fair share of resources;
- cloud computing customers may wish to verify that the provider executes their code as intended.

In these scenarios, software and hardware faults, misconfigurations, break-ins, and deliberate manipulation can lead to an abnormal execution, which can be costly to users and operators, and may be difficult to detect. When such a malfunction occurs, it is difficult to establish who is responsible for the problem, and even more challenging to produce evidence that proves a party's innocence or guilt. For example, in a cloud computing environment, failures can be caused both by bugs in the customer's software and by faults or misconfiguration of the provider's platform. If the failure was the result of a bug, the provider would like to be able to prove his own innocence, and if the provider was at fault, the customer would like to obtain proof of that fact.

AVMs address these problems by providing users with the capability to detect faults, to identify the faulty node, and to produce *evidence* that connects the fault to the machine that caused it. These properties are achieved by running systems inside a virtual machine that 1) maintains a log with enough information to reproduce the entire execution of the system, and that 2) associates each outgoing message with a cryptographic record that links that action to the log of the execution that produced it. The log enables users to detect faults by replaying segments of the execution using a known-good copy of the system, and by cross-checking the externally visible behavior of that copy with the previously observed behavior. AVMs can provide this capability for any black-box binary image that can be run inside a VM.

AVMs detect integrity violations of an execution without requiring the audited machine to run hardware or software components that are trusted by the auditor. When such trusted components are available, AVMs can be extended to detect some confidentiality violations as well, such as private data leaking out of the AVM.

Our recent work [112] makes three contributions: 1) it introduces the concept of AVMs, 2) it presents the design of an *accountable virtual machine monitor (AVMM)*, and 3) it demonstrates that AVMs are practical for a specific application, namely the detection of cheating in multi-player games. Cheat detection is an interesting example application because it is a serious and well-understood problem for which AVMs are effective: they can detect a large and general class of cheats. Out of 26 existing cheats we downloaded from the Internet, AVMs can detect every single one—without prior knowledge of the cheat's nature or implementation.

We have built a prototype AVMM based on VMware Workstation, and used it to detect real cheats in Counterstrike, a popular multi-player game. Our evaluation shows that the costs of accountability in this context are moderate: the frame rate drops by 13%, from 158 fps on bare hardware to 137 fps on our prototype, the ping time increases by about 5 ms, and each player must store or transmit a log that grows by about 148 MB per hour after compression. Most of this overhead is caused by logging the execution; the additional cost for accountability is comparatively small. The log can be transferred to other players and replayed there during the game (online) or after the game has finished (offline).

While our evaluation in [112] focuses on games as an example application, AVMs are useful in other contexts, e.g., in p2p and federated systems, or to verify that a cloud platform is providing its services correctly and is allocating the promised resources [111]. Our prototype AVMM already supports techniques such as partial audits that would be useful for such applications, but a full evaluation is part of future work.

5.4 Reliable accounting in a hybrid CDN

This effort is led by Paarijaat Aditya, with contributions from Andreas Haeberlen (now UPenn), Bruce Maggs (Duke), Mingchen Zhao (UPenn), Yin Lin (Duke), Bill Wishon (Akamai), and Peter Druschel.

We use accountability techniques to provide reliable accounting in a hybrid peer-to-peer (p2p) and infrastructure-based content distribution network. We apply and evaluate these techniques in the context of the Akamai commercial CDN.

Background The NetSession Download Manager (DLM) is a client-side software agent distributed by Akamai. To access the content distributed by selected Akamai customers (i.e., content providers), clients have to install the DLM on their computers. The DLM uses a combination of p2p downloads from other DLMs and downloads from the Akamai edge nodes to distribute the customer's bulk content (e.g., video) efficiently.

With the introduction of p2p transfers among DLM clients, Akamai's CDN has adopted a hybrid p2p and infrastructure based architecture. This architecture uses the trusted infrastructure nodes to control p2p transfers, thereby avoiding many of the challenges associated with more decentralized p2p systems. For instance, infrastructure nodes maintain a centralized index of which DLMs store which content, control which DLMs download from which other DLMs, and provide authenticated metadata, enabling clients to verify the integrity of content obtained from other peers.

Nevertheless, the introduction of client-side software and p2p transfers introduces new challenges to reliable accounting and monitoring. Clients can modify the DLM software in ways that are not detectable by the infrastructure, and they can tamper with the records of past p2p transfer activity before those records are uploaded to the infrastructure. In particular,

- clients who tamper with their logs can cause inconsistencies in the up- an downloads reported by different peers, which can affect the accuracy of the CDN's billing.
- clients can tamper with the client software in order to reduce the quality of service provided by the systems. For instance, a client can abort peer transfers or upload incorrect content blocks, thus forcing a peer to re-try the download from another peer.

We have successfully demonstrated one such attack in the live system, under controlled conditions and with Akamai's prior approval.

In the current system, it is very difficult to attribute these problems to a particular peer. As a result, the CDN is faced with a choice to either tolerate the resulting problems, or to risk taking action against a well-behaved client. So far, Akamai has not observed such behavior; however, as the deployment expands, it is only a matter of time until the system will face such attacks.

Approach We have designed and implemented, and are in the process of evaluating, a set of mechanisms that significantly raise the bar for client-based attacks on a hybrid CDN. The systems relies on the tamper-evident logs we used in our earlier accountability work. In addition, we had to develop new techniques to enable scalable and efficient fault detection in the context of a hybrid CDN. Specifically, we had to design techniques to reduce the overhead on the infrastructure nodes associated with tamper-evident log checking and fault detection:

- To avoid the overhead of checking the signature of every message transfer recorded in the tamper-evident logs, we developed a technique that allows clients to reduce the number of signatures within a tamper-evident log to one per peer that a node has communicated with during the time period covered by the log. The technique dramatically reduces the number of signatures the infrastructure has to check, without requiring trust in the DLM clients.
- To avoid the overhead of replay-based fault detection, we have developed a fault detector that checks logs based on an abstract model of the state machine that a correct DLM client is expected to follow. The hand-crafted detector is orders of magnitude more efficient than the generic replay-based fault detectors used in earlier work.
- Additionally, to detect attacks by colluding sets of clients (e.g., those based on a botnet), we developed a set of statistical tests that detect deviation from normal peer activity, which may indicate a collusion attack. The model of normal peer activity is based on a set of month-long peer activity traces we obtained from the live Akamai system.

A IMC 2011 submission is currently in preparation, which will present an analysis of the peer activity in live traces we obtained from the Akamai system. For NSDI 2011, we are preparing a submission describing our accountability technique for a hybrid CDN.

5.5 Internet packet attestation

This effort is led by Andreas Haeberlen (now UPenn), with contributions from Pedro Fonseca, Rodrigo Rodrigues and Peter Druschel.

With cybercrime on the rise, governments, law enforcement agencies, and copyright owners are increasingly calling for better *attribution* on the Internet [148]. Here, attribution means the ability to identify whoever is responsible for sending a given set of network packets. With reliable attribution in place, it would be much easier to quickly establish countermeasures to an ongoing attack and to hold cybercriminals accountable for their actions.

Unfortunately, reliable attribution is difficult because there is no direct support for it in the Internet—packets do not have “license plates” that tie them irrefutably to the party responsible for sending them [69]. Moreover, adding such a capability to the Internet would face substantial technical, legal, political and administrative challenges. It would need strong user authentication and secure software on every network-attached device, and it would require an internationally accepted user certification authority and jurisdiction to bring criminals to justice no matter where they reside. It would also raise privacy concerns, and potentially undermine the Internet as a platform for whistleblowers and dissidents. Worst of all, such a capability would still not be sufficient to catch sophisticated cybercriminal who rely on multi-stage attacks through a series of compromised machines owned and operated by other victims [68].

Clark and Landau [69] recently made similar points and added that not only is strong attribution insufficient to cover multi-stage attacks, it is also not strictly necessary to investigate cybercrime. What victims and law enforcement agencies need most is evidence sufficient to (1) confidently take immediate steps to stop an ongoing attack, and (2) provide a starting point for an investigation of the responsible party. In serious cases, such an investigation must in any case rely on additional evidence, obtained through a search warrant or by following the trail of money, to be admissible in a court of law.

Today, network administrators and law enforcement agents rely on circumstantial evidence, such as log entries and IP addresses, to attribute traffic to a source. For example, if a server log contains an entry that indicates illegal traffic from a particular IP address, this IP address is combined with other information (such as IP prefix ownership and DHCP server logs) to identify a particular subscriber, who is then presumed responsible [174].

Relying on such evidence alone for attribution is problematic. First, the process involves several manual steps, which is labor-intensive and error-prone. The consequences of mistakes can be severe: innocent users can be falsely accused [194, 95] or even go to jail [125]. Second, it is easy to create a perfect forgery of a log entry that incriminates an arbitrary user—all that is needed is that user’s current IP address. Researchers at the University of Washington have demonstrated the problem by generating DMCA take-down notices against printers and wireless access points [164], and it is not difficult to imagine the damage a malicious attacker could do. Finally, when a real cybercriminal is caught, he can plausibly deny responsibility by pointing to the weakness of the evidence [99, 135].

In this project, we propose to add a capability to the Internet that provides evidence of a packet's immediate origin. The evidence is strong enough to (1) enable swift and confident action to stop an ongoing attack, (2) convince ISPs to take action against a subscriber who appears to originate offensive traffic, (3) protect falsely accused subscribers, and (4) in the case of a multi-stage attack, provide a starting point for an investigation of the ultimately responsible party. Moreover, our proposed technique protects users' privacy and can be deployed today.

Specifically, we propose a primitive called *packet attestation*, which determines whether a given set of packets was sent over a given access link. Thus, packet attestation corroborates the evidence available today, by verifying that an alleged attack packet originated from the access link associated with the packet's source IP address. The primitive strikes a favorable tradeoff between the strength of the evidence it provides on the one hand, and its privacy implications and deployability on the other.

Attestations preserve user privacy because they only confirm what the querier already knows; a malicious querier would have to guess both the complete payload of a packet and its transmission time to learn anything nontrivial. At the same time, attestations corroborate the circumstantial evidence available about an attack. In case of a negative attestation, they protect users from false accusations. In case of a positive attestation, they give victims and network operators a basis for decisive steps to stop an attack, and law enforcement agencies a basis to start an investigation in serious cases.

Packet attestation does not require changes to the Internet architecture and can be deployed incrementally in the current Internet. Indeed, the technical requirements are surprisingly simple: it is sufficient to add a few strategically placed boxes that record a hash of every packet they see, and that store these hashes for a limited time. Its main overhead is storage; we estimate that a global deployment would require, on average, about 32 commodity hard disks per autonomous system. Packet attestation does *not* require changes to existing core routers, protocols, applications, or end hosts. It also does *not* increase packet sizes or require cryptography on the critical path.

To test whether a deployment of packet attestation would be practical, we have conducted a feasibility study to estimate the cost of a global deployment, and we have built and evaluated a prototype to measure both performance and possible adverse effects on key performance metrics, such as packet loss rate and jitter. Taken together, our results indicate that, at least in principle, a deployment of packet attestation would be feasible today. A paper describing this work is currently under review at SIGCOMM

2011.

5.6 Protecting data integrity with storage leases

This work is led by Ansley Post (now at Google Zürich), with contributions from Paarijaat Aditya, Rodrigo Rodrigues and Peter Druschel.

Society's ever-increasing dependence on digital information underlines the importance of protecting data stored in a digital format. Corporations, government, and other professional organizations are well aware of the need to protect their digital assets, since data loss can directly translate into operational disruption, financial losses, or the loss of irreplaceable cultural and historical artefacts. But individuals also increasingly store valuable digital information, such as personal financial and professional documents, as well as photos, videos, personal letters and other information of substantial sentimental value.

There are many threats to the durability and integrity of digitally stored data, ranging from failure or destruction of storage media and devices, bugs in various parts of the software stack (e.g., device drivers, storage system software, operating systems, or applications), operator error (either intentional or accidental), or deliberate manipulation by malicious intruders or malware running on a machine with access to the data.

Many of these threats can be mitigated by storing data redundantly. For instance, error-correcting codes can mask bit errors, striping or mirroring can mask block or device failures, and backup data copies can enable recovery from catastrophic storage system failures. However, threats from malicious intruders, viruses and worms, software bugs and operator errors may affect *all* copies of stored data that are on-line and writable. Redundancy alone is not sufficient to mitigate these threats.

To protect the data from such threats, backup copies must be maintained in either off-line or write-once storage. Enterprise storage solutions include such archival storage. Typically, data is backed up on tapes that are unmounted after a backup session. Tape cartridges, magneto-optical and ultra-density optical disks offer a write-once option, offering additional protection for existing data while the medium is mounted. Finally, high-end network-attached storage systems can be programmed to accept only the first write to a given storage block (WORM storage [121]), even though the underlying storage is based on conventional disk drives.

However, individual home users and small businesses often do not have the time, expertise, or budget to rely on advanced solutions. High-end solutions like tape robots and network-attached WORM storage systems are fully automated but too expensive; whereas low-end write-once optical disks and magnetic tape drives are inexpensive, but require diligence and regular attention by an administrator.

Diligent users and small businesses might rely on semi-automated backup utilities like Apple's TimeMachine [195] or Microsoft's Backup and Restore Center [206]. While these tools make it relatively easy to create periodic snapshots of the data, the archived snapshots are typically online and writeable. Therefore, the data remains vulnerable to security compromises, operator error or software bugs that cause previously backed up data to be deleted [98].

As part of this project, we develop the idea of *storage leases*. A storage lease enabled device associates each data block with a lease period, during which the block cannot be written. Like off-line or write-once storage, leases protect data from (accidental or intentional) deletion or corruption, without requiring mechanical action and without committing storage indefinitely. Storage leases provide a powerful abstraction, and can be implemented with a small trusted computing base. When leases are implemented within the firmware of a mechanical or solid-state disk, for instance, then its guarantees hold despite errors or compromise in the operating system, or in file or storage system software.

In combination with a backup application, storage leases can bring enterprise-grade data protection to consumers, homes and small businesses. For instance, backup applications like TimeMachine automatically create and maintain period snapshots. By storing the snapshots under a lease for the snapshot's retention period (e.g., daily snapshots kept for a week, weekly snapshots kept for a few months, etc.), the snapshots are protected from software errors, operator mistakes, malware and other system intrusions. Furthermore, storage leases are useful in the context of other applications such as peer-to-peer or cloud storage.

5.7 Policy-based storage

This effort is led by Anjo Vahldiek, with contributions from Paarijaat Aditya, Rodrigo Rodrigues, Johannes Gehrke (Cornell) and Peter Druschel.

Increasingly, individuals and organizations entrust their valuable data to third parties for storage and processing. Individuals use Web email services (e.g., Gmail, HotMail), content sharing sites (e.g., Flickr, YouTube), and storage/backup services (e.g., MobileMe, Windows Live Mesh, DropBox), while organizations use cloud storage services like Amazon S3.

In general, the providers offering these services are reputable companies that can be expected to look after their customers, follow best practices and respect applicable laws. Today, however, customers simply have to trust that (i) they understand the provider's policies; and that (ii) the provider follows these policies and (iii) takes appropriate measures to safeguard customer's data from various threats. A customer has no way to verify what the provider actually does (provider accountability), nor does the customer have a way to prove to their own customers (or interested authorities) how the data is used and safeguarded (customer accountability).

Specifically, customers have no way to answer the following concerns about the data they are entrusting to the provider:

- what measures does the provider take to prevent the loss, corruption or unauthorized use of data due to storage media or device failures, provider software errors, mistakes or malicious activity by provider's employees?
- does the provider's use and storage of data respect the customer's policies, as well as laws applicable to the customer's data?
- how can the customer have confidence that the provider takes appropriate actions regarding customer's data, and how can the customer prove this fact to his own customers or the authorities?

Customers of third-party storage services should have a way to control and verify key aspects of how their data is stored and used, and they should be able to prove these facts to third parties.

Approach: We are exploring the design and use of *policy-based storage* to address this problem. A policy-based storage system is a storage device or enclosure that supports policies. Each data object stored in policy-based storage is associated with a policy provided by the object's originator (i.e., the customer). The storage system allows only accesses to the data object that are consistent with this policy. Moreover, the storage systems issues signed certificates about the properties of the storage system and the policy under which a data object is stored.

Examples of storage system properties include the type, manufacturer, capacity, service life, reliability, media fault events, geographic location and network attachment. Examples of storage policies include the following:

Identity-based access control Allow access only to requestors who present the required identity certificate.

Attestation-based access control Allow access only to requestors who prove that they run a specific hardware/software configuration.

Quota-based access control Allow only a fixed number of read accesses.

Storage lease Allow write access only after a given date.

Time capsule Allow read access only after a given date.

Expiration Allow read access only until a given date.

Guarantees: The guarantees provided by a policy-based storage device depend on the trustworthiness of the storage manufacturer and the correctness of the firmware that implements the access control. A concern regarding the trustworthiness of storage manufacturers is that they might collude with the cloud provider. However, storage system manufacturers are multi-national companies focused on providing storage solutions, and it seems far-fetched to assume that they would risk their reputation by colluding with a cloud provider. A concern about the correctness of the firmware is related to its complexity. However, the storage system's firmware, while of substantial complexity, is still likely to be far smaller and subject to a much lower rate of change than a typical provider system.

We believe that there is substantial value in the additional control and transparency afforded to the customer of a third-part storage service. The customer can control and verify, for instance, that a given number of replicas of the data are stored in diverse locations and in storage systems of a different type, and can control who can access the data under what conditions.

We are currently designing and evaluating prototype implementations of policy-based storage devices, and we plan to submit a first paper on this work to NSDI 2011 or FAST 2012.

