Overview of Research, Teaching, and Service at UCLA (2014-2018)

Miryung Kim (miryung@cs.ucla.edu)

Highlights. I published 27 publications since 2014, where 20 appear in top tier venues (ICSE, ASE, VLDB, CHI, SoCC, SIGMOD, and TSE): 11 full, 4 journal, and 5 short papers. In terms of my career total, I published 67 papers, where 42 appear in top tier venues (ICSE, FSE, ASE, VLDB, CHI, SoCC, PLDI, OOPSLA, SIGMOD and TSE): 24 full, 5 journal, and 13 short papers.

My philosophy is to encourage students to focus on producing top quality research. As a result, my group produced 3 professors in US—Baishakhi Ray (University of Virginia ⇒ Columbia University, New York), Na Meng (Virginia Tech), and Myoungkyu Song (University of Nebraska, Omaha). In terms of industry impact, Huawei recently tech-transferred my group's work on interactive code clone search, Broad Institute is trying our big data analytics debugging technologies, and PepperData tech-transferred the key ideas of our Apache Spark debugger. I received 6 new grants, 3 new gifts, and 3 transfered grants since 2014—8M in total, where my direct cost share is 3.5M—including a 4.9M ONR grant of which I am a PI.

Research and Scholarship

My research focuses on software engineering (SE). I design automated software analysis and development tools to improve developer productivity and software correctness. I conduct user studies with professional software engineers and carry out statistical analysis of open source projects to allow data-driven decisions for designing novel SE tools.

Over the last five years, my role has evolved beyond producing research papers in top-tier SE venues. I have assembled a cross-disciplinary research team (software engineering and database systems) to address urgent research challenges in the era of data science and to pioneer a new sub-field on SE tools for big data analytics. I have conducted scientific, comprehensive studies of professional data scientists and disseminated our findings in both academia and industry.

Empirical Studies of Data Scientists in Software Teams

I initiated academia and industry coalition to investigate the emerging role of data scientists. Data scientists are becoming popular in software industry, e.g., Facebook, LinkedIn, and Microsoft, as software companies produce large data on user behavior, machine usage, and software quality metrics. To understand this emerging role, in collaboration with Microsoft Research, I conducted the first in-depth interview study of data scientists across several product groups at Microsoft. I investigated their education and training background, their missions in software engineering contexts, and the type of problems on which they work [ICSE 16b].

Building on this work, we conducted the largest scale, the most comprehensive study of almost 800 professional data scientists. We clustered these data scientists based on the time spent for various activities and identified 9 distinct clusters, and their corresponding characteristics [TSE 18]. This quantification and sub-categorization of data scientists is important, because although many companies are investing in data science and universities are creating data science graduate programs, we lack scientific, empirical understandings of what challenges data scientists face, what problems they work on, what tools they use, and how they fit in within organizations. Data scientist workforce is the fastest growing job market in US, and my studies should inform how organizations should employ and train this new workforce effectively to improve their productivity. These studies were published at ICSE 2016 and TSE Journal First. *ICSE is the top software engineering conference and TSE is No.1 Journal in software engineering*. I am the first author of both papers. I also disseminated our findings to data science practitioners in industry [Strata].

Interactive and Automated Debugging for Big Data Analytics

I pioneered a new sub-field on SE tools for big data analytics by assembling a cross-disciplinary team of software engineering and database systems researchers with Tyson Condie. An abundance

of data in science, engineering, national security, and health care has led to the emerging field of *big data analytics* and has accelerated the adoption of distributed big data system technologies such as Apache Hadoop and Apache Spark. However, the current big data computing model lacks the kinds of interactive debugging features found in traditional desktop computing, forcing data scientists to debug their applications by trial and error. Enabling interactive debugging on the cloud is challenging, because pausing the entire computation reduces throughput (efficiency). It is clearly infeasible for a user to inspect billion of records at real time. Even launching remote debuggers to individual workers cannot scale for big data computing.

To address these challenges, we designed a set of novel interactive debugging primitives for big data computing, called BIGDEBUG [ICSE 16a]. It emulates a breakpoint through incremental replay from the latest materialization point and streams selected program states on demand by leveraging dynamic guard compilation and deployment. BIGDEBUG also enables a user to identify the origin of faulty outputs by re-architecting the underlying big data runtime with *data provenance* [VLDB 15; VLDBJ 17]. Replay debugging can be optimized through *incremental computation* [SoCC 16].

When a failure, incorrect result, or outlier is generated, the programmer may want to pinpoint its root cause. To reduce the burden of manual debugging, we designed BIGSIFT [SoCC 17], a new automated debugging technique that combines delta debugging (DD) and data provenance (DP) with several key optimizations. Given a test oracle function, it automatically finds a minimum subset of input records responsible for a faulty output. In comparison to state-of-the arts, BIGSIFT improves fault localizability by $\sim 10^3$ to 10^7 and reduces the debugging time as much as $66\times$.