5.8 Other work

In this section, we briefly describe a selection of other work that was completed by members of the Distributed Systems group during the reporting period.

Bazaar: Strengthening user reputations in online marketplaces: Online marketplaces are now a popular way for users to buy and sell goods over the Internet. On these sites, user reputations—based on other users' feedback concerning prior transactions—are used to assess the likely trustworthiness of users. However, because accounts are often free to obtain, user reputations are subject to manipulation through white-washing, Sybil attacks, and user collusion. This manipulation leads to wasted time and significant monetary losses for defrauded users, and ultimately undermines the usefulness of the online marketplace.

Ansley Post and Alan Mislove (now at UPenn) proposed Bazaar, a system that addresses the limitations of existing online marketplace reputation systems. Bazaar calculates user reputations using a max-flow-based technique over the network formed from prior successful transactions, thereby limiting reputation manipulation. Unlike existing approaches, Bazaar provides strict bounds on the amount of fraud that malicious users can conduct, regardless of the number of identities they create. An evaluation based on a trace taken from a real-world online marketplace demonstrates that Bazaar is able to bound the amount of fraud in practice, while only rarely impacting non-malicious users [168].

Online social networking research: Ansley Post, Alan Mislove (now at UPenn) and Peter Druschel contributed to some of the research going on in the Networked Systems group (described in Section 7) [199, 200, 157].

6 The Large Scale Internet Systems Group

6.1 Overview

This report covers the period May 2009 – April 2011. This group worked broadly in two areas: Internet routing and private online advertising. In the former, we worked on a number of aspects of global routing scalability and control. The latter represents a new research area, both for the group and for the larger community.

Personnel: The group is led by faculty member Paul Francis, and consists of postdocs Ruichuan Chen and Bin Cheng, PhD students Ekin Akkus and Alexey Reznichenko, and a Research Software Developer Michael Ohlmann. Past members during this review period include sabbatic visitor Zartash Uzmi (Lahore), postdocs Hamed Haddadi and Saikat Guha, and interns Carolyn Hafernik (Penn State) and Ahsan Tariq (Lahore).

Publications and presentations: The group published papers in NSDI [107], the ACM Internet Measurement Conference [106], ACM Hotnets [108], and the eCommerce conference IFIP I3E [110]. Paul Francis gave three keynote addresses, at IFIP Networking 2009 in Aachen Germany, Euromicro SEAA 2009 in Patras Greece (both in Internet routing scalability), and ICCCN 2010 in Zurich (in private advertising). He gave invited talks at Colorado University, Kentucky University, Cambridge University, Telefonica Labs, the NANOG network operator's meeting, Denver University, Eurecom, ETH, KAIST, Peking University, Tsinghua University, Tokyo University, Seoul National University, and NTT Research Labs.

External collaborations: The group collaborated with Huawei research labs in Peking, AT&T research in New Jersey, and the TU-Kaiserslautern CS Dept.

Service: Francis served as the Managing Director (roughly equivalent to department head), and as the chair of the ISTC (the committee that oversees IT for both MPI-SWS and MPI-INF). Francis served on the CCS11, NSDI11, and Hotnets11 program committees.

6.2 Internet routing

The scale of global Internet routing, in the form of the routing protocol BGP (Border Gateway Protocol), has been a problem for years. It is handled in practice through a combination of brute force and deployment compromises. On the brute force side, router vendors like Cisco engineer large amounts of very fast memory into their hardware in order to accommodate some years of future growth. The deployment compromises ultimately limit the speed at which BGP converges to correct routes, and limits the number of enterprise networks that can multi-home to the Internet. Multi-homing, where an enterprise connects to multiple ISPs, is important for reliability. So, in the end, the scale of BGP reduces the reliability of the Internet for many users.

Much of the networking research community is following the lead of the USA NSF, and is working on clean-slate solutions to the routing scalability problem. In the extreme, what this means is that researchers ignore the installed base, and design protocols as though they will get to deploy these protocols from scratch. While much of this work is very interesting, we believe that it is highly unlikely that any of it will ever be deployed. Therefore, we choose to work on what we call "dirty slate" approaches. This means that we constrain our solutions to those that not only can be deployed as small incremental changes to existing equipment or protocols, but critically, to those solutions for which there is an immediate economic pay-off. In other words, we are only interested in solutions that might actually get deployed.

Our work in Internet routing consists of two research projects and one tech transfer project.

Virtual Aggregation Tech Transfer: The tech transfer effort, led by Paul Francis, is to standardize a routing technique called "Virtual Aggregation" in the IETF. Virtual Aggregation reduces hardware routing table size by as much as ten times, and can be deployed with only configuration changes to routers (no software changes). The research side of this work was done at Cornell before coming to MPI-SWS. The Chinese telecommunications vendor Huawei took an interest in this work, but preferred a solution that allowed for simpler configuration even if it means that router software changes are required. Working closely with Huawei researchers, we extended Virtual Aggregation along these lines, and produced the required standards documents (Internet Drafts in the IETF). We have been able to push this work very close to final Informational RFC status in the IETF. In all, this effort required thrice yearly attendance at the IETF, and roughly a dozen drafts spread over three separate IETF documents [91, 93, 92]. We expect to complete this effort by late 2011. Although this tech transfer is a substantial amount of work, we feel it is important both for insuring that our research has impact, as well as for staying abreast of real Internet infrastructure problems and informing future research efforts.

Address-based Route Reflection: A second project, led by Ruichuan Chen and working with researchers from AT&T, works to solve the long-standing problem of looping and instability in BGP Route Reflectors (RR). These problems have existed in BGP ever since the technique of route reflection was introduced into BGP in order to deal with scaling problems. In practice, ISPs can avoid these problems through careful engineering and configuration of their RR overlays. There are two problems with this status quo. First, ISP topology design has evolved over the years as MPLS layer-two tunneling becomes widespread, and ISPs introduce techniques like traffic engineering and fast failover. The old conventions for RR engineering are not a good match for these new tunneled infrastructures. Second, there is always a danger that misconfigurations will trigger loops and instabilities.

At the root of the problem is the fact that RRs filter routing information as it passes through them. As a result, different RRs in an ISP receive different routing information, and can therefore make inconsistent decisions. This is a departure from the original BGP invariant that all BGP routers have the same information. We solve this problem through a simple change in the way RRs partition work. Current RRs partition work according to topology: routers send routing information to nearby RRs, which pass them on to other RRs. In our approach, we partition work according to address ranges. As a result, each RR sees complete routing information for some fraction of the address range, and can therefore make consistent decisions.

Our approach scales better for virtually all measures, and provably eliminates looping and instabilities. This work, the research leg of which is largely finished, has been submitted to SIGCOMM. Looking forward we plan to publicize the results among the Internet operational and standardization communities.

Routing Table Compression: A third project, initiated and led by visiting Prof. Zartash Uzmi and also working with researchers from AT&T Research, explores techniques for compressing the size of the Internet Routing Table without changing the external behavior of routers. The problem is that the hardware forwarding table in routers is constrained to hold only a certain number of entries. If the routing table grows larger than this number, then the routers must either be upgraded at significant expense, or the routers must be repurposed into roles where a full routing table can be avoided (i.e., customer edge routers). Roughly speaking, our basic solution exploits cases where there are adjacent address spaces whose physical next hop is the same, even though the paths to the destinations are different. The routing table entries for these adjacent spaces can be compressed into a single entry in the hardware forwarding table.

Previous work had shown how to do optimal compression for a given routing table, but required that the full table be re-compressed with each change in the routing table. As a result, the approach was not practical. In our work, we designed incremental algorithms and proved their correctness.

The incremental approach is not optimal, and indeed slowly drifts away from the optimal with each new update. As a result, from time to time the full table must be recompressed. We worked through the engineering challenges required to make our approach practical. Using data from AT&T, we showed that our approach can reduce forwarding table size by roughly 50% without any external change to the router behavior. This can extend the lifetime of routers easily by several years. Perhaps more importantly, it can give the installed base some breathing room should the exhaustion of IPv4 addresses result in increased growth of the routing table.

This work has been submitted to SIGCOMM, and we are in the process of standardizing it in the IETF.

6.3 Private online advertising

This is a new research area, both for Paul Francis' group and for that matter the research community. In this work, we address the increasing amount of user privacy loss being caused by the advertising industry. This loss takes primarily two forms. First is the privacy loss due to user tracking: i.e., the use of cookies to track user web pages visits and user searches. This tracked information goes not only to the aggregators that initially collect it, but also to third party companies that purchase it. Second is the privacy loss to advertisers due to demographic targeting (i.e., targeting advertisements to gay users of Facebook).

Initially our research focused on the first problem: user tracking. We designed an advertising system called Privad that eliminates the need for tracking by keeping user profiles strictly within user computers [107, 108, 110]. This is done through client software running on user computers. In our model, users are kept anonymous from ad networks through third-party proxies. Encryption prevents the proxies from learning anything about users. The client software profiles users, and requests ads for user interest categories. The user interests are kept separate and unlinkable, preventing the ad network from building up unique user profiles. For each interest category, the ad network transmits ads across multiple demographics. The client software selects the appropriate ads for the user, and displays these locally. Reports of clicks and views are also anonymous and unlinkable.

Our goal is to build a practical system that will serve as a realistic and attractive alternative to tracking. To do this, we needed to take into account a number of practical issues such as click fraud and auctions, as well as insure that current business models (i.e., pay-per-click and pay-per-view) are accommodated.

We have spent much of the last year building a working system that we plan to deploy 2nd quarter 2011. The client is a Firefox add-on, and we expect to deploy it to thousands of users by piggy-backing it on existing add-ons (and paying the developers of these add-ons for the privilege: roughly $0.10 per user per year). In this deployment, we will use the shopping.com and amazon.com product APIs as a source of advertisements. Specifically, when a user does a search on a shopping website (as identified from a list of roughly 30K catalogued sites), the search is transmitted to our broker via our proxy as an "interest". This search is then given to shopping.com or amazon.com, which returns a set of product descriptions matching the search. These descriptions are turned into Google text ads, and over time shown to the user in normal Google ad boxes. By showing users ads that match their interests, we hope to demonstrate effective user targeting while preserving user privacy.

As part of our research in private advertising, we did a measurement study to try to determine the amount of demographic targeting being done today by Google and Facebook [106]. Surprisingly, we were unable to find much targeting being done by Google. Not surprisingly, we did find that Facebook does substantial amounts of demographic targeting. We discovered, however, that this targeting can be exploited to reveal private and sensitive information. Specifically, Facebook advertisers can discover that Facebook users are gay even when Facebook users use privacy controls to limit knowledge of their sexual preference to friends only. Worse, we found instances where advertisers were targeting gay users in this way with ads that had no overt gay theme (i.e., a nursing school in Florida). As a result, when clicking on such an ad, the user would have no clue that he or she is revealing sexual preference to the advertiser. Even before publication in IMC, this aspect of our worked was picked up by the popular press. For a few days, this was a front-page story on every major US new website (CNN, NY Times, NPR, Fox, MSN, etc.).

It is exactly this kind of problem that the second phase of our research hopes to address. We believe that the key to the solution lies at the interface between the advertiser and the broker. Today, brokers allow advertisers to target specific demographics (age, gender, location, salary, sexual preference, etc.). As a result, advertisers know something about users when they click on ads. In general, however, advertisers don't explicitly care about a user's demographics...what they care about is predicting whether a user is likely to be interested in a given product or service. If we can identify effective ways to identify user interests without invoking demographics, then we can remove user demographics from the broker/advertise interface.

There is substantial evidence that we'll be able to do just this. Of course, product searches can directly identify user interests. Beyond this, amazon.com for instance uses an item-by-item style of recommender when suggesting that "customers who browsed for these items also browsed for ....". This type of recommender is demographically adnostic, and can be implemented in a privacy-preserving fashion. This suggests a broker/advertiser interface whereby the advertiser supplies the broker with 1) the ad, and 2) the landing pages (the pages the user goes to after a click), and 3) other products or services similar to that being advertised. The broker uses this information to associate the advertised product or service with user interests. The broker also serves the landing page. If after seeing the ad and the landing pages, the user continues with a purchase, then the user knows what personal information is being revealed to the advertiser. For instance, if the user buys a tennis shoe, the user knows he or she is revealing an interest in tennis. Likewise if a user signs up for a gay dating site, the user knows that he or she is revealing that he or she is gay.

This research has a number of challenges we hope to address in the coming year: Can this style of recommender be done privately and scalably? Is it effective? What (if anything) is lost by giving up demographic targeting? Do we at least need to allow targeting of location? What if the advertiser wants to use demographic information not for targeting, but for determining the best "creative" or "pitch"? Can this be done while preserving privacy?

7 The Networked Systems Group

7.1 Overview

This section describes the activities of the Networked Systems group between May 2009 and April 2011. The group's research focuses on understanding and building *complex* networked systems. In particular, we tackle the challenges posed by the growing *complexity* (i.e., scale, heterogeneity, decentralized control, and dynamic evolution) of today's networked systems. The goals of recent projects include (a) making Internet access infrastructures transparent, (b) enabling efficient and cost-effective bulk content delivery in the Internet, (c) understanding the dynamics of online social networks, and (d) leveraging social networks to design better information sharing systems.

Personnel: The group is led by Krishna Gummadi. It is currently comprised of five graduate students (Massimiliano Marcon since July 2006, Bimal Viswanath since November 2008, Farshad Kooti from October 2010, Mainack Mondal from October 2010, and Juhi Kulshetra from April 2011).

Marcel Dischinger graduated with a PhD in October 2010. He took a position as a senior software engineer at Barracuda Networks. Meeyoung Cha spent two years as a post-doc with the group. She took a position as an assistant professor at KAIST, Korea in September 2010. Alan Mislove, a PhD student from Distributed Systems group, was co-advised by Krishna Gummadi and Peter Druschel. Alan took a position as an assistant professor at Northeastern University in Fall 2009.

Collaborations: The group members have close collaborations with the distributed and dependable systems groups led by Peter Druschel and Rodrigo Rodrigues, respectively. External collaborators include researchers from Microsoft Research (Stefan Sariou, Emre Kiciman, and Ratul Mahajan), UCSD (Amin Vahdat), Northeastern (Alan Mislove), AT&T (Balachander Krishnamurthy) and Telefonica (Pablo Rodriguez and Nikolaos Laoutaris).

Publications: The group members regularly publish their research in the top conferences and workshops in their field. Members have co-authored papers at NSDI [73], SIGCOMM [200], WSDM [157], ICWSM [59, 20], NOSSDAV [146], HotCloud [183] and WOSN [199].

Software, tools, and data: The group strives to make its software, tools, and data sets publicly available to the extent possible. To date, over 400 research groups at universities and research labs worldwide have used the data sets we gathered as part of our measurement studies of online social networks. Over the last two years, more than 400,000 end users worldwide have used the Glasnost software we designed to test the traffic management policies of their access ISPs (e.g., cable and DSL providers).

Press: The group's research results have also been covered widely by the popular press. Articles describing the group's social network research have appeared in numerous popular news media and technology blogs worldwide including the New York Times, Harvard Business Review, MIT Technology Review, New Scientist, Wired magazine, Slashdot, Businessweek, Max-Planck Research magazine (Germany), Sueddeutsche Zeitung (Germany), Science TV (Korea), and MTV (Brazil). The group's research on making Internet access network infrastructures more transparent continues to attract the attention of popular technology blogs like Slashdot. Previously, the findings of the Glasnost project (on which ISPs are blocking BitTorrent) were extensively reported in over 300 popular news media worldwide, including the Associated Press, WSJ, and the New York Times.

Technology transfer: The group's research continues to attract considerable attention from industry and policy makers. *Measurement Lab* [75], an open platform for researchers to deploy measurement tools, founded by the group members in collaboration with other academic and industry partners, currently hosts several research projects targeting a transparent Internet. The group members are also in discussions with FCC, the telecom regulator in the US, about the possibility of making Glasnost tools available on the FCC managed website, broadband.gov. Additionally, Glasnost results continue to be referenced by several telecom regulators during policy discussions, including regulators in the US and Europe.

Teaching and invited talks: Krishna Gummadi co-taught (with Rodrigo Rodrigues) an advanced graduate systems seminar course at Saarland University in Winter 2010. Krishna also organized the SWS systems seminar in Summer 2010. In addition, Bimal Viswanath served as a teaching assistant for the undergraduate Operating Systems course at Saarland University in 2009.

Krishna Gummadi has given invited talks at the International Workshop on the Social Web organized by KAIST in Korea, the Workshop on Social Networks and Distributed Systems held in conjunction with PODC 2010, the 2020 Networking Summit organized by Cambridge University and Telefonica Research in Barcelona, and the Winter School on Hot Topics in Distributed Systems organized by INRIA in France. He also gave a tutorial on Online Social Networks at SIGMETRICS 2009.

Service: Krishna Gummadi has served on the program committees of SIGCOMM 2009, HotNets 2009, WOSN 2009, WWW 2010, IMC 2010, CoNext 2009 and 2011, NetEcon 2011, and DBSocial 2011. In addition, he served as the guest editor of the special issue of JSAC on Measurement of Internet Topologies.

Within MPI-SWS, Krishna served on the graduate student admissions committee in 2009 and 2010, and ran the MPI-SWS internship program from 2009 to 2011. Outside MPI-SWS, Krishna serves on the Measurement Lab's [75] steering committee, which is responsible for developing the platform's structure and organizational policies.

7.2 Research agenda

The group is interested in understanding and building complex networked systems. Our recent research is motivated by the growing complexity of emerging networked systems, such as the social Web, Internet access network infrastructures, and cloud computing platforms. In this research statement, we will first explain why studying complex networked systems is hard and then describe our research approach. Later we will describe contributions from our studies of three emerging complex networked systems and their impact within and outside the research community.

7.2.1 Approach

The complexity of large-scale networked systems like the Web or the Internet is characterized by their (a) *massive scale* - they are comprised of a large number of components, (b) *tremendous heterogeneity* - the components that comprise the systems are very different from one another, (c) *decentralized control* - the components are owned and managed by multiple independent entities with potentially different goals, and (d) *dynamic evolution* - the systems evolve as their components change over time. Compared to traditional stand-alone systems like PCs or mainframes, complex networked systems pose new research challenges.

First, due to their massive scale and tremendous heterogeneity, it is impossible to predict the overall behavior of complex systems from their fundamental design principles alone. So an important challenge when studying a complex system lies in understanding how the system works in practice. To understand how a system works, we gather detailed measurements of the system in real-world deployment. We then analyze the measurements and construct models to explain how the overall (macroscopic) system properties emerge from the interactions of individual (microscopic) components that comprise the system.

Second, the decentralized nature of control of the system components implies that the inter-component interactions will be driven by the selfish interest of the component owners. When designing complex systems, we account for the economic or social incentives of their participants.

Third, complex systems are never static; the components that comprise them constantly evolve. Managing the evolution of deployed complex systems poses another difficult challenge. We use measurements and analysis to identify the system invariants as well as the trends in component evolution. We leverage the insights so gathered to propose new system designs. We implement and deploy new system designs in practice, completing the feedback loop essential to managing constantly evolving systems.

In summary, we use a combination of measurement, analysis, design, and deployment methodologies to understand and build complex networked systems.

7.3 Enabling the social Web

Recently online social networking sites, such as Facebook, YouTube, and Twitter, have become tremendously popular. Users join these sites to connect with other users and share content. With user populations running into hundreds of millions, these sites herald the emergence of the social Web. Several key trends distinguish the social Web from the traditional Web. They include democratization of content publishing allowing individuals to share user generated and personal content (e.g., users sharing family photos and videos over Facebook), collaborative ranking and filtering of content to help users search for useful content (e.g., users rating videos on YouTube), and word-of-mouth based discovery and dissemination of information (e.g., users propagating information to their friends in Twitter). Despite the popularity of the social Web, it is still in its infancy and existing social Web systems suffer from a number of shortcomings. For example, sharing personal content over social networking sites today raises severe privacy concerns, collaborative content ranking is susceptible to manipulation by malicious users, and few understand the dynamics of word-of-mouth based content propagation.

Our long-term research goal is to enable the social Web, i.e., to help the social Web achieve its full potential. To engineer a better social Web for tomorrow, one needs to conduct systematic studies of the current social Web to understand its strengths and weaknesses. Since the social network is central to the functioning of the social Web, it is also important to leverage relevant insights from the fields of sociology and network theory when designing social Web systems. Accordingly, we have adopted a methodology that ranges from conducting measurement studies of the social Web to leveraging the insights from sociological studies and network theories to build and deploy practical social Web systems. Below we highlight our key contributions.

In the past, we conducted the first large-scale measurement study and analysis of the structure of multiple popular online social networks, namely Flickr, YouTube, LiveJournal, and Orkut [155]. We also studied link formation in different social networks to better understand how the networks evolve to exhibit many common structural properties, e.g., power-law degree distributions [154]. Further, we studied how the strength of social links, measured by the amount of activity or interaction between users connected by the links, evolves over time [199].

More recently, we focussed on leveraging the data collected in our measurement studies as well as the insights obtained from our data analysis for designing new systems and evaluating them.

(a) Leveraging social networks to build better social Web systems: Compared to the traditional Web, finding useful content in the social Web poses new challenges. In the former, explicit links called hyperlinks between content (typically web pages) are leveraged by search engines to rank or estimate the relevance of content for a search query. For example, Google's popular PageRank algorithm exploits the link structure of the Web to rank relevant and trustworthy content highly. In contrast, explicit links are by and large non-existent between the content on the social Web. This is in part because a large fraction of the social Web content is non-textual, for example photos and videos, which are hard to be linked to each other. Further, most social Web content is generally of short-term interest and users publishing the content do not bother to connect them to other related content. As a result, users rely on feedback of other users to rank and filter content. For example, videos matching a YouTube search query are ranked based on the number of user views or comments or average user rating. Similarly, videos reported as spam by users are filtered from the results. Leveraging both implicit (e.g., number of views) and explicit (e.g., users' rating) feedback from a large community of content consumers allows content in the social Web to be rated rapidly.

The reliance on feedback of other users, many of them unknown or unrelated to the user issuing the query raises concerns about relevance and trust. For example, a user might be interested in the feedback of other users with whom the user shares similar interests rather than the feedback of arbitrary users in the system. Further, content rating systems are vulnerable to a class of attacks called Sybil attacks, where a few malicious attackers can create a large number of fake user accounts in the system and manipulate content voting to promote spam. Such attacks have been successfully launched in the recent past against popular social networking sites like YouTube and Digg.

One promising approach is to leverage the link structure of the social network connecting the users to identify groups of users that share similar interests and trust, similar to the way search engines use the link structure of the Web to find useful content. The key insight here is that links are typically formed between users who either share some common interests or are familiar with (trust) each other. Below, we describe our attempts to build systems leveraging social networks to identify relevant and trustworthy sources of information.

Leveraging social networks to identify users with similar interests: In the past, we designed a system called PeerSpective to test the hypothesis that content accessed by a user might be relevant (of interest) to the user's friends [153]. PeerSpective exploits a user's browsing history to improve the search results for the user's friends in a social network. We deployed a prototype of PeerSpective within our institute and our results showed that a link between two users in a social network indicates a certain degree of similarity in interests between the users,

More recently, we focused on inferring groups of users with shared interests by analyzing the social network's structure. Our hypothesis is that groups of nodes that form tightly connected communities in the network graph are likely to share some similarities in interests. However, we found that existing community detection algorithms do not work well over social network graphs like Facebook. We designed a new algorithm that works quite effectively over some (though not all) real-world social networks [157]. In ongoing work, we are investigating better ways to identify communities of users with similar interests by leveraging (a) insights we gained from analyzing social network structures and (b) insights from sociology about the different types of motivations that lead users to form communities.

Leveraging social networks to identify trustworthy users: Recently, researchers have proposed a number of schemes that attempt to defend against Sybil (multiple identity) attacks by leveraging social networks. They are based on the observation that while Sybil attackers can create arbitrarily large number of fake identities, it would be hard for the attackers to establish a large number of links to honest users in the system. As a result, Sybil nodes tend to be poorly connected to the rest of the network, compared to the non-Sybil nodes. Most existing Sybil defense schemes rely on this observation and analyze social networks' structure to identify Sybil nodes and block them from interacting with non-Sybils. We refer to them as Sybil identity detection schemes.

We studied Sybil identity detection algorithms and their performance over real-world social networks [200]. Our analysis revealed that the effectiveness of Sybil identity detection schemes is limited over real-world social networks that exhibit well-defined community structures. So in follow-up work, we proposed a fundamentally different approach to leveraging social networks against Sybil attacks: Sybil tolerance [198]. Rather than focus on identifying nodes as Sybils, Sybil tolerance focuses on designing systems that strictly bound the impact of Sybil nodes. We proposed a general methodology for designing Sybil tolerant systems using credit networks, which provide a way to model trust between nodes in a social network and support payments between arbitrary nodes. To demonstrate the effectiveness of our approach, we designed and evaluated two Sybil tolerant systems: (1) Ostra: a system that leverages social networks to thwart spammers [156] and (2) Genie: a system that leverages social networks to thwart large-scale crawls of social networking sites [198].

(b) Developing a better understanding of word-of-mouth based information discovery and propagation: Word-of-mouth is a unique and popular way to discover content on the social Web. Information can propagate from one user to another user along links in the social network by word-of-mouth. The popular Twitter social network epitomizes this trend. Hundreds of millions of tweets (short messages) are posted by millions of users every day to their followers and some of the tweets are in turn forwarded to other users causing a word-of-mouth dissemination of information. Recently there has been a lot of excitement about applying word-of-mouth techniques to marketing, political, and public information campaigns.

Despite the considerable buzz surrounding word-of-mouth based information propagation, little is known about the dynamics of word-of-mouth propagation. While sociologists have long proposed theories about word-of-mouth propagation, few have been empirically verified or derived. It has proven hard to conduct rigorous and large-scale studies of how information spreads in a society using conventional methods like user surveys. On the other hand, the popularity of online social networking sites and the public access to information on these sites provides a unique opportunity to study word-of-mouth based information dissemination at an unprecedented scale. Developing a better understanding of word-of-mouth propagation is also the first step towards engineering better viral advertisement campaigns in the future.

In the past, we conducted some of the earliest studies of information dissemination in online social networks by collecting and analyzing large-scale traces of photo dissemination in the Flickr social network [61]. To study word-of-mouth based information propagation at a larger scale, we gathered large-scale measurements from the Twitter social network with explicit permission from the site operators. The data we gathered includes over 1.7 billion messages exchanged and propagated by over 50 million Twitter users. We found that nearly a third of all tweets contain a URL (link) to Web content. Such word-of-mouth based Web content discovery has become a major driver of traffic to many websites today. We used the data to characterize several aspects of the newly popular word-of-mouth phenomena including its impact on URL popularity, its dependence on users with a large number of followers, its effect on the diversity of information discovered by users, and the structure of content propagation paths in the network [173].

We also used the data to study the role or influence of individual users in propagating the information in the social network [59]. We analyzed various measures of a user's influence and investigated the dynamics of user influence across topics and time. Our study suggests that topological measures such as in-degree alone reveals very little about the influence of a user. Further, we find that user influence is not gained spontaneously or accidentally, but through concerted effort such as limiting tweets to a single topic. Our findings have important implications for people organizing viral marketing campaigns.

More recently, we used the data to characterize the news media landscape in Twitter [20]. In particular, we studied how users in Twitter are exposed to news from a larger and diverse set of media sources by the way of tweets forwarded by users they follow. Our analysis shows that media organizations reach a considerably larger audience through indirect exposure via social links. Similarly, we found that indirect exposure increases the diversity of political opinions seen by users: between 60-98% of the users who directly followed media sources with only a single political leaning (left, right, or center) are indirectly exposed to media sources with a different political leaning.

(c) Addressing the privacy crisis in the social Web: Unlike the traditional Web, where content publishing is dominated by organizations like companies, universities, governments, the social Web is fueled by content published by individual users. A lot of the content published and shared on the social networking sites tends to be personal in nature (e.g., photos and videos of family and vacations), which raises severe privacy concerns. The concerns vary from unintended exposure of personal information to one's friends causing social embarrassment to leakage of personal data to companies that might misuse the data violating the privacy of individual users. The privacy concerns are real; not a day goes by without a new incident of privacy violation using data on the social Web being reported in major news media. Our goal is to design systems that alleviate users' privacy concerns when sharing personal content and interacting with other users on the social Web.

Regaining control over data sharing from social networking sites: Today users share their privacy-sensitive personal data (e.g. family videos) by uploading it to data centers owned and managed by social networking sites like Facebook. In the process, they lose control over their data sharing. The site operators not only place restrictions on the type of data that can be shared but they also share ownership of the copyrights to the content and retain the right to change their privacy policies unilaterally, without requiring users' consent.

We designed the Stratus system to enable users to regain control over their data sharing [146]. In Stratus, users share their data directly from their home networks, which are under their control. We built inexpensive, low-power, always-on personal data servers at home using modern residential gateways and used them to share personal data. We argue that recent trends, such as the availability of large, inexpensive storage devices and always-on, high-speed broadband connectivity, bode well for a future where data sharing is done from homes. We deployed Stratus in 10 households and showed that Stratus can deliver social content directly from end user homes with good availability and performance.

Managing privacy in social networking sites: The growing popularity of content sharing over social websites requires end users to be content managers. Today, for every single piece of data shared on sites like Facebook – every wall post, photo, status update, friend request, and video – the uploader must decide which of his friends, group members, and other Facebook users should be able to access the data. Given the per-user average of 130 friends and 80 groups – compounded with the average 90 pieces of content uploaded per user per month – it is unsurprising that we are in the midst of a privacy management crisis, wherein the task of simply managing access to their content has become a significant mental burden for many users.

In ongoing work [109], we are exploring several technological approaches to address the privacy management crisis in Facebook: First, we are building an application that makes it easier for people to understand who can see their data. Second, we have implemented a mechanism that can automatically infer groups among a user's friends by analyzing the structure of the social network. These groups can be used to share data with a subset of the user's friends. Finally, we argue that researchers and developers should shift their focus from setting access controls (i.e., which users have access) for a data item to managing exposure of (i.e., which users have seen) the data item. For example, even though many users may have access to a piece of data, most may not view it, meaning the information is not widely exposed. It may even be possible to infer the expectation a user has about his or her data exposure from looking at the past history of accesses to the user's data or those of similar users. For example, if there are a significant number of requests that show unusual access patterns (e.g., an avalanche of requests to old data posted by the user several years ago or from others who are far away in the network), then it may be worth warning the user as well.

Impact: Our results were published and presented at top conferences and workshops (less than 10-20% acceptance rates) in the fields of Internet measurements, networking, distributed systems, social media, and Web studies. The papers appeared at SIGCOMM [200], which is the top conference in the broad area of networking, at WOSN [199], which is a top workshop for online social networking research, at ICWSM [59, 20], which is a top conference for social media studies, at WSDM [157], which is a top conference in the areas Web search and data mining.

More broadly, our papers in the area represent the first studies either analyzing the social Web or proposing new system designs for the social Web. Consequently, they are widely cited by other researchers in the community. For example, to date, our study analyzing online social networks' structure [155] has been cited 445 times (according to Google Scholar) since its publication three years ago. Further, 8 of the 12 papers we published before 2010 [61, 199, 154, 60, 156, 151, 155, 153] have each received 40 or more citations, while the 3 papers published in 2010 [200, 157, 59] have, to date, received 6, 14, and 29 citations respectively.

These results were also covered in numerous popular news media and technology blogs worldwide including the New York Times, Harvard Business Review, MIT Technology Review, New Scientist, Wired magazine, Slashdot, Businessweek, MaxPlanck Research (Germany), Sueddeutsche Zeitung (Germany), Science TV (Korea), and MTV (Brazil).

7.4 Bringing transparency to Internet access networks

A large and rapidly growing proportion of users connect to the Internet via broadband access networks such as residential DSL, cable, and cellular networks. Broadband access networks are often the bottleneck in the last mile of today's Internet. Their characteristics critically affect Internet applications, including voice-over-IP, online games, and peer-to-peer content sharing/delivery systems. Despite their widespread deployment, broadband access networks today are notoriously opaque to researchers, system designers, and end users. Not only are most access ISPs hesitant to reveal details about their network deployments, but the massive scale and heterogeneity of their network infrastructures present a very challenging environment for researchers studying them.

Our research goal was to make Internet access networks more transparent to researchers, system designers, and end users. Transparency has many uses: consumers can make a more informed choice when selecting their ISPs, applications can be adapted to work better under the network policies, and regulators can monitor ISPs and hold them accountable, if they deviate from their contractual obligations

To improve network transparency, we built novel measurement tools, testbeds, and systems. We deployed them in practice to measure commercial access network deployments at scale and analyze the performance of distributed applications over these networks. What makes this research challenging is that the tools, testbeds and systems have to work with little to no cooperation from ISPs managing the networks. Below we highlight our key technical contributions towards improving access network transparency.

In the past, we designed tools to measure broadband access networks and the performance of transport protocols over them [113]. We used the tools to conduct the first large-scale measurement study of major cable and DSL ISPs in North America and Europe and to study the performance of several transport protocols over the Internet at large. Our findings revealed important ways in which residential networks differ from how the Internet is conventionally thought to operate [72]. We proposed and prototyped a new Internet testbed design called SatelliteLab that enables researchers to evaluate their systems over broadband access networks [71].

More recently, we designed and deployed a widely-used network measurement platform called Glasnost that enables end users to accurately verify whether their access ISPs are shaping (i.e., rate-limiting, blocking, redirecting) traffic generated by their different applications [74, 73]. ISPs are increasingly deploying a variety of middle-boxes to monitor and to manipulate the performance of user applications. Most ISPs do not disclose the details of their network deployments even to their customers. With Glasnost, end users can connect to one or more of the geographically distributed Glasnost servers and run controlled experiments to accurately establish whether an ISP is discriminating against a particular type of traffic. The key to Glasnost's design lies in conducting flows belonging to different applications under nearly identical (controlled) network environmental conditions and comparing their performance.

We have been maintaining a public deployment of the Glasnost platform since March 2008. To date, more than 600,000 users around the world have used Glasnost to verify their ISPs' traffic shaping policies. Our results show that Glasnost enables end users to discover previously undisclosed traffic shaping policies of their ISPs and that data from Glasnost can be reliably used to verify whether ISPs are meeting their contractual obligations.

Impact: Our findings had a significant impact both within and outside the research community. Our research results were published at NSDI, which is a top conference in the area of networked systems [73]. The data sets gathered as part of the study have been used by several tens of research groups around the world. The public deployment of the Glasnost system has been used by over 600,000 end users from over 3,000 ISPs in 157 countries worldwide to test their ISPs' traffic shaping policies. Further, Glasnost results were widely reported in the popular news media, including the Associated Press, WSJ, New York Times and over 300 other newspapers worldwide, raising consumer awareness about traffic shaping policies deployed by ISPs. The results were also used by telecom regulators responsible for monitoring ISP practices in Europe, North America, and Asia. Interestingly, the three largest ISPs, which were identified by Glasnost as blocking BitTorrent application traffic, have since stopped the practice, suggesting that greater transparency into ISPs' traffic management policies can help hold ISPs accountable.