My student Muhammad Gulzar and I led BIGDEBUG and BIGSIFT, which were published in ICSE 2016 and SoCC 2017. Other collaborative works were published in VLDB and SoCC [VLDB 15; SoCC 16; VLDBJ 17]. *ICSE is the top software engineering conference, SoCC is the top conference on cloud computing, and VLDB is the top database conference.* I am the PI of an NSF project on Interactive and Automated Debugging for Big Data Analytics.

Coping with Code Duplication in Software Systems

My foundational work on code duplication has enabled me to lead a new research team to address *software debloating* and *delayering*, which must be urgently addressed to secure our nation's cyber infrastructure. I am the PI of an Office of Naval Research (ONR) project, Synergistic Software Customization (4.9M, co-PIs: Harry Xu and Jens Palsberg). Below are the details on code clone search, testing, and removal.

Code duplication created by copy and paste is common in large software. To help developers ensure that they applied similar changes to all relevant locations, my students and I designed an interactive code search approach, called CRITICS [ICSE 15b]. When a developer selects a sub region within a software patch, it allows a developer to interactively create an *abstract diff template* by parameterizing the edit content. By matching the template against the rest of the program, it then finds code clones and detects anomalies. We evaluated CRITICS through user studies at Salesforce.com. Human subjects using CRITICS answered questions about similar changes more correctly with more time saving. *This work was published in the top software engineering conference [ICSE 15b].*

Developers often find it difficult to check behavioral differences between similar code. The problem is exacerbated when some clones are tested, while counterparts are not. We designed an automated code transplantation and differential testing technique to compare behavioral differences between clones [ICSE 17], called GRAFTER. It was demonstrated in the Air Force project on cyber-physical system repair. *This work was published in the top software engineering conference [ICSE 17].*

When developers perform similar changes to code clones, this repetitive editing may present an opportunity to reduce code duplication. Extracting a reusable abstraction from identical code is done in a straightforward manner; however, the challenge lies when code clones have variations in type usages, method call targets, and control flow structures. My students and I investigated a new automated clone removal refactoring approach that can handle *type variations*, *method variations*,

variable, expression variations, *multiple variables to return*, and *non-local jump statements* [ICSE 15a]. This is the most advanced clone removal refactoring algorithm, pushing the limit of reducing code duplication. *This work was published in the top software engineering conference [ICSE 15a].*

Mining, Visualizing, and Assessing Code Examples at Scale

I created a new research thrust between software engineering and human computer interaction to tackle the new frontier of mining software repositories research—*usability* and *information delivery*.

There is a growing interest in leveraging large collections of open-source repositories such as GitHub. The hypothesis here is that common application programming interface (API) usage inferred from a large code corpus may represent a desirable pattern that a programmer must follow. We designed a new API usage mining technique, called EXAMPLECHECK [ICSE 18] that captures the temporal ordering of API calls, the enclosing control structures (i.e., exceptions being caught), and the guard conditions (or predicates) protecting these API calls. Using the patterns mined from 380K Java repositories on GitHub, we investigated API misuse on Stack Overflow, a popular online Q&A forum for software development. We found that one third of SO snippets may have potential API usage violations that could produce program crashes and resource leaks. Surprisingly, highly-voted posts are not necessarily more reliable than posts with fewer votes. *This work was published in the top software engineering conference [ICSE 18].*

Despite this growing interest of mining a large collection of open source repositories, there is no easy way for a user to understand the commonalities and variances among a massive number of related code examples. In collaboration with UC Berkeley, we designed a novel interactive visualization, called Examplore [CHI 18] that summarizes hundreds of code examples in one synthetic code skeleton. It also displays statistical distributions for canonicalized statements, while allowing a user to drill down to concrete details. My student Tianyi Zhang is an equal first author. *This work was published in the top human computer interaction conference [CHI 18].*

Software Tools and Industry Impact

As a software engineering researcher, I believe we should go extra miles in packaging and hardening innovative software analysis technologies so that they are usable by software professionals. My group has demonstrated our tools at premier venues for formal research demonstrations [ICSE Demo 14; FSE Demo 14a; FSE Demo 14b; FSE Demo 16; SIGMOD Demo 17]. Huawei recently tech-transferred CRITICS [ICSE 15b; FSE Demo 14a] as an internal clone search tool. This attests to the fact that our tool scales to industry projects and is easily usable by practitioners. The Broad Institute for Genomics Research is trying BIGDEBUG and BIGSIFT [ICSE 16a; SoCC 17] for Genome Analysis Toolkit written in Apache Spark. PepperData tech-transferred the key ideas of BIGDEBUG.