The success of the Glasnost platform catalyzed the establishment of MeasurementLab [75], an open platform for researchers to deploy Internet measurement tools. MeasurementLab is a collaborative effort by academic researchers, government regulators, and industry partners like Google. Today it hosts several research projects targeting a more transparent Internet.

7.5 Building trustworthy cloud computing infrastructures

The computing industry is witnessing a paradigm shift towards cloud computing, where regular users and enterprises outsource their data and computation to remote data centers operated by cloud service providers like Google, Microsoft and Amazon. While many economic and technological factors are driving this trend, the current cloud computing model suffers from a serious drawback: customers are expected to entrust their data to cloud providers with little say over how their data is managed. The lack of control over how data is handled in the cloud is a big deterrent for potential customers of cloud services. In particular, the possibility of losing confidential corporate data, and the need to ensure that data stays within national or other jurisdictional boundaries for regulation compliance reasons are often cited as the main concerns with the use of cloud computing. Below we describe how we plan to address this problem.

To improve customer trust in cloud computing, we advocate a cloud computing model in which (a) customers specify policies defining the requirements for hardware and software that is allowed to manipulate their data and (b) cloud providers collaborate with customers to ensure that the policies are enforced. To construct cloud services that adhere to this model, we proposed a new abstraction, called policy-sealed data, where data is sealed to a policy (i.e., encrypted) and can be unsealed (i.e., decrypted) only by nodes whose configurations match the policy. We designed a system called Excalibur that implements policy-sealed data in cloud infrastructures. To demonstrate that Excalibur is practical, we used it to extend the open-source Eucalyptus cloud management software. Our experiments showed that Excalibur scales well and that our extension to Eucalyptus incurs a modest overhead compared to its base version, while providing customers with better protections regarding how their data is managed.

Impact: We published a position paper outlining our vision for building trustworthy cloud computing infrastructures at HotCloud 09 workshop [183]. This paper has been cited 47 times so far and was rated as one of the 5 cool computing projects by the CIO magazine in 2009. We have a full paper on the Excalibur system under submission [185].

8 The Rigorous Software Engineering Group

8.1 Overview

Personnel. The Rigorous Software Engineering Group research group is led by faculty member Rupak Majumdar, and consists of postdoctoral scholar K.C. Shashidhar, graduate student Zilong Wang, and visiting student Zhenyue Long. This group started at MPI-SWS in July of 2010, when Majumdar moved to Kaiserslautern. Shashidhar, Wang, and Long joined in October. In addition, the group collaborates with Majumdar's graduate students Indranil Saha, Manu Jose, Sai Tetali, and Majid Zamani at UCLA.

Publications. In the last year, we have published in top conferences in several different areas of computer science. Main publications include: programming languages (PLDI 2010 [84], PLDI 2011 [128]), verification (CAV 2010 [96]), hybrid and embedded systems (RTSS 2009 [145], EMSOFT 2010 [21], HSCC 2011 [144, 178]), and VLSI design automation (ICCAD 2010 [127], DAC 2010 [126]).

External Collaborations. We collaborate actively with research groups at IMDEA, Spain (Pierre Ganty), Intel Strategic CAD Labs (Amit Goel, Sava Krstic), IST Austria (Krish Chatterjee), TU Munich (Andrey Rybalchenko), University of California, Los Angeles (Lei He, Todd Millstein, Jens Palsberg, Paulo Tabuada), and University of California, San Diego (Ranjit Jhala),

Awards. Majumdar received a Sloan Foundation Fellowship in 2010. The paper [21] won the ACM SIGBED Best Paper Award at EMSOFT 2010. The paper [126] received a best-paper nomination at DAC 2010 (but did not win). The papers [96, 127, 63] were invited to special journal issues after the conference.

Service. Majumdar is the chair of the faculty recruitment committee. He designed and helped implement a web-based application for managing faculty, post-doc, PhD, and intern applications to the institute.

8.2 Research agenda

The goal of our research is to provide algorithmic and tool support for better construction and analysis of complex software, hardware, and mixed systems. In the past year, the group has focused on four main directions:

1. Algorithms for the end-to-end verification of embedded control systems (publications in RTSS 09, EMSOFT 10).

2. A theory for robust cyber-physical system design (publication in HSCC 11).

3. Tool support for the development and verification of embedded control systems (publications in MEMOCODE 10, HSCC 11).

4. Parameterized verification of software protocols (publications in CAV 10, PLDI 10).

We now describe these directions, and our main results, in more detail.

8.3 Design and verification of embedded control applications

In model-based development of embedded control applications, one starts with a mathematical model of the system to be controlled (the "plant"), and uses techniques from control theory to design a controller that ensures that the closed-loop system consisting of the plant and the controller satisfy design requirements such as stability and performance. The controller is then implemented in software and hardware, by mapping the control computations onto platform resources and writing code that implements the control laws as well as co-ordinates the behavior of different tasks in the system. In an ideal scenario, one verifies properties of the system at the model level, and uses a code-generator to automatically implement the system.

In practice, the link between models and code is not so simple. The mathematical model for controller synthesis abstracts many aspects of the implementation: classical control theory assumes that the controller is computed continuously and instantaneously, and to infinite precision. The actual implementation, on the other hand, must schedule tasks, and represent computations with fixed precision. So, while the mathematical analysis computes not just a controller but also a proof of correctness under idealized mathematical models, it is not clear if the proof of correctness remains valid for the implementation. Since software implementations of controllers for physical subsystems form the core of many modern safety-critical systems such as aircraft flight control and automotive engine control, we would like to design a methodology that can transfer the proofs performed at the mathematical level to properties of the implementation.

In our research, we have developed a methodology and a tool to perform automated static analysis of embedded controller code for stability of the controlled physical system. Our methodology is based on the following separation of concerns. First, we analyze the controller mathematical models to derive bounds on the implementation errors that can be tolerated while still guaranteeing properties of the system, such as stability. Second, we automatically analyze the controller software to check if the maximal implementation error is within the tolerance bound computed in the first step. In our initial work, we assume that the only source of imprecision is due to finite-precision arithmetic.

We have implemented this methodology in Costan, a tool to check stability for controller implementations. Using Costan, we analyzed a set of control examples whose mathematical models are given in Simulink and whose C implementation is generated using Real-Time Workshop. Our technique combines analysis of the mathematical controller models and automated analysis of source code to guarantee stability properties. Our first paper on the topic [21] received the ACM SIGBED Best Paper Award at EMSOFT 2010. We are currently working on extending our results to other sources of error, for example, control packet drops due to non-schedulability.

8.4 A theory of robustness for discrete synthesis

Current system design and verification techniques take a 0-1 view of the world: a property either holds or it doesn't, and a synthesized system provides no guarantees if the environment deviates in any way from the model. In practice, this view can be overly restrictive. First, CPSs need to operate for extended periods of time in environments that are either unknown or difficult to describe and predict at design time. For example, sensors and actuators may have noise, there can be mismatches between the dynamics of the physical world and its model, software scheduling strategies can change dynamically. Thus, asking for a model that encompasses all possible scenarios places an undue burden on the programmer, and the detailed bookkeeping of every deviation from nominal behaviour renders the specifications difficult to understand and maintain. Second, even when certain assumptions are violated at run-time, we would expect the system to behave in a *robust* way: either by continuing to guarantee correct behaviour or by ensuring that —even in the presence of small perturbations— the resulting behaviour only deviates modestly from the desired behaviour. Unfortunately, current design methodologies for CPSs fall short in this respect: the Boolean view cannot specify or guarantee acceptable behavior when small changes occur in the physical world, in the software world, or in their interaction.

In a recent paper [144], we provide a theory and algorithmic tools for the design of robust discrete controllers for ω-regular properties on discrete transition systems. Our starting point is the observation that a notion of robustness and associated design tools have been successfully developed in continuous control theory. There, the control designer designs the system for the nominal case, while bounding the effects of uncertainties and errors on the performance of the system. Our goal is to provide a similar theory and algorithmic tools in the presence of both discrete and continuous changes on the one hand, and in the presence of more complex temporal specifications —given, for example, in linear temporal logic (LTL) or as ω-automata— on the other hand. We do this in three steps.

First, robustness is a topological concept. In order to define it, we need to give meaning to the words "closeness" or "distance." For this, we define a metric on the system states. Second, instead of directly modeling the effect of every disturbance, we model a *nominal* system (the case with no disturbance), together with a set of (unmodeled) disturbances whose effect can be bounded in the metric. That is, while making no assumption on the nature or origin of disturbances, we assume that the disturbances can only push a state to another one within a distance γ. Third, under these three assumptions, we show how we can derive strategies for ω-regular objectives that are robust in that the deviation from nominal behaviour can be bounded as a function of the disturbance and the parameters of the system.

To illustrate this last point, consider *reachability properties* $\Diamond F$, where the system tries to reach a given set of states F. We give algorithms to compute strategies that ensure F is reached in the nominal case, and additionally, when disturbances are present, guarantee that the system reaches a set F' which contains states at most distance $\Delta(\gamma)$ from F, where Δ is an increasing function of the magnitude γ of the disturbance. More importantly, we show that an arbitrary strategy obtained through classical automata-theoretic constructions (e.g., [149, 209]) may only provide trivial robustness guarantees (e.g., a bounded disturbance can force the system to reach any arbitrary state). We show how similar arguments can be made to bound the system dynamics for Büchi and parity (and thus, for all LTL) specifications under the presence of disturbances.

Technically, our constructions lift arguments similar to arguments in robust control based on control Lyapunov functions to the setting of ω-regular properties. For reachability, the correspondence is simple: we require that the strategy decrements a "rank function" at a rate that depends on the distance to the target. For parity, the argument is more technical, and uses progress measures for parity games [129, 158]. Finally, we show algorithms to compute strategies with these properties based on shortest path algorithms on graphs.

In contrast to strategies computed by classical automata-theoretic algorithms, the strategies computed by our algorithm ensure that the behaviours of the controlled system under disturbances satisfy a related property which depends on the magnitude of the disturbance. We show an application of our theory to the design of controllers that tolerate infinitely many transient errors provided they occur infrequently enough.

8.5 Tool support for embedded control systems

Over the past year, we have developed automated tools for the design and analysis of embedded control systems. First, we have developed a tool based on symbolic execution to generate test cases for controller implementations in C. The tool builds on our infrastructure for Splat, a concolic testing tool for C programs, and adds support for floating point numbers and non-linear arithmetic. Internally, our tool generates interval-valued constraints, and uses the decision procedure HySat to discharge the symbolic constraints. Additionally, our tool implements additional functionality, based on symbolic constraint solving: for each variable, it computes upper and lower bounds on the values; for each output, it returns the sensitivity of the output to input perturbations. Our tool is now used at Toyota's testing lab in Gardena.

Second, we have developed Pessoa [178], a tool for the automatic synthesis of controllers. Pessoa takes as input (1) specifications in an easily determinizable subset of LTL and (2) systems described as sets of linear differential equations (which are internally abstracted to a finite state system which is ϵ-bisimilar to the original dynamical system). We have applied Pessoa to a set of robot motion planning examples.

8.6 Parameterized verification

Many software programs are written for *parameterized* settings. For example, a memory management system is designed to work for any number of processes requesting memory, and for any (sufficiently large) memory. A challenge in the verification of such systems is to be able to show that the system is correct, *no matter how the parameters are instantiated.* In general, parameterized verification is undecidable, even when each instantiation of the parameters yields a finite-state system. Thus, automatic tools for uniform verification must come up with heuristics that are sufficient to infer uniform inductive invariants in practical examples. In a recent paper [84], we describe a technique for parameterized verification that merges ideas from parameterized hardware and protocol verification—verification by invisible invariants and symmetry reduction—with ideas from software verification—template-based invariant generation and satisfiability checking for quantified formulæ (modulo theories). The combination enables us to precisely model and analyze unbounded systems while taming state explosion.

We applied our algorithm to the automated verification of transactional memory implementations, checking for the property that a transactional memory implementation ensures strict serializability, for any number of processes and any number of memory locations. We used the formalization of TM protocols from [104]. Our technique enables automated proofs that two-phase locking (TPL), dynamic software transactional memory (DSTM), and transactional locking II (TL2) systems ensure strict serializability. The verification is challenging since the systems are unbounded in several dimensions: the number and length of concurrently executing transactions, and the size of the shared memory they access, have no finite limit. In contrast, state-of-the-art software model checking tools such as Blast and Tvla are unable to validate either system, due to inherent expressiveness limitations or state explosion.

In addition, with my student Manu Jose, I have worked on problems in logic and physical synthesis for FPGAs to maximize design reliability (additional collaborators: Lei He (UCLA) and Yu Hu (University of Alberta)) ([127, 126]), and on debugging techniques based on symbolic execution and MAX-SAT ([128]).

8.7 The next year

Over the next year, we shall maintain our focus on these problems. In particular, we are currently working on the following topics.

A closed-loop symbolic test generation tool. We are currently extending the Splat infrastructure to be able to symbolically simulate closed-loop designs, where part of the system (e.g., the plant model) is specified as a hybrid automaton, and part of the system (e.g., the controller implementation) is given as C code. Technically, our symbolic simulation algorithm will combine symbolic model checking algorithms for hybrid automata (e.g., those based on polyhedral manipulations [122] or more recent versions that compute careful polyhedral approximations for linear dynamics [103]) with concolic execution techniques that we have already implemented in Splat. The challenge in building the system is allowing seamless integration of the techniques, and making it scale to real industrial examples. Even though individual pieces of the tool have been developed before, to the best of our knowledge, there is no currently available tool that combines these features (symbolic co-analysis of models and code).

We are in contact with engineers from Fraunhofer IESE at Kaiserslautern, from Toyota Motors, and from China Electric Grid, and hope to evaluate our tool on examples from these companies. (Initial experiments with Splat at Toyota has demonstrated promising results.)

[People involved in the project: Indranil Saha (UCLA), K.C. Shashidhar (MPI-SWS), Zilong Wang (MPI-SWS).]

Control-aware schedulability. We are extending our results on end-to-end verification of control systems. Currently, control computations are passed to the real-time OS as hard real-time tasks (i.e., those that must meet their deadlines), and then a schedulability analysis is performed to check if all hard real-time tasks can be scheduled. If they cannot, the conventional wisdom is to add resources. Our observation is that one can go back to the control system design and calculate the maximal fraction of control packets that can be "dropped" while still maintaining control system performance. This calculation gives a more relaxed schedulability problem, where the scheduler can drop tasks occasionally, as long as the tasks are run "enough" times. We have built a prototype based on this idea, and we are exploring both static and dynamic scheduling schemes.

[People involved in the project: Indranil Saha and Majid Zamani (both UCLA).]

A tool for parametric verification of protocols. We are extending our work on automated verification of parameterized protocols by considering new invariant generation algorithms. Our first attempt was based on an algorithm to infer quantified interpolants, but our experimental results were disappointing: interpolant based techniques seem quite brittle, and fail to converge for very simple protocols. Our second attempt is based on heuristics that are themselves based on finite model finding and syntactic generalization. We are currently attempting fully automatic proofs of cache coherence protocols such as German's protocol to demonstrate the effectiveness of our algorithms.

[People involved in the project: Sai Deep Tetali (UCLA), Amit Goel (Intel), Sava Krstic (Intel).]

9 The Dependable Systems Group

9.1 Overview

This report covers the period from May 2009–April 2011. The group's mission is to develop reliable, highly-available software systems, by improving the methods used to build such systems.

Personnel:

The group is led by Rodrigo Rodrigues, who joined the institute in January 2008. It currently includes one postdoc, Allen Clement who completed his PhD at UT Austin in 2010, six PhD students: Pramod Bhatotia (Master's from IIT Kanpur, joined in 2009), Daniel Porto (Master's from the Federal University of Paraiba, joined in 2010), Pedro Fonseca (5-year undergraduate degree from IST in Lisbon, joined in 2008), Cheng Li (undergraduate degree from Nankai University, joined in 2009), Nuno Santos (Master's from IST in Lisbon, joined in 2008), and Alexander Wieder (Master's from Saarland University, joined in 2010), one Master's student, João Carreira (3-year undergraduate degree from IST in Lisbon), and one undergraduate student, Stefan Tombers.

Collaborations: Group members work in close collaboration with the majority of the groups in the institute. In particular, active collaborations exist with the distributed systems and operating systems group led by Peter Druschel, with the networked systems group led by Krishna Gummadi, with the programming languages and systems group led by Umut Acar, and with the rigorous software engineering group led by Rupak Majumdar.

Additionally, various members of the group collaborate with faculty and researchers from various leading worldwide research institutions, namely, with professors from Cornell University, EPFL, Technion, and the New University of Lisbon, and researchers from Yahoo!, Microsoft, NEC, Deutsche Telekom, and Google.

Rodrigo Rodrigues is also leading the participation of the institute in the VICCI cloud computing research testbed. This testbed consists of seven geographically dispersed compute clusters, located in several worldwide universities, namely Princeton, Stanford, Georgia Tech, University of Washington, ETH Zurich, and University of Tokyo, in addition to MPI-SWS. As part of this testbed we are concluding the installation of a local cluster with a total of 840 CPU cores, 3.4TB RAM, 210TB storage, and four programmable OpenFlow switches.

Publications: The group has published in most of the elite conferences among the broad systems research community, namely OSDI 2010 [112], and twice in EuroSys 2011 [87, 97]. We have also had a steady presence in what is considered the flagship conference of the more specialized dependability community, DSN [94, 88]. We published an article that was featured in the cover of the October 2010 edition of the Communication of the ACM [88]. In terms of publications in journals, we have also published in the Transactions on Parallel and Distributed Systems [89] and Transactions on Dependable and Secure Computing [172]. Finally, we had a number of workshop and short conference papers, also present in highly competitive venues, namely HotCloud [183], LADIS [55, 205], and PODC [204]. In the context of their industry experience, group members have also published in HotMobile [184] and HiPC [186]. This last paper won a best paper award.

Support: The research of the Dependable Systems Group has been supported by the base funding from the Max Planck Society, by a Faculty Research and Engagement award from Yahoo!, and by a research grant from Amazon in the form of AWS credits.

Software / technology transfer: We made available several software prototypes in the context of various research projects described below.

We also participated in the following provisional patent application filed in October 2010: “Protecting Data Integrity with Storage Leases”. Inventors: Peter Druschel, Rodrigo Rodrigues, and Ansley Post.

Teaching: Rodrigo Rodrigues co-taught a core graduate course on Operating Systems at Saarland University during the Summer semester of 2009, a core graduate course on Distributed Systems during the Summer semester of 2010, and a graduate seminar on Recent Advances in Computer Systems during the Winter semester of 2010/11. He lectured at the second winter school on Hot Topics in Distributed Computing in March 2009. Rodrigo will be teaching the Operating Systems course again in 2011. Nuno Santos was a TA for the Operating Systems course in 2009, Alexander Wieder was a TA for the Distributed Systems course in 2010, and Pedro Fonseca and Nuno Santos were TAs for the Recent Advances in Computer Systems course in 2010/11.

Service: Rodrigo Rodrigues served as Program Committee co-chair for three workshops: the 8th International Workshop on Peer-to-Peer Systems (IPTPS 2009), the 5th Workshop on Hot Topics in System Dependability (HotDep 2009), and the Workshop on Theory and Practice of Byzantine Fault Tolerance (BFTW3). Additionally, he served on the program committees of the following conferences and workshops: 1st International Workshop on Dependability of Clouds, Data Centers and Virtual Computing Environments (DCDV 2011), 4th Annual International Systems and Storage Conference (SYSTOR 2011), 5th EuroSys Doctoral Workshop (EuroDW 2011), 29th IEEE Symposium on Reliable Distributed Systems (SRDS 2010), 4th Workshop on Recent Advances in Intrusion-Tolerant Systems (WRAITS 2010), 40th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2010), 7th USENIX Symposium on Networked Systems Design and Implementation (NSDI '10), 9th International Workshop on Peer-to-Peer Systems (IPTPS '10), and 3rd ACM SIGOPS International Workshop on Large Scale Distributed Systems and Middleware (LADIS 2009).

Rodrigo Rodrigues also served on three doctoral thesis committees during this period, one in Rice University, one at the TU Darmstadt, and one in the MPI-SWS graduate program. He also gave a keynote talk at the Workshop on Cryptography and Security in Clouds in Zurich.

In terms of conference organization, Rodrigo Rodrigues served as workshop co-chair and publicity co-chair of the 23rd ACM Symposium on Operating Systems Principles (SOSP'11).

Within MPI-SWS Rodrigo Rodrigues is the chair of the graduate studies committee, responsible for admissions, recruiting, and running the daily operation of the MPI-SWS graduate program.

9.2 Research agenda

The research of the dependable systems group focuses on improving the reliability of software systems. This goal is increasingly important, as software systems become more and more prevalent, and is also increasingly challenging, as these very same systems grow in size and complexity.

Following this objective, the group's research has been inspired by two important trends that have shaped the computing landscape in the last few years, and which have provided an interesting source of reliability challenges and opportunities.

The first trend is that computer systems make increasing use of computing resources that are offered as services over the Internet, and hosted in large data centers operated by a diverse set of providers. This trend, for which the term "cloud computing" was coined, is motivated by a serendipitous alignment of interests between the providers of "cloud" services, who leverage economies of scale and their own expertise in managing a large-scale infrastructure, and the users of these services, who obtain elastic access to vast resources. Cloud customers are thus able to obtain high scalability without incurring a large upfront investment, and are also freed from having to manage part of their IT infrastructure.

The second trend is that processor speeds are no longer increasing, and instead current architectures rely on hardware parallelism to improve performance with the increase of the number of processors per chip. This in turn has implications for the design of software, which has to become more concurrent to take advantage of this increased processing capacity.

These trends raise a series of new challenges, which form the basis for most of the technical problems that our research addresses. But, simultaneously, they also represent new opportunities, in particular since extracting value from this increased scale and processing capacity can take various different forms, not only in the most obvious way of making systems faster and capable of implementing more functionality, but also by making them more reliable. The latter vector is precisely the one we intend to explore.

9.3 Data center-scale and cloud computing

One of the lessons learned from the deployment of systems of the scale and complexity of the ones that form the back-end of cloud infrastructures is that events that used to be considered too rare to be worth worrying about become the norm, and no longer the exception. To give some examples, there have been common reports of issues like message corruption going undetected by standard protocol checksums, or (similarly) storage systems silently returning incorrect values.

Given this situation, the area of Byzantine Fault Tolerance (BFT), which studies techniques to handle faults that cause system components to fail by arbitrarily deviating from their specified behavior, offers a solid foundation to tackle such problems. During the past decade, the study of techniques to handle Byzantine faults received significant attention in the systems community, with most of the research contributions centered on making BFT protocols perform better. But despite these performance advances, the adoption of BFT techniques in practical settings has been limited, and therefore our take on this problem attempts to take a step back and ameliorate other possible obstacles to the adoption of BFT.

Highly available BFT replication with Zeno. Data center-scale services replicate their state across multiple sites to be able to withstand a catastrophic outage of an entire data center. While BFT replication could be applied in this setting, traditional BFT protocols would become unavailable if a small fraction of their replicas are unreachable. This is because these protocols favor strong safety guarantees (consistency) over liveness (availability). This is in contrast to the approach taken by the replicated systems that are used by providers of cloud and other data center-scale services like Google or Amazon, which strive to achieve high availability for the replicated service, if necessary at the expense of weakening the semantics provided by that service. To fill this gap, and increase the chances of adoption of BFT protocols in this setting, we proposed Zeno, the first protocol to provide eventual consistency while tolerating Byzantine faults. Zeno was published at NSDI'09 [189].

Byzantium: BFT databases with snapshot isolation. Another computer system that forms a crucial part of the computer infrastructure of any organization, and where weaker forms of consistency are also commonly employed are database systems. In the case of databases, they often resort to so-called weaker isolation levels to improve performance in comparison to more strict semantics. In another piece of work, we proposed a system called Byzantium, which is the first to exploit the use of *snapshot isolation* in the context of BFT replication of database systems. This scheme improves substantially on previous proposals for BFT replication of databases because it allows transactions to be executed concurrently (which is crucial for performance) and relies on no centralized components of whose correctness and integrity the system depends (which is crucial for fault tolerance). Byzantium was published at EuroSys'11 [97].

BFT in large-scale, dynamic systems. On a separate avenue of research, but also in the context of data center-scale services, we have studied how BFT protocols should be adapted to handle two distinctive characteristics of these services: their large scale, and the fact that they need to cope with a dynamic membership in the set of machines that collectively provide the service. This led to the creation of new BFT building blocks, namely a service to keep track of a large, dynamic system membership, a BFT replication protocol that builds on top of this service and allows for changes in the replica set, and a BFT key/value storage system that uses this replication protocol. This was published in the Transactions on Dependable and Secure Computing journal [172].

A fault model for the data center. Despite these advances, practitioners are still skeptical about the adoption of BFT techniques, claiming that they are designed to tolerate a worst-case scenario of an attacker controlling the faulty machine and causing it to behave in the worst possible way. This capability is perceived as unnecessary because the protected environment of the data center has high security fences that guard against these kinds of threats. To address these issues, we started collaborating with researchers at Yahoo! to develop alternative techniques that are better suited for a data center environment. Intuitively, we are developing a fault tolerance mechanism for an important class of data center computations that is akin to the use of checksums or error-correcting codes when transmitting messages through a noisy network channel. In particular, it is possible to add a small, parameterizable amount of redundancy that enables us to trade cost for coverage. A preliminary design was presented in a LADIS'10 workshop paper [55].

Detecting misbehavior of remote computations. Another distinctive feature of cloud computing is that computations are outsourced to remote cloud providers instead of being run locally. This raises concerns about the integrity and confidentiality of the data and computations that are offloaded to the cloud. The Byzantine fault model provides a good foundation for developing solutions to the problem of ensuring the integrity of outsourced computations, since a cloud provider that does not implement a correct service can be seen as a Byzantine participant of a distributed system. However, one of the obstacles we faced when applying Byzantine fault detection protocols in this context was that such protocols require deep understanding and modifications to the application being deployed for making it fully deterministic and that all communication be funneled through the protocol library. To address these limitations, we proposed a technology called accountable virtual machines (AVMs). AVMs combine techniques for Byzantine fault detection from the PeerReview protocol with techniques for deterministic replay of virtual machines, to allow them to (i) record a log with enough information to reproduce the entire execution of the system, and (ii) associate each outgoing message with a cryptographic record that links that action to the log of the execution that produced it. This enables auditing a remote execution by replaying it using a known-good copy of the system, and cross-checking the traffic produced by the remote system with the behavior of that copy. We used AVMs to detect cheating in online games [112] and to detect deviations from the expected behavior of web applications that are outsourced to a cloud provider [203]. This work was published at OSDI'10 [112].

Using TPMs to tame uncertainty in the cloud. Another important technology that is well-suited for addressing the problem of ensuring the integrity and confidentiality of computations that are outsourced to a cloud provider is trusted computing, and in particular trusted platform modules (TPMs) that ship with commodity hardware. However, applying this technology in the context of cloud services is not straightforward, since the primitives that are made available by trusted computing platforms are not designed for a scenario where data is manipulated by a large and dynamic set of hosts. These primitives are also very inefficient, which makes it prohibitive to expose such primitives to external customers of cloud services. To address this, we proposed a new set of trusted computing concepts that build upon the existing ones, but are geared towards the specific characteristics of the cloud environment. We also defined an architecture and a set of protocols that enabled us to implement these new trusted computing concepts, hence forming a new root of trust for building trusted cloud services. A preliminary design appeared in the HotCloud'09 workshop [183], and a full paper is under submission to SOSP'11 [185].

Orchestrating the choice of cloud resources. Orthogonal to trust issues, cloud customers are also faced with the task of choosing which cloud services to subscribe to. As the number of cloud services increases, and the services and pricing models themselves become more complex, this task also becomes increasingly complex. In another research project, we proposed Conductor, a system that enables cloud customers to determine the optimal set of cloud services to make use of, given a budget that specifies a maximum monetary cost, time to completion, or a combination. To achieve this, Conductor (i) models the computations and the set of cloud resources that are available as a linear program, (ii) uses optimization techniques to choose an initial deployment plan, and (iii) monitors the deployment to check for deviations from the expected progress (due to faults or mispredictions) and adjusts the computation accordingly. Preliminary results were presented in short PODC'10 and LADIS'10 papers [204, 205].

Incremental MapReduce. Finally, another area that emerged as a consequence of the access to vast computing power located in data centers with tens to hundreds of thousands of machines was the development of languages and associated runtime environments for data-parallel processing of vast amounts of data. In particular, the MapReduce programming model emerged as a de facto standard for large-scale data analysis. However, as online data sets grow over time, MapReduce computations become increasingly more expensive. Furthermore, often the same computation runs on evolving data sets repeatedly. This motivates an incremental approach where results are incrementally updated as data evolves, instead of being recomputed from scratch. In a research project called Incoop, we proposed a generic MapReduce framework for incremental computations. Incoop detects changes to the inputs to computations and enables the automatic update of the outputs by employing an efficient, fine-grained propagation mechanism. This work is under submission to the SOCC'11 conference [54], and a short position paper was accepted for publication at the HotCloud'11 workshop [53].

9.4 Concurrency

One of the proposed uses of Byzantine fault tolerance is to tolerate faults stemming from software bugs, which can, for instance, cause the application to output wrong values. However, the elephant in the room as far as this use of BFT is concerned is that when these software bugs are deterministic, all the replicas will activate the same bug and fail in a correlated manner, defeating the purpose of BFT replication.

This observation, coupled with the aforementioned pressure for applications to become more parallel, which makes them more prone to subtle concurrency bugs, led us to study the potential of using multiple thread schedules in concurrent applications to create the necessary diversity to withstand Byzantine faults caused by software bugs.

Understanding the effects of concurrency bugs. We pursued a principled approach by first conducting a study of bug logs of a large-scale concurrent server application. This study led to some interesting findings, namely that there is a significant fraction of bugs that cause Byzantine behavior, and also that these were correlated with latent concurrency bugs, which silently corrupt data structures and are exposed to the user potentially much later than the interleaving that triggers the bug. This work was published in the DSN'10 conference [88].

Finding non-crash concurrency bugs. The next step towards finding a way to tolerate these Byzantine bugs is to find a mechanism to detect them, so that faulty thread schedules can be overruled by correct ones. This led us to study techniques for software testing and dynamic bug detection. Traditional techniques assume bugs manifest themselves by causing the program to crash (e.g., due to an illegal memory access) or that the programmer wrote a set of assertions that flag the existence of a bug during the test. This does not address our need to find bugs that silently corrupt internal state or manifest themselves by returning incorrect answers. To find such bugs, we proposed a way to implicitly extract a specification, both for the correct state and the correct output of the application, by testing if the application obeys linearizable semantics. This new scheme, coupled with a method for systematically creating different thread interleavings, led to finding a considerable number of bugs in MySQL. This work was published in EuroSys'11 [87].

Recovering from latent bugs in multi-threaded applications. The future direction for this project is to pursue a replication approach where different replicas are executed at run-time and with different thread schedules. In this part of the project, the intention is to first focus on latent bugs, and exploit the fact that these indicate a temporal gap between the state becoming corrupt and the corresponding failure becoming visible to the clients or users of the application. Thus the goal is to explore an opportunistic approach in which the spare capacity of multi-core processors is used to run the redundant thread schedules, to try to recover from latent bugs before they are exposed.

9.5 Peer-to-peer systems

Lessons learned from peer-to-peer systems. Our work in this field consisted of reporting our experience with the implementation of a full information overlay network that routes in a single hop in most cases, which appeared in the Transactions on Parallel and Distributed Systems journal [89]. We also published a conference paper, which appeared in DSN'09, on how to handle the threat of worms that spread rapidly by following links in a peer-to-peer overlay [94]. Finally, to close the circle on our exploration of this line of research, we also published an article that was featured on the cover of the Communications of the ACM that gives an overview of a decade of research and deployment of peer-to-peer systems [171].

10 The Software Analysis and Verification Group

10.1 Overview

This report covers the period of October 2010 – March 2011. This group focuses on the formal specification and verification of software systems, in developing theories and tools for rigorously applying formal reasoning to build correct software systems.

Personnel: The group currently consists solely of Viktor Vafeiadis, who joined MPI-SWS in October 2010 from the University of Cambridge.

Collaborations: We collaborate closely with Derek Dreyer's group and to a lesser extent with Umut Acar's group. We also collaborate with researchers at the University of Cambridge (Jaroslav Sevcik, Peter Sewell), Microsoft Research Cambridge (Josh Berdine, Byron Cook, Matthew Parkinson), IMDEA (Alexey Gotsman, Aleksandar Nanevski), INRIA (Francesco Zappa Nardelli), and Purdue (Suresh Jagannathan).

Publications: In the past six months, we have had two publications in top conferences (POPL 2011 [187], LICS 2011 [124]), and we have two papers currently under submission to MFPS 2011 and SAS 2011.

Service: Viktor has served on the program committee of CAV 2011 and has been an external reviewer for LICS 2011, FM 2011, MFPS 2011, and JOT. Together with Rupak Majumdar and Derek Dreyer, he served on the MPI-SWS faculty recruiting committee in 2011.

Teaching: This spring, Derek and Viktor will co-teach a graduate seminar on *Concurrent Program Logics*. The course will cover the long line of program logics for reasoning about concurrent imperative programs, beginning with the early work on Hoare logic and rely-guarantee reasoning and leading to recent work on RGSep, deny-guarantee, and concurrent abstract predicates.

10.2 Research agenda

The group's main research interest lies in developing theories and tools for enabling formal reasoning about software correctness and performance. The goal of this research is to improve software quality by making it possible to build provably correct software components. This involves coming up with rigorous mathematical specifications of software components, such as data structure libraries and compilers, and developing custom proof techniques for proving adherence to those specifications. The group's research so far can be divided into two areas:

1. Designing new program logics and proving soundness of existing ones. Program logics, such as Hoare logic and separation logic, are techniques for structuring proofs about program correctness. Our contributions so far are (i) a new soundness proof for concurrent separation logic, and (i) an extension to separation logic to reason about low-level programs in the presence of garbage collection.
2. Producing verified optimising compilers, especially for programming languages with concurrent semantics. Our contributions so far are (i) an extension of Leroy's Compcert with TSO concurrency, and (ii) verification of compiler optimisations for removing unnecessary memory fence instructions.

Now we describe each of these contributions in more detail.

Program logics

Program logics are formalisms for reasoning about programs. The benefit of using a program logic rather than doing a proof directly over the semantics of the program to be verified is that program logics assist in developing and structuring the proofs. Good program logics permit modular reasoning following the program structure and provide convenient notation to deal with mundane reasoning aspects, such as pointer aliasing.

SL in the presence of GC Separation logic [170] is a program logic that is particularly well-suited for reasoning about pointer-manipulating programs. Its assertions describe heaps (mappings from memory addresses to values) and include a multiplicative separating conjunction connective $(*)$ with the following semantics: a heap, h. satisfies $P * Q$ iff it can be split into two domain-disjoint heaps, one satisfying P and the other satisfying Q. A Hoare triple, $\{P\}\ C\ \{Q\}$, in separation logic requires that the command, C, can run safely in an initial state described by precondition, P. A direct consequence of this interpretation of triples is the following very useful frame rule:

$$\frac{\{P\}\ C\ \{Q\}}{\{P * F\}\ C\ \{Q * F\}}$$

which says that C preserves any assertion, F, about state that is disjoint from its precondition.