Funding

My research is supported by competitive, peer-reviewed funding sources. I am the PI of Office of Naval Research, Synergistic Software Customization (4.9M) and NSF Interactive and Automated Debugging for Big Data Analytics (900K). In addition, I received 4 new grants as a co-PI (NSF, Intel, NSF, and Air Force) and 3 new gifts as a PI (Huawei, Google, and Okawa Foundation) since 2014.

Future Work

Building on my work on SE for big data analytics, I would like to focus on the problem of designing, debugging, and testing complex AI/MI systems. Current artificial intelligence (AI) and machine learning (ML) technologies are not sufficiently democratized, and building complex AI and ML systems requires trial and error exploration for model selection, data cleaning, feature selection, and parameter tuning. I submitted a Dagstuhl proposal, "SE4ML – Software Engineering for AI-ML-based Systems" to bring SE and ML communities to work together on this important challenge.

Teaching and Mentoring

Undergraduate. I revamped an existing undergraduate course to emphasize systematic engineering methods for large-scale software development. CS 130 is a required class, with both a heavy lecture component and a capstone project component. The revamped class requires students to understand various object-oriented design patterns, grasp the foundations of software verification (Hoare logic and weakest preconditions), and write tests using symbolic execution and path condition analysis. CS 130 team projects produced many start-up worthy ideas. I invited guest panelists from industry and students appreciated receiving feedback from industry. The enrollment from 2015 to 2018 was 87, 85, 113, and 123. My overall instructor rating was 8.06, 6.81, 7.51, and 7.67.

Graduate. Prior to my arrival, UCLA did not offer a graduate class in software engineering. I created a new graduate course to teach the foundation of automated software engineering technology. My instructor rating was 8.33, 8.26, and 8.81.

To teach new advances in tools and environments for developing big data analytics, I designed CS 239 Data Science in Software Engineering. It covers mining software repositories and software metrics analysis techniques together with necessary statistics, data mining, and big data systems. My overall instructor rating was 7.62 and 8.27. I created a new graduate seminar to discuss recent advances in tools and environments for developing big data analytics with focus on software tooling, environments, and system stacks. The instructor rating is 8.2.

My average instructor rating between 2014 and 2018 is 7.97, while the department average in the same period is 7.63.

Student Advising and Mentoring. My philosophy is to encourage students to focus on producing top quality research and to become the next generation leaders in computing. As a result, I produced three tenure-track assistant professors. This is an exceptional faculty placement record, given my career stage: Baishakhi Ray (PhD 2013, University of Virginia $\rightarrow$ Columbia University), Na Meng (PhD 2014, Virginia Tech), and Myoungkyu Song (Postdoc 2015, U of Nebraska, Omaha).

I have been successful at creating a strong pipeline of graduate student researchers at UCLA: Tianyi Zhang (PhD student, started in 2013), Muhammad Gulzar (PhD student, started in 2014), Shaghayegh Mardani (PhD student, started in 2015), Jia Teoh (PhD student, started in 2016), Aishwarya Sivaraman (PhD student, started in 2017), Li Rong (PhD student, co-supervised with Van den Broeck, will start in 2018), and Bobby Bruce (Postdoc, will start in 2018). My mentoring helped students to produce a strong publication record, and their achievements are recognized by a prestigious ACM Student Research Competition Gold Medal at ICSE 2018 (Muhammad Gulzar), Google PhD Fellowship (Muhammad Gulzar), and Google Outstanding Graduate Student Research Award from UCLA CS (Tianyi Zhang).

Service

Department Service. The 8 year department review for Computer Science in 2014 found that diversity is a major area that needs improvement. By the request of former department chairs, I undertook the responsibility of *CS Diversity Committee Chair* to improve the diversity climate within my department. This is not an adhoc, transient, side activity. *It is an officially recognized department committee responsibility with long-term service commitment and organized, substantial effort.*

Under my leadership as a diversity committee chair, we achieved the following tangible outcomes: (1) a new undergraduate introductory programming class was created to make it easier for students without prior programming experience to enter CS; (2) sending female undergraduates to the Grace Hopper Celebration of Women in Computing conference was institutionalized as an annual program; (3) a local chapter of Association for Computing Machinery (ACM)-W was formed and I have been a faculty sponsor for this female student group; (4) a new distinguished speaker series was created to inspire women and under-represented students; and (5) I led the department-wide

proposal to join the BRAID (Building, Recruiting And Inclusion for Diversity) affiliates. UCLA was 8 out of 22 universities selected for this national STEM diversity program in 2016.