Separation logic has been studied extensively in C-like languages, where there is no garbage collection (GC), and in ML-like languages, where garbage collection is taken for granted and is ignored from a verification perspective, but has never been studied in a low-level C-like language interfacing to a garbage collector. Such a scenario is not unrealistic: it occurs whenever C code has to interoperate with O'Caml or Java code.

There are two main challenges in reasoning about such code.

First, C code, being unsafe, can invalidate the internal invariants of the garbage collector and, moreover, may temporarily do so to gain performance so long as it ensures that no GC occurs in the meantime. More concretely, the O'Caml GC requires every value stored in memory to be tagged so as to indicate whether it represents a numerical value or a pointer. As it is more efficient to operate on untagged values, a C program should be allowed to do so, so long as it can prove that when GC occurs, all values are properly tagged. We address this challenge by providing a two level logic: At the inner level, we can reason about the program fragments that do not necessarily preserve the GC invariants and where no GC occurs. At the outer level, we reason about programs that involve GC, but all our proof rules require that programs preserve the GC invariants. We also have proof rule for moving from the inner level to the outer level; the rule requires that the all modified variables store GC-safe values.

Second, we would like to support the frame rule. In the presence of GC, its soundness, however, is far from obvious. If GC occurs during the execution of C, then it can modify the state represented by F. In particular, it can deallocate unreachable memory cells and (in the case of a copying collector), relocate some of the reachable memory cells. To support the frame rule, our assertions do not describe the physical heap, but rather a GC-insensitive logical heap. Asserting $e \mapsto e'$ does not mean that e is physically allocated; it merely says that e is allocated in the logical heap. To know that e is physically allocated (and therefore to be able to access it safely), e must be a program variable or a value stored in a heap cell that is reachable from a program variable.

(This is joint work with Chung-Kil Hur and Derek Dreyer and will appear in LICS 2011 [124].)

CSL soundness Concurrent separation logic [163] (CSL) is an extension of separation logic for reasoning about concurrent programs. Its main novelty over previous concurrent program logics is the notion of ownership transfer between a thread and a resource bundle.

Due to this notion, however, its soundness is a rather delicate subject either requiring that the logic has 'precise' resource invariants or that it does not have the conjunction rule. After its inception, it took a couple of years for the first variant of the logic to be proved sound and several more for the second variant. The first soundness proof by Brookes [58] introduced an instrumented semantics for defining the meaning of triples, a step that was blindly copied in most later soundness proofs of CSL and its extensions. The problem with these soundness proofs is not only that the meaning of CSL judgments is defined with respect to an unrealistic semantics, but also that the instrumented semantics actually hinders the actual proof.

We have recently come up with a considerably simpler soundness proof that dispenses the intermediate semantics, and describes the meaning of the CSL judgments directly in terms of a standard program semantics.

(This work is currently under submission to MFPS 2011 [196].)

Verified compilation

Compilers are ideal candidates for full functional verification. They are sizable programs, perform quite complicated program transformations, whose soundness often depends on separate program analyses, and yet their correctness can be formally specified.

Given the semantics of programs in the source and target languages, a successful run of the compiler has to produce a program in the target language that behaves equivalently to the source program. In case the source programming language is non-deterministic (e.g., an imperative language with concurrency), then the compiler is allowed to reduce the nondeterminism of the program by deciding, for example, not to spawn a new thread when the cost of doing so is too high. In this case, the behaviours of the target program produced by the compiler should merely be a subset of those of the source program.

CompCertTSO Our first contribution in the area of verified compilation was to extend Leroy's Compcert compiler [137] to a concurrent setting, by giving an operational semantics to a multi-threaded version of Clight and x86 assembly programs following the TSO relaxed memory model, and adapting the Compcert soundness proof to the concurrent setting. As part of the adaptation, we had to restrict one optimisation (common subexpression elimination), because in its unrestricted form it is unsound for concurrent programs.

(This is joint work with Jaroslav Sevcik, Francesco Zappa Nardelli, Suresh Jagannathan, and Peter Sewell, that appeared at POPL 2011 [187]. We are currently working towards a journal version of this paper.)

Fence optimisations In a follow-up project, we implemented two new TSO-specific optimisations that remove redundant memory barrier instructions, and proved their soundness in Coq. While the optimisations we implemented are very cheap to perform by a standard thread-local control flow analysis, their correctness is much more subtle and required a different proof strategy than the optimisations we had encountered in the CompcertTSO project. To carry out the proof, we came up with a non-standard global simulation argument, which effectively incorporates a boolean prophecy variable in an otherwise forward simulation.

(This is joint work with Francesco Zappa Nardelli, and is currently under submission to SAS 2011 [197].)

11 The Verification Systems Group

11.1 Overview

This report covers the period of April 2009–December 2009. The group focuses on the development of rigorous analysis and verification methods for improving reliability of software systems.

Personnel The group is led by Andrey Rybalchenko (joined in September 2007 from EPFL), and currently has one master's student, Nuno Lopes (joined in February 2009 from Instituto Superior Técnico Lisbon); two graduate students, Ashutosh K. Gupta (joined in September 2007 from EPFL) and Ruslan Ledesma Garza (joined in April 2009 from Universidad de las Américas Puebla); and two postdoctoral researchers, Juan Navarro Pérez (joined in January 2008 from University of Manchester) and Corneliu Popeea (joined in September 2008 from University of Singapore).

Rati Gelashvili from Tbilisi University completed an internship project "Computation of Hilbert bases" in the summer of 2009.

Collaborations The group has established collaborations leading to joint projects and publications. We collaborate with the distributed systems group (led by Peter Druschel) at the MPI-SWS, the information security and cryptography group (led by Michael Backes) at Saarland University, and with groups at EPFL, University of Freiburg, UC Los Angeles, MIT, University of Tel-Aviv, University of Tsukuba, and Microsoft Research labs in Cambridge and Redmond.

Publications The group has published regularly in the top-tier venues in its field and in related areas, as a result of collaborations. The research results were published in four conferences, VMCAI [143] CSF [132], ICLP [142], and FMCAD [70], as well as two workshops, SPIN [180] and LIS [134].

Tools In early 2009 the group started the development of DAHL, a programming system for distributed applications [142]. DAHL pursues three design objectives: make distributed programs easy to write, simplify their analysis and verification, and ensure their efficient execution. Our evaluation of DAHL implementations of two complex distributed protocols and a distributed software model checker [143] indicates that DAHL scales to its goals.

Service Juan Navarro Pérez served as a program committee member of MICAI 2009 and LA-NMR 2009. Furthermore, Juan was an external reviewer for TASE, LPNMR, and the journal of Theory and Practice of Logic Programming (TPLP).

Corneliu Popeea was an external reviewer for POPL 2010.

Andrey Rybalchenko served as a program committee member of POPL 2010 and PLPV 2010.

Teaching Andrey gave a tutorial at the 11th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNACS) in September 2009.

11.2 Research agenda and results

The group's main research interests are in the development of theories, algorithms, and tools for improving the reliability of software systems. We focus on rigorous techniques that provide mathematically sound guarantees. The group is committed to achieving a balance between research on theoretical foundations, algorithmic support, and applications. Our main research results are outlined below.

11.2.1 Software analysis and verification

Finding heap-bounds for hardware synthesis Dynamically allocated and manipulated data structures cannot be translated into hardware unless there is an upper bound on the amount of memory the program uses during all executions. This bound can depend on the generic parameters to the program, i.e., program inputs that are instantiated at synthesis time.

In collaboration with Byron Cook, Stephen Magill, Jiri Simsa, Satnam Singh, and Viktor Vafeiadis [70], we propose a constraint based method for the discovery of memory usage bounds, which leads to the first- known C-to-gates hardware synthesis supporting programs with non-trivial use of dynamically allocated memory, e.g., linked lists maintained with malloc and free. We illustrate the practicality of our tool on a range of examples.

Subsumer-First: steering symbolic reachability analysis Symbolic reachability analysis provides a basis for the verification of software systems by offering algorithmic support for the exploration of the program state space when searching for proofs or counterexamples. The choice of exploration strategy employed by the analysis has direct impact on its success, whereas the ability to find short counterexamples quickly and—as a complementary task—to efficiently perform the exhaustive state space traversal are of utmost importance for the majority of verification efforts. Existing exploration strategies can optimize only one of these objectives which leads to a sub-optimal reachability analysis, e.g., breadth-first search may sacrifice the exploration efficiency and chaotic iteration can miss minimal counterexamples.

In collaboration with Rishabh Singh [180], we present *subsumer-first*, a new approach for steering symbolic reachability analysis that targets both minimal counterexample discovery and efficiency of exhaustive exploration. Our approach leverages the result of fixpoint checks performed during symbolic reachability analysis to bias the exploration strategy towards its objectives, and does not require any additional computation. We demonstrate how the subsumer-first approach can be applied to improve efficiency of software verification tools based on predicate abstraction. Our experimental evaluation indicates the practical usefulness of the approach: we observe significant efficiency improvements (median value 40%) on difficult verification benchmarks from the transportation domain.

11.2.2 Distributed programming and applications

Applying Prolog to develop distributed systems Development of distributed systems is a difficult task. Declarative programming techniques hold a promising potential for effectively supporting programmer in this challenge. While Datalog-based languages have been actively explored for programming distributed systems, Prolog received relatively little attention in this application area so far.

In collaboration with Atul Singh [142], we present a Prolog-based programming system, called DAHL, for the declarative development of distributed systems. DAHL extends Prolog with an event-driven control mechanism and built-in networking procedures. Our experimental evaluation using a distributed hash-table data structure, a protocol for achieving Byzantine fault tolerance, and a distributed software model checker – all implemented in DAHL – indicates the viability of the approach.

Distributed and predictable software model checking In [143], we present a predicate abstraction and refinement-based algorithm for software verification that is designed for the distributed execution on compute nodes that communicate via message passing, as found in today's compute clusters. A successful adaptation of predicate abstraction and refinement from sequential to distributed setting needs to address challenges imposed by the inherent non-determinism present in distributed computing environments. In fact, our experiments show that up to an order of magnitude variation of the running time is common when a naive distribution scheme is applied, often resulting in significantly worse running time than the non-distributed version.

We present an algorithm that overcomes this pitfall by making deterministic the counterexample selection in spite of the distribution, and still efficiently exploits distributed computational resources. We demonstrate that our distributed software verification algorithm is practical by an experimental evaluation on a set of difficult benchmark problems from the transportation domain.

11.2.3 Software security

Approximation and randomization for quantitative information-flow analysis. Quantitative information-flow analysis (QIF) is an emerging technique for establishing information-theoretic confidentiality properties. Automation of QIF is an important step towards ensuring its practical applicability, since manual reasoning about program security has been shown to be a tedious and expensive task. Existing automated techniques for QIF fall short of providing full coverage of all program executions, especially in the presence of unbounded loops and data structures, which are notoriously difficult to analyze automatically.

In collaboration with Boris Köpf [132], we propose a blend of approximation and randomization techniques to bear on the challenge of sufficiently precise, yet efficient computation of quantitative information flow properties. Our approach relies on a sampling method to enumerate large or unbounded secret spaces, and applies both static and dynamic program analysis techniques to deliver necessary over- and under-approximations of information-theoretic characteristics.

A multi-modal framework for achieving accountability in multi-agent systems In collaboration with Simon Kramer [134], we present a multi-modal, model-theoretic framework for achieving accountability in multi-agent systems through formal proof. Our framework provides modalities for knowledge, provability, and time. With these modalities, we formalise the two main aspects of accountability, which are: soundness (accountability proper), i.e., for correct agents, the provability of their correctness by themselves; and completeness (auditability), i.e., for faulty agents, the eventual provability of their faultiness by others. In our framework, the accountability proof of a particular system is reduced to the proof of a few key lemmata, which the system designer needs to establish for a considered system.

References

[1] S. Abramsky, K. Honda, and G. McCusker. A fully abstract game semantics for general references. In *IEEE Symposium on Logic in Computer Science (LICS)*, 1998.

[2] U. Acar, P. Buneman, J. Cheney, J. V. den Bussche, N. Kwasnikowska, and S. Vansummeren. A graph model of data and workflow provenance. In *USENIX/ACM Workshop on the Theory and Practice of Provenance*, 2010.

[3] U. A. Acar, A. Ahmed, J. Cheney, and R. Perera. A calculus for provenance: Computations that explain their work. Submitted for publication, 2011.

[4] U. A. Acar, G. E. Blelloch, and R. D. Blumofe. The data locality of work stealing. *Theory of Computing Systems (TOCS)*, 35(3):321—347, 2002.

[5] U. A. Acar, G. E. Blelloch, R. Ley-Wild, K. Tangwongsan, and D. Türkoğlu. Traceable data types for self-adjusting computation. In *Programming Language Design and Implementation*, 2010.

[6] U. A. Acar, G. E. Blelloch, K. Tangwongsan, and D. Türkoğlu. Robust kinetic convex hulls in 3D. In *Proceedings of the 16th Annual European Symposium on Algorithms*, September 2008.

[7] U. A. Acar, A. Charguéraud, and M. Rainey. Oracle scheduling: Controlling granularity in implicity parallel languages. Submitted, 2011.

[8] U. A. Acar, A. Cotter, B. Hudson, and D. Türkoğlu. Dynamic well-spaced point sets. In *SCG '10: Proceedings of the 26th Annual Symposium on Computational Geometry*, 2010.

[9] U. A. Acar, A. Cotter, B. Hudson, and D. Türkoğlu. Dynamic well-spaced point sets. Submitted, 2011.

[10] U. A. Acar, A. Cotter, B. Hudson, and D. Türkoğlu. Parallelism in dynamic well-spaced point sets. In *Symposium on Parallelism in Algorithms and Architectures*, 2011.

[11] U. A. Acar, B. Hudson, and D. Türkoğlu. Kinetic well-spaced point sets in 2D. In *Fall Workshop on Computational Geometry*, 2010.

[12] U. A. Acar, B. Hudson, and D. Türkoğlu. Kinetic mesh-refinement in 2D. In *SCG '11: Proceedings of the 27th Annual Symposium on Computational Geometry*, 2011.

[13] U. A. Acar, A. Ihler, R. Mettu, and O. Sümer. Adaptive Bayesian inference. In *Neural Information Processing Systems (NIPS)*, 2007.

[14] U. A. Acar, A. Ihler, R. Mettu, and O. Sümer. Adaptive inference on general graphical models. In *Uncertainty in Artificial Intelligence (UAI)*, 2008.

[15] U. A. Acar, A. Ihler, R. Mettu, and O. Sümer. Maintaining MAP configurations with applications to protein sidechain packing. In *IEEE/SP 15th Workshop on Statistical Signal Processing (SSP)*, 2009.

[16] U. A. Acar, A. Ihler, R. Mettu, and O. Sümer. Adaptive Bayesian inference. Submitted, 2011.

[17] A. Ahmed. *Semantics of types for mutable state*. PhD thesis, Princeton University, 2004.

[18] A. Ahmed. Step-indexed syntactic logical relations for recursive and quantified types. In *European Symposium on Programming (ESOP)*, 2006.

[19] A. Ahmed, D. Dreyer, and A. Rossberg. State-dependent representation independence. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 2009.

[20] J. An, M. Cha, K. P. Gummadi, and J. Crowcroft. Media Landscape in Twitter: A World of New Conventions and Political Diversity. In *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media (ICWSM)*, Barcelona, Spain, July 2011.

[21] A. Anta, R. Majumdar, I. Saha, and P. Tabuada. Automatic verification of control system implementations. In *EMSOFT 10: Embedded Software*. ACM, 2010.

[22] A. W. Appel and D. McAllester. An indexed model of recursive types for foundational proof-carrying code. *ACM Transactions on Programming Languages and Systems*, 23(5):657–683, 2001.

[23] A. W. Appel, P.-A. Melliès, C. D. Richards, and J. Vouillon. A very modal model of a modern, major, general type system. In *ACM*

SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL), 2007.

[24] B. Aydemir, A. Charguéraud, B. C. Pierce, R. Pollack, and S. Weirich. Engineering formal metatheory. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 2008.

[25] M. Backes, M. Berg, and B. Köpf. Non-uniform distributions in quantitative information-flow. In *ACM AsiaCCS 2011*, 2011.

[26] M. Backes, I. Cervesato, A. D. Jaggard, A. Scedrov, and J.-K. Tsay. Cryptographically sound security proofs for basic and public-key kerberos. *International Journal of Information Security*, 125, 2011.

[27] M. Backes, T. Chen, M. Dürmuth, H. P. A. Lensch, and M. Welk. Tempest in a teapot: Compromising reflections revisited. In *30th IEEE Symposium on Security and Privacy (S&P 2009), 17-20 May 2009, Oakland, California, USA*, pages 315–327, 2009.

[28] M. Backes, O. Ciobotaru, and A. Krohmer. Ratfish: A file sharing protocol provably secure against rational users. In *Computer Security - ESORICS 2010, 15th European Symposium on Research in Computer Security, Athens, Greece, September 20-22, 2010. Proceedings*, pages 607–625, 2010.

[29] M. Backes, G. Doychev, M. Dürmuth, and B. Köpf. Speaker recognition in encrypted voice streams. In *Computer Security - ESORICS 2010, 15th European Symposium on Research in Computer Security, Athens, Greece, September 20-22, 2010. Proceedings*, pages 508–523, 2010.

[30] M. Backes, P. Druschel, A. Haeberlen, and D. Unruh. Csar: A practical and provable technique to make randomized systems accountable. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2009, San Diego, California, USA, 8th February - 11th February 2009*, 2009.

[31] M. Backes, M. Dürmuth, S. Gerling, M. Pinkal, and C. Sporleder. Acoustic side-channel attacks on printers. In *19th USENIX Security Symposium, Washington, DC, USA, August 11-13, 2010, Proceedings*, pages 307–322, 2010.

[32] M. Backes, M. P. Grochulla, C. Hritcu, and M. Maffei. Achieving security despite compromise using zero-knowledge. In *Proceedings of the 22nd IEEE Computer Security Foundations Symposium, CSF 2009, Port Jefferson, New York, USA, July 8-10, 2009*, pages 308–323, 2009.

[33] M. Backes, M. P. Grochulla, C. Hritcu, and M. Maffei. Achieving security despite compromise using zero-knowledge. In *Proceedings of the Automated Reasoning for Security Protocol Analysis and Issues in the Theory of Security (ARSPA-WITS'09)*, 2011.

[34] M. Backes, M. Hamerlik, A. Linari, M. Maffei, C. Tryfonopoulos, and G. Weikum. Anonymity and censorship resistance in unstructured overlay networks. In *On the Move to Meaningful Internet Systems: OTM 2009, Confederated International Conferences, CoopIS, DOA, IS, and ODBASE 2009, Vilamoura, Portugal, November 1-6, 2009, Proceedings, Part I*, pages 147–164, 2009.

[35] M. Backes, D. Hofheinz, and D. Unruh. CoSP: a general framework for computational soundness proofs. In *Proceedings of the 2009 ACM Conference on Computer and Communications Security, CCS 2009, Chicago, Illinois, USA, November 9-13, 2009*, pages 66–78, 2009.

[36] M. Backes, C. Hritcu, and M. Maffei. Union and intersection types for secure protocol implementations. In *Theory of Security and Applications (TOSCA '11, formerly ARSPA-WITS)*, Saarbrücken, Germany, 2011. to appear.

[37] M. Backes, C. Hritcu, M. Maffei, and T. Tarrach. Type-checking implementations of protocols based on zero-knowledge proofs. In *Proceedings of the Workshop on Foundations of Computer Security (FCS'09)*, 2009.

[38] M. Backes, B. Köpf, and A. Rybalchenko. Automatic discovery and quantification of information leaks. In *30th IEEE Symposium on Security and Privacy (S&P 2009), 17-20 May 2009, Oakland, California, USA*, pages 141–153, 2009.

[39] M. Backes, S. Lorenz, M. Maffei, and K. Pecina. Anonymous webs of trust. In *Privacy Enhancing Technologies, 10th International Symposium, PETS 2010, Berlin, Germany, July 21-23, 2010. Proceedings*, pages 130–148, 2010.

[40] M. Backes, S. Lorenz, M. Maffei, and K. Pecina. Brief announcement: anonymity and trust in distributed systems. In *Proceedings of the 29th Annual ACM Symposium on Principles of Distributed Computing, PODC 2010, Zurich, Switzerland, July 25-28, 2010*, pages 237–238, 2010.

[41] M. Backes, M. Maffei, and E. Mohammadi. Computationally sound abstraction and verification of secure multi-party computations. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2010, December 15-18, 2010, Chennai, India*, pages 352–363, 2010.

[42] M. Backes, M. Maffei, and K. Pecina. A security API for distributed social networks. In *Proceedings to the Grande Region Security and Reliability Day (SecDay 2010)*, Saarbrücken, Germany, 2010.

[43] M. Backes, M. Maffei, and K. Pecina. Brief announcement: Securing social networks. In *Proceedings of the 30th Annual ACM Symposium on Principles of Distributed Computing, PODC 2011*, 2011.

[44] M. Backes, M. Maffei, and K. Pecina. A security API for distributed social networks. In *Proc. Network and Distributed System Security Symposium (NDSS'11)*, pages 35–51. Internet Society, 2011.

[45] M. Backes, M. Maffei, K. Pecina, and R. M. Reischuk. G2C: Cryptographic protocols from goal-driven specifications. In *Theory of Security and Applications (TOSCA '11, formerly ARSPA-WITS)*, Saarbrücken, Germany, 2011. to appear.

[46] M. Backes, M. Maffei, and D. Unruh. Computationally sound verification of source code. In *Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS 2010, Chicago, Illinois, USA, October 4-8, 2010*, pages 387–398, 2010.

[47] M. Backes, K. Pecina, and M. Maffei. Anonymous webs of trust. In *Proceedings of the Grande Region Security and Reliability Day (SecDay 2010)*, Saarbrücken, Germany, 2010.

[48] M. Backes and R. M. Reischuk. Cryptographic protocols from declarative specifications. In *Proceedings of the Grande Region Security and Reliability Day (SecDay 2010)*, Saarbrücken, Germany, 2010.

[49] M. Backes and D. Unruh. Computational soundness of symbolic zero-knowledge proofs. *Journal of Computer Security*, 18(6):1077–1155, 2010.

[50] N. Benton and C.-K. Hur. Biorthogonality, step-indexing and compiler correctness. In *ICFP*, 2009.

[51] N. Benton, C.-K. Hur, A. Kennedy, and C. McBride. Strongly typed term representations in Coq. *Journal of Automated Reasoning*, 2011. Special issue on binding, substitution and naming. To appear.

[52] L. Bergstrom, M. Fluet, M. Rainey, J. Reppy, and A. Shaw. Lazy tree splitting. In *ICFP 2010*, pages 93–104, New York, NY, USA, Sept. 2010. ACM Press.

[53] P. Bhatotia, A. Wieder, I. E. Akkus, R. Rodrigues, and U. Acar. Large-scale incremental data processing with change propagation. In *2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud '11)*, June 2011.

[54] P. Bhatotia, A. Wieder, R. Rodrigues, U. Acar, and R. Pasquini. Incoop: MapReduce for incremental computations. Under submission to the 2nd ACM Symposium on Cloud Computing (SOCC 2011).

[55] P. Bhatotia, A. Wieder, R. Rodrigues, F. Junqueira, and B. Reed. Reliable data-center scale computations. In *Proceedings of the 4th International Workshop on Large Scale Distributed Systems and Middleware (LADIS '10)*, pages 1–6, July 2010.

[56] M. Blume, U. A. Acar, and W. Chae. Extensible programming with first-class cases. In *Proceedings of the 11th ACM SIGPLAN International Conference on Functional Programming*, pages 239–250. ACM Press, 2006.

[57] R. Bose and J. Frew. Lineage retrieval for scientific data processing: a survey. *ACM Comput. Surv.*, 37(1):1–28, 2005.

[58] S. Brookes. A semantics for concurrent separation logic. *Theoretical Computer Science*, 375(1-3):227–270, 2007.

[59] M. Cha, H. Haddadi, F. Benevenuto, and K. P. Gummadi. Measuring User Influence in Twitter: The Million Follower Fallacy. In *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media (ICWSM)*, Washington, DC, May 2010.

[60] M. Cha, A. Mislove, B. Adams, and K. P. Gummadi. Characterizing Social Cascades in Flickr. In *Proceedings of the 1st ACM SIGCOMM Workshop on Social Networks (WOSN)*, August 2008.

[61] M. Cha, A. Mislove, and K. P. Gummadi. A Measurement-driven Analysis of Information Propagation in the Flickr Social Network. In *Proceedings of the 18th Annual World Wide Web Conference (WWW)*, April 2009.

[62] A. Charguéraud. The locally nameless representation. *Journal of Automated Reasoning*, 2011. To appear.

[63] K. Chatterjee and R. Majumdar. Discounting in games across time scales. In *Gandalf 10*. ENTCS, 2010.

[64] Y. Chen, J. Dunfield, M. A. Hammer, and U. A. Acar. Implicit self-adjusting programming. Submitted, 2011.

[65] J. Cheney, A. Ahmed, and U. A. Acar. Provenance as dependency analysis. In *Proceedings of the Eleventh International Symposium on Database Programming*, September 2007.

[66] J. Cheney, A. Ahmed, and U. A. Acar. Provenance as dependency analysis. *Mathematical Structures in Computer Science (MSCS) Special Issue on Programming Language Interference and Dependence*, 2010. To appear.

[67] J. Cheney, L. Chiticariu, and W. C. Tan. Provenance in databases: Why, how, and where. *Foundations and Trends in Databases*, 1(4):379–474, 2009.

[68] D. Clark and S. Landau. The problem isn't attribution: it's multi-stage attacks. In *Proceedings of the Re-Architecting the Internet Workshop*, ReARCH '10, 2010.

[69] D. Clark and S. Landau. Untangling attribution. In *Proceedings of a Workshop on Deterring CyberAttacks: Informing Strategies and Developing Options for U.S. Policy*, 2010.

[70] B. Cook, A. Gupta, S. Magill, A. Rybalchenko, J. Simsa, S. Singh, and V. Vafeiadis. Finding Heap-Bounds For Hardware Synthesis. In *Formal Methods in Computer Aided Design (FMCAD)*, 2009.

[71] M. Dischinger, A. Haeberlen, I. Beschastnikh, K. P. Gummadi, and S. Saroiu. SatelliteLab: Adding Heterogeneity to Planetary-Scale Network Testbeds. In *Proceedings of the ACM SIGCOMM 2008*, Seattle, WA, August 2008.

[72] M. Dischinger, A. Haeberlen, K. P. Gummadi, and S. Saroiu. Characterizing Residential Broadband Networks. In *Proceedings of the ACM/USENIX Internet Measurement Conference (IMC)*, San Diego, CA, October 2007.

[73] M. Dischinger, M. Marcon, S. Guha, K. P. Gummadi, R. Mahajan, and S. Saroiu. Glasnost: Enabling Users to Detect Traffic Differentiation. In *Proceedings of the 7th Usenix Symposium on Networked Systems Design and Implementation (NSDI)*, San Jose, CA, April 2010.

[74] M. Dischinger, A. Mislove, A. Haeberlen, and K. P. Gummadi. Detecting BitTorrent Blocking. In *Proceedings of the 8th ACM/USENIX Internet Measurement Conference (IMC)*, Vouliagmeni, Greece, October 2008.

[75] C. Dovrolis, K. P. Gummadi, A. Kuzmanovic, and S. Meinrath. Measurement Lab: Overview and an Invitation to the Research Community. In *Proceedings of the ACM SIGCOMM Computer Communication Review (CCR), Editorial Section*, New Delhi, India, August 2010.

[76] D. Dreyer, A. Ahmed, and L. Birkedal. Logical step-indexed logical relations. In *IEEE Symposium on Logic in Computer Science (LICS)*, 2009. To appear.

[77] D. Dreyer, A. Ahmed, and L. Birkedal. Logical step-indexed logical relations. *Logical Methods in Computer Science*, 2011. Special issue devoted to selected papers from LICS 2009. To appear.

[78] D. Dreyer, K. Crary, and R. Harper. A type system for higher-order modules. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 2003.

[79] D. Dreyer, R. Harper, and M. M. T. Chakravarty. Modular type classes. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 2007.

[80] D. Dreyer, G. Neis, and L. Birkedal. The impact of higher-order state and control effects on local relational reasoning. In *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, 2010.

[81] D. Dreyer, G. Neis, A. Rossberg, and L. Birkedal. A relational modal logic for higher-order stateful ADTs. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 2010.

[82] D. Dreyer and A. Rossberg. Mixin' up the ML module system. In *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, 2008.

[83] J. Dunfield. Untangling typechecking of intersections and unions. In *Proceedings of the Workshop on Intersection Types and Related Systems (ITRS '10)*, volume 45 of *EPTCS*, pages 59–70, 2011.

[84] M. Emmi, R. Majumdar, and R. Manevich. Parameterized verification of transactional memories. In *PLDI 10: Programming Languages Design and Implementation*. ACM, 2010.

[85] M. Fiore and C.-K. Hur. On the mathematical synthesis of equational logics. *Logical Methods in Computer Science*, 2011. To appear.

[86] M. Fluet, M. Rainey, J. Reppy, and A. Shaw. Implicitly threaded parallelism in Manticore. *Journal of Functional Programming*, FirstView:1–40, 2011.

[87] P. Fonseca, C. Li, and R. Rodrigues. Finding complex concurrency bugs in large multi-threaded applications. In *Proc. of European Conference on Computer Systems (EuroSys 2011)"*, Apr. 2011.

[88] P. Fonseca, C. Li, V. Singhal, and R. Rodrigues. A study of the internal and external effects of concurrency bugs. In *DSN 2010 - 40th IEEE/IFIP International Conference on Dependable Systems and Networks*, June 2010.

[89] P. Fonseca, R. Rodrigues, A. Gupta, and B. Liskov. Full information lookups for peer-to-peer overlays. *IEEE Transactions on Parallel and Distributed Systems*, 20(9), Sept. 2009.

[90] B. Ford and J. Iyengar. Efficient cross-layer negotiation. In *Proceedings of the Eighth ACM Workshop on Hot Topics in Networks (HotNets-VIII)*, New York, NY, Oct 2009.

[91] P. Francis, X. Xu, H. Ballani, D. Jen, R. Raszuk, and L. Zhang. FIB Suppression with Virtual Aggregation. Internet Engineering Task Force (IETF) draft-ietf-grow-va-04, Feb 2011.

[92] P. Francis, X. Xu, H. Ballani, D. Jen, R. Raszuk, and L. Zhang. Auto-Configuration in Virtual Aggregation. Internet Engineering Task Force (IETF) draft-ietf-grow-va-auto-03.txt, Feb 2011.

[93] P. Francis, X. Xu, H. Ballani, R. Raszuk, and L. Zhang. Simple Virtual Aggregation (S-VA). Internet Engineering Task Force (IETF) draft-ietf-grow-simple-va-02, Mar 2011.

[94] F. Freitas, E. Marques, R. Rodrigues, C. Ribeiro, P. Ferreira, and L. Rodrigues. Verme: Worm containment in overlay networks. In *International Conference on Dependable Systems and Networks (DSN)*, June 2009.

[95] C. Gaither. Recording industry withdraws suit. Boston Globe, `http://www.boston.com/business/articles/2003/09/24/recording_industry_withdraws_suit/`.

[96] P. Ganty, R. Majumdar, and B. Monmege. Bounded underapproximations. In *CAV 10: Computer-Aided Verification*. Springer, 2010.

[97] R. Garcia, N. Preguica, and R. Rodrigues. Efficient middleware for byzantine fault-tolerant database replication. In *Proc. of European Conference on Computer Systems (EuroSys 2011)"*, Apr. 2011.

[98] R. Garret. A Time Machine time bomb. `http://rondam.blogspot.com/2009/09/time-machine-time-bomb.html`.

[99] S. Gaudin. Spoofing defense dissed by security experts. InformationWeek, `http://www.informationweek.com/news/security/cybercrime/showArticle.jhtml?articleID=189500626`, June 2006.

[100] J.-Y. Girard. *Interprétation fonctionelle et élimination des coupures de l'arithmétique d'ordre supérieur*. PhD thesis, Université Paris VII, 1972.

[101] G. Gonthier. Formal proof — the four-color theorem. *Notices of the AMS*, 55(11):1382–93, 2008.

[102] G. Gonthier, B. Ziliani, A. Nanevski, and D. Dreyer. How to make ad hoc polymorphism less ad hoc. Submitted for publication, Mar. 2011.

[103] C. L. Guernic and A. Girard. Reachability analysis of hybrid systems using support functions. In *CAV 09: Computer-Aided Verification*, volume 5643 of *Lecture Notes in Computer Science*, pages 540–554. Springer, 2009.

[104] R. Guerraoui, T. A. Henzinger, B. Jobstmann, and V. Singh. Model checking transactional memories. In *PLDI*, pages 372–382, 2008.

[105] S. Guha, K. Biswas, B. Ford, S. Sivakumar, and P. Srisuresh. NAT Behavioral Requirements for TCP. Internet Engineering Task Force (IETF) RFC 5382, Oct 2008.

[106] S. Guha, B. Cheng, and P. Francis. Challenges in Measuring Online Advertising Systems. In *Proceedings of the 2010 Internet Measurement Conference (IMC)*, 2010.

[107] S. Guha, B. Cheng, and P. Francis. Privad: Practical Privacy in Online Advertising. In *Procedings of USENIX NSDI*, 2011.

[108] S. Guha, A. Reznichenko, K. Tang, H. Haddadi, and P. Francis. Serving Ads from localhost for Performance, Privacy, and Profit. In *Proceedings of 9th IFIP conference on e-Business, e-Services, and e-Society*, Nancy, France, Sept. 2009.

[109] K. P. Gummadi, A. Mislove, and B. Krishnamurthy. Addressing the Privacy Management Crisis in Online Social Networks. In *The IAB Workshop on Internet Privacy, Position Paper*, Boston, MA, December 2010.

[110] H. Haddadi, S. Guha, and P. Francis. Not All Adware is Badware : Towards Privacy-Aware Advertising. In *Proceedings of ACM Hotnets09*, Oct. 2009.

[111] A. Haeberlen. A case for the accountable cloud. In *Proceedings of the 3rd ACM SIGOPS International Workshop on Large-Scale Distributed Systems and Middleware (LADIS'09)*, Oct. 2009.

[112] A. Haeberlen, P. Aditya, R. Rodrigues, and P. Druschel. Accountable virtual machines. In *Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation (OSDI'10)*, Oct. 2010.

[113] A. Haeberlen, M. Dischinger, K. P. Gummadi, and S. Saroiu. Monarch: A Tool to Emulate Transport Protocol Flows over the Internet at Large. In *Proceedings of the 6th ACM/USENIX Internet Measurement Conference (IMC)*, Rio de Janeiro, Brazil, October 2006.