University Service. I serve as a Faculty-In-Residence for UCLA Office of Residential Life to help students' intellectual, social, and personal development. This is a 3 year-term, serious service commitment (12 hours per week on average) to design programming for student engagement such as panels, academic seminars, and career mentoring sessions for a broader population of UCLA undergraduates. I received the Faculty/Staff of the Year Award in 2017, given by the National Residence Hall Honorary at UCLA, which recognizes outstanding faculty-in-residence mentors who go above and beyond the call of duty. This honor was given to 1 out of 19 faculty-in-residence professors in 2017. I received Doc Stevensen Award of Outstanding Faculty-in-Residence in 2018.

Professional Service. I am an Associate Editor for IEEE Transactions on Software Engineering (No 1. journal in SE) and Journal of Empirical Software Engineering (No 2. journal in SE). I am a Program Co-Chair for ICSME 2019–the top software maintenance and evolution conference. I am a Program Co-Chair for Visions and Reflections Paper Track at FSE 2016. I am a General Chair for MSR 2016–MSR is the fastest growing conference in SE, and the most attended ICSE co-located event with 150 to 200 attendees. I am on the Program Board of ICSE 2016, 2017, and 2018. ICSE follows the program board model where PC members are responsible for reviews, while the PB members are similar to associate editors or area chairs.

I am a Senior Member of Association for Computing Machinery (ACM) since 2016 and I received an Okawa Foundation Research Award in 2015.

References

[CHI 18] Elena L. Glassman, Tianyi Zhang, Björn Hartmann, and Miryung Kim. "Visualizing API Usage Examples at Scale". In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. CHI '18. Montreal QC, Canada: ACM, 2018, 580:1–580:12.

[FSE Demo 14a] Tianyi Zhang, Myoungkyu Song, and Miryung Kim. "Critics: An Interactive Code Review Tool for Searching and Inspecting Systematic Changes". In: *Proceedings of the 22Nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*. FSE 2014. Hong Kong, China: ACM, 2014, pp. 755–758.

[FSE Demo 14b] Everton L. G. Alves, Myoungkyu Song, and Miryung Kim. "RefDistiller: A Refactoring Aware Code Review Tool for Inspecting Manual Refactoring Edits". In: *Proceedings of the 22Nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*. FSE 2014. Hong Kong, China: ACM, 2014, pp. 751–754.

[FSE Demo 16] Muhammad Ali Gulzar, Matteo Interlandi, Tyson Condie, and Miryung Kim. "BigDebug: interactive debugger for big data analytics in Apache Spark". In: *Proceedings of the 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering, FSE 2016, Seattle, WA, USA, November 13-18, 2016*. 2016, pp. 1033–1037.

[ICSE 15a] Na Meng, Lisa Hua, Miryung Kim, and Kathryn S. McKinley. "Does Automated Refactoring Obviate Systematic Editing?" In: *37th IEEE/ACM International Conference on Software Engineering, ICSE 2015, Florence, Italy, May 16-24, 2015, Volume 1*. 2015, pp. 392–402.

[ICSE 15b] Tianyi Zhang, Myoungkyu Song, Joseph Pinedo, and Miryung Kim. "Interactive Code Review for Systematic Changes". In: *37th IEEE/ACM International Conference on Software Engineering, ICSE 2015, Florence, Italy, May 16-24, 2015, Volume 1*. 2015, pp. 111–122.

[ICSE 16a] Muhammad Ali Gulzar, Matteo Interlandi, Seunghyun Yoo, Sai Deep Tetali, Tyson Condie, Todd Millstein, and Miryung Kim. "BigDebug: Debugging Primitives for Interactive Big Data Processing in Spark". In: *Proceedings of the 38th International Conference on Software Engineering*. ICSE '16. Austin, Texas: ACM, 2016, pp. 784–795.

[ICSE 16b] Miryung Kim, Thomas Zimmermann, Robert DeLine, and Andrew Begel. "The emerging role of data scientists on software development teams". In: *Proceedings of the 38th International Conference on Software Engineering, ICSE 2016, Austin, TX, USA, May 14-22, 2016*. 2016, pp. 96–107.

[ICSE 17] Tianyi Zhang and Miryung Kim. "Automated transplantation and differential testing for clones". In: *Proceedings of the 39th International Conference on Software Engineering, ICSE 2017, Buenos Aires, Argentina, May 20-28, 2017*. 2017, pp. 665–676.

[ICSE 18] Tianyi Zhang, Ganesha Upadhyaya, Anastasia Reinhardt, and Hridesh Rajan adn Miryung Kim. "Are Code Examples on an Online Q&A Forum Reliable? A Study of API Misuse on Stack Overflow". In: *Proceedings of the 40th International Conference on Software Engineering, ICSE 2018, Gothenburg, Sweden, May 27-June 3, 2018*. 2018, pp