[114] A. Haeberlen and P. Kuznetsov. The Fault Detection Problem. In *Proceedings of the 13th International Conference on Principles of Distributed Systems (OPODIS'09)*, Dec. 2009.

[115] C. Hall, K. Hammond, S. P. Jones, and P. Wadler. Type classes in Haskell. *ACM Transactions on Programming Languages and Systems*, 18:241–256, 1996.

[116] M. Hammer and U. A. Acar. Memory management for self-adjusting computation. In *ISMM*, pages 51–60, 2008.

[117] M. A. Hammer, U. A. Acar, and Y. Chen. CEAL: a C-based language for self-adjusting computation. In *Proceedings of the 2009 ACM SIGPLAN Conference on Programming Language Design and Implementation*, June 2009.

[118] M. A. Hammer, G. Neis, Y. Chen, and U. A. Acar. Self-adjusting stack machines. Submitted, 2011.

[119] R. Harper and M. Lillibridge. A type-theoretic approach to higher-order modules with sharing. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 1994.

[120] R. Harper and C. Stone. A type-theoretic interpretation of Standard ML. In *Proof, Language, and Interaction: Essays in Honor of Robin Milner*. MIT Press, 2000.

[121] M. Hayakawa. WORM Storage on Magnetic Disks Using SnapLock Compliance and SnapLock Enterprise. Technical Report TR-3263, Network Appliance, 2007.

[122] T. A. Henzinger, P.-H. Ho, and H. Wong-Toi. Hytech: A model checker for hybrid systems. *Software Tools for Technology Transfer*, 1(1-2):110–122, 1997.

[123] C.-K. Hur and D. Dreyer. A Kripke logical relation between ML and assembly. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 2011.

[124] C.-K. Hur, D. Dreyer, and V. Vafeiadis. Separation logic in the presence of garbage collection. In *IEEE Symposium on Logic in Computer Science (LICS)*, 2011. To appear.

[125] India News. Techie jailed due to Airtel mistake. `http://www.indiaenews.com/technology/20071103/78777.htm`, Nov. 2007.

[126] M. Jose, Y. Hu, L. He, and R. Majumdar. Rewiring for robustness. In *DAC 10: Design Automation Conference*. ACM, 2010.

[127] M. Jose, Y. Hu, and R. Majumdar. On power and fault-tolerant optimization in FPGA physical synthesis. In *ICCAD 10: International Conference on Computer-Aided Design*. ACM, 2010.

[128] M. Jose and R. Majumdar. Cause clue clauses: Error localization using maximum satisfiability. In *PLDI 11: Programming Languages Design and Implementation*. ACM, 2011.

[129] N. Klarlund. *Progress measures and finite arguments for infinite computations*. PhD thesis, Cornell University, 1990.

[130] G. Klein, J. Andronick, K. Elphinstone, G. Heiser, D. Cock, P. Derrin, D. Elkaduwe, K. Engelhardt, R. Kolanski, M. Norrish, T. Sewell, H. Tuch, and S. Winwood. seL4: formal verification of an operating-system kernel. *Communications of the ACM*, 53(6):107–115, 2010.

[131] B. Köpf and M. Dürmuth. A provably secure and efficient countermeasure against timing attacks. In *Proceedings of the 22nd IEEE Computer Security Foundations Symposium, CSF 2009, Port Jefferson, New York, USA, July 8-10, 2009*, pages 324–335, 2009.

[132] B. Köpf and A. Rybalchenko. Approximation and randomization for quantitative information-flow analysis. In *IEEE Computer Security Foundations Symposium (CSF)*, 2010.

[133] B. Köpf and G. Smith. Vulnerability bounds and leakage resilience of blinded cryptography under timing attacks. In *Proceedings of the 23rd IEEE Computer Security Foundations Symposium, CSF 2010, Edinburgh, United Kingdom, July 17-19, 2010*, pages 44–56, 2010.

[134] S. Kramer and A. Rybalchenko. A multi-modal framework for achieving accountability in multi-agent systems. In *ESSLLI Workshop on Logics in Security (LIS)*. 2010.

[135] D. Kravets. Defense planting seeds of doubt with RIAA jurors. Wired, `http://www.wired.com/threatlevel/2007/10/defense-plantin/`, 2007.

[136] J. Launchbury and S. L. Peyton Jones. State in Haskell. *LISP and Symbolic Computation*, 8(4):293–341, 1995.

[137] X. Leroy. A formally verified compiler back-end. *Journal of Automated Reasoning*, 43(4):363–446, 2009.

[138] R. Ley-Wild. *Programmable Self-Adjusting Computation.* PhD thesis, Department of Computer Science, Carnegie Mellon University, May 2010.

[139] R. Ley-Wild, U. A. Acar, and G. Blelloch. Non-monotonic self-adjusting computation, 2010. In preperation.

[140] R. Ley-Wild, U. A. Acar, and M. Fluet. A cost semantics for self-adjusting computation. In *Principles of Programming Languages*, 2009.

[141] R. Ley-Wild, M. Fluet, and U. A. Acar. Compiling self-adjusting programs with continuations. In *Proceedings of the International Conference on Functional Programming*, 2008.

[142] N. Lopes, J. Navarro, A. Rybalchenko, and A. Singh. Applying Prolog to develop distributed systems. In *Intl. Conf. on Logic Programming (ICLP)*, 2010.

[143] N. Lopes and A. Rybalchenko. Distributed and predictable software model checking. In *Intl. Conf. on Verification, Model Checking, and Abstract Interpretation (VMCAI)*, 2011.

[144] R. Majumdar, E. Render, and P. Tabuada. Robust discrete synthesis against unspecified disturbances. In *HSCC 11: Hybrid Systems Computation and Control.* ACM, 2011.

[145] R. Majumdar and I. Saha. Symbolic robustness analysis. In *RTSS 09: Real-Time Systems Symposium.* IEEE, 2009.

[146] M. Marcon, B. Viswanath, M. Cha, and K. P. Gummadi. Sharing Social Content from Home: A Measurement-driven Feasibility Analysis. In *Proceedings of the 21st International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, Vancouver, Canada, June 2011.

[147] B. Matthias, U. A. Acar, and W. Chae. Exception handlers as extensible cases. In *Proceedings of the 6th Asian Symposium on Programming Languages and Systems*, APLAS '08, pages 273–289, 2008.

[148] M. McConnell. Mike McConnell on how to win the cyber-war we're losing. The Washington Post, February 28, 2010.

[149] R. McNaughton. Infinite games played on finite graphs. *Annals of Pure and Applied Logic*, 65:149–184, 1993.

[150] A. Mendelzon, A. Ríos, and B. Ziliani. Swapping: a natural bridge between named and indexed explicit substitution calculi. In *International Workshop on Higher-Order Rewriting (HOR)*, 2010.

[151] A. G. Miklas, K. K. Gollu, K. K. Chan, S. Saroiu, K. P. Gummadi, and E. de Lara. Exploiting Social Interactions in Mobile Systems. In *Proceedings of the 9th International Conference on Ubiquitous Computing (UbiComp)*, Innsbruck, Austria, September 2007.

[152] G. Miller. A scientist's nightmare: Software problem leads to five retractions. *Science*, 314(5807):1856–1857, December 2006.

[153] A. Mislove, K. P. Gummadi, and P. Druschel. Exploiting Social Networks for Internet Search. In *Proceedings of the 5th ACM SIGCOMM Workshop on Hot Topics in Networks (HotNets)*, November 2006.

[154] A. Mislove, H. S. Koppula, K. P. Gummadi, P. Druschel, and B. Bhattacharjee. Growth of the Flickr Social Network. In *Proceedings of the 1st ACM SIGCOMM Workshop on Social Networks (WOSN)*, August 2008.

[155] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and Analysis of Online Social Networks. In *Proceedings of the 7th ACM/USENIX Internet Measurement Conference (IMC)*, October 2007.

[156] A. Mislove, A. Post, K. P. Gummadi, and P. Druschel. Ostra: Leverging trust to thwart unwanted communication. In *Proceedings of the 5th Usenix Symposium on Networked Systems Design and Implementation (NSDI)*, April 2008.

[157] A. Mislove, B. Viswanath, K. P. Gummadi, and P. Druschel. You are who you know: Inferring user profiles in Online Social Networks. In *Proceedings of the 3rd ACM International Conference of Web Search and Data Mining (WSDM'10)*, New York, NY, February 2010.

[158] K. Namjoshi. Certifying model checkers. In *CAV 01: Computer Aided Verification*, volume 2102 of *Lecture Notes in Computer Science 2102*, pages 2–13. Springer-Verlag, 2001.

[159] A. Nanevski, V. Vafeiadis, and J. Berdine. Structuring the verification of heap-manipulating programs. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 2010.

[160] G. Neis. Non-parametric parametricity. Master's thesis, Universität des Saarlandes, 2009.

[161] G. Neis, D. Dreyer, and A. Rossberg. Non-parametric parametricity. In *ACM SIGPLAN International Conference on Functional Programming (ICFP)*, 2009.

[162] G. Neis, D. Dreyer, and A. Rossberg. Non-parametric parametricity. *Journal of Functional Programming*, 2011. Special issue devoted to selected papers from ICFP 2009. To appear.

[163] P. O'Hearn. Resources, concurrency and local reasoning. *Theoretical Computer Science*, 375(1-3):271–307, 2007.

[164] M. Piatek, T. Kohno, and A. Krishnamurthy. Challenges and directions for monitoring P2P file sharing networks. In *Proc. HotSec*, July 2008.

[165] A. Pitts and I. Stark. Operational reasoning for functions with local state. In *Higher-Order Operational Techniques in Semantics (HOOTS)*, 1998.

[166] G. D. Plotkin and M. Abadi. A logic for parametric polymorphism. In *International Conference on Typed Lambda Calculi and Applications (TLCA)*, 1993.

[167] A. Post, J. Navarro, P. Kuznetsov, and P. Druschel. Autonomous storage management for personal devices with podbase. In *Proceedings of the 2011 USENIX Annual Technical Conference (USENIX ATC'11)*, Portland, OR, June 2011.

[168] A. Post, V. Shah, and A. Mislove. Bazaar: Strengthening user reputations in online marketplaces. In *Proceedings of the 8th Symposium on Networked Systems Design and Implementation (NSDI'11)*, Boston, MA, March 2011.

[169] F. Pottier and V. Simonet. Information flow inference for ML. *ACM Trans. Prog. Lang. Sys.*, 25(1):117–158, Jan. 2003.

[170] J. C. Reynolds. Separation logic: A logic for shared mutable data structures. In *IEEE Symposium on Logic in Computer Science (LICS)*, pages 55–74, 2002.

[171] R. Rodrigues and P. Druschel. Peer-to-peer systems. *Communications of the ACM*, 53:72–82, October 2010.

[172] R. Rodrigues, B. Liskov, K. Chen, M. Liskov, and D. Schultz. Automatic reconfiguration for large-scale reliable storage systems. *Transactions on Dependable and Secure Computing*, Sept. 2010. PrePrint ISSN: 1545-5971.

[173] T. Rodrigues, J. An, F. Benevenuto, M. Cha, and K. P. Gummadi. An Analysis of Word-of-mouth based Discovery of the Web. In preparation for submission to the 11th ACM SIGCOMM/Usenix Internet Measurement Conference (IMC 2011).

[174] J. Rosenblatt. Fully automated DMCA processing. Presentation at the Security Professionals Conference (SEC), 2009.

[175] A. Rossberg. Generativity and dynamic opacity for abstract types. In *ACM SIGPLAN Symposium on Principles and Practice of Declarative Programming (PPDP)*, 2003.

[176] A. Rossberg. Dynamic translucency with abstraction kinds and higher-order coercions. In *Conference on the Mathematical Foundations of Programming Semantics (MFPS)*, 2008.

[177] A. Rossberg, C. V. Russo, and D. Dreyer. F-ing modules. In *ACM SIGPLAN Workshop on Types in Language Design and Implementation (TLDI)*, 2010.

[178] P. Roy, P. Tabuada, and R. Majumdar. Pessoa 2.0: A controller synthesis tool for cyber-physical systems. In *HSCC 11: Hybrid Systems Computation and Control.* ACM, 2011.

[179] C. V. Russo. *Types For Modules.* PhD thesis, LFCS, University of Edinburgh, 1998.

[180] A. Rybalchenko and R. Singh. Subsumer-First: Steering Symbolic Reachability Analysis. In *SPIN Workshop on Model Checking of Software (SPIN)*, 2009.

[181] A. Sabelfeld and A. C. Myers. Language-based information-flow security. *IEEE J. Selected Areas in Communications*, 21(1), 2003.

[182] A. Saibi. Typing algorithm in type theory with inheritance. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 1997.

[183] N. Santos, K. P. Gummadi, and R. Rodrigues. Towards trusted cloud computing. In *Proc. Workshop on Hot Topics in Cloud Computing (HotCloud '09)*, San Diego, CA, USA, June 2009.

[184] N. Santos, H. Raj, S. Saroiu, and A. Wolman. Trusted language runtime (tlr): Enabling trusted applications on smartphones. In *ACM HotMobile 2011, the twelfth Workshop on Mobile Computing Systems and Applications*, Mar. 2011.

[185] N. Santos, R. Rodrigues, K. P. Gummadi, and S. Saroiou. Excalibur: Gaining customer trust in cloud services. Under submission to the 23rd ACM Symposium on Operating Systems Principles (SOSP 2011).

[186] V. Saxena, Y. Sabharwal, and P. Bhatotia. Performance evaluation and optimization of random memory access on multicores with high productivity. In *The 17th annual IEEE International Conference on High Performance Computing (HiPC 2010)*, Dec. 2010.

[187] J. Sevcik, V. Vafeiadis, F. Zappa Nardelli, S. Jagannathan, and P. Sewell. Relaxed-memory concurrency and verified compilation. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 2011.

[188] Y. Simmhan, B. Plale, and D. Gannon. A survey of data provenance in e-science. *SIGMOD Record*, 34(3):31–36, 2005.

[189] A. Singh, P. Fonseca, P. Kuznetsov, R. Rodrigues, and P. Maniatis. Zeno: Eventually consistent byzantine fault tolerance. In *Proc. 6th USENIX Symposium on Networked Systems Design and Implementation (NSDI '09)*, Boston, USA, Apr. 2009.

[190] P. Srisuresh, B. Ford, S. Sivakumar, and S. Guha. NAT Behavioral Requirements for ICMP. Internet Engineering Task Force (IETF) RFC 5508, Apr 2009.

[191] J. Strauss, C. Lesniewski-Laas, J. M. Paluska, B. Ford, R. Morris, and F. Kaashoek. Device transparency: A new model for mobile storage. In *SOSP Workshop on Hot Topics in Storage and File Systems (HotStorage '09)*, Big Sky, MT, Oct 2009.

[192] O. Sumer, U. A. Acar, A. Ihler, and R. Mettu. Fast parallel and adaptive updates for dual-decomposition solvers. In *Proceedings of the Conference on Artificial Intelligence (AAAI)*, 2011.

[193] E. Sumii and B. Pierce. A bisimulation for type abstraction and recursion. *J. ACM*, 54(5):1–43, 2007.

[194] R. Sylvester. IP address typo leads to a false arrest in Kansas. The Wichita Eagle, `http://www.kansas.com/mld/eagle/news/local/crime_courts/12620843.htm`.

[195] Time Machine. `http://www.apple.com/macosx/features/timemachine.html`.

[196] V. Vafeiadis. Concurrent separation logic and operational semantics. Submitted to MFPS, 2011.

[197] V. Vafeiadis and F. Zappa Nardelli. Verifying fence elimination optimisations. Submitted to SAS, 2011.

[198] B. Viswanath, A. Clement, M. Mondal, P. Druschel, K. P. Gummadi, and A. Mislove. Beyond Good and Evil: Building Social Network-based Sybil-tolerant Systems. Under submission to the ACM SIGCOMM 2011.

[199] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi. On the evolution of user interaction in Facebook. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Social Networks (WOSN'09)*, August 2009.

[200] B. Viswanath, A. Post, K. P. Gummadi, and A. Mislove. An Analysis of Social Network-based Sybil Defenses. In *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication (SIGCOMM'10)*, New Delhi, India, August 2010.

[201] D. Vytiniotis, G. Washburn, and S. Weirich. An open and shut typecase. In *ACM SIGPLAN Workshop on Types in Language Design and Implementation (TLDI)*, 2005.

[202] P. Wadler and S. Blott. How to make ad-hoc polymorphism less ad hoc. In *ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 1989.

[203] A. Wieder. Accountable web applications. Master's thesis, Saarland University, 2010. Thesis advisor: Rodrigo Rodrigues.

[204] A. Wieder, P. Bhatotia, A. Post, and R. Rodrigues. Brief announcement: Modelling MapReduce for optimal execution in the cloud. In *Proceeding of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing (PODC '10)*, pages 408–409, July 2010.

[205] A. Wieder, P. Bhatotia, A. Post, and R. Rodrigues. Conductor: orchestrating the clouds. In *Proceedings of the 4th International Workshop on Large Scale Distributed Systems and Middleware (LADIS '10)*, pages 44–48, July 2010.

[206] Windows Backup and Restore. `http://www.microsoft.com/windows/windows-7/features/backup-and-restore.aspx`.

[207] H. Yang. Relational separation logic. *Theoretical Computer Science*, 375(1–3):308–334, 2007.

[208] L. Ziarek, S. Jagannathan, M. Fluet, and U. A. Acar. Speculative n-way barriers. In *DAMP '09: Proceedings of the 4th Workshop on Declarative Aspects of Multicore Programming*, pages 1–12, 2008.

[209] W. Zielonka. Infinite games on finitely coloured graphs with applications to automata on infinite trees. *Theor. Comput. Sci.*, 200(1-2):135–183, 1998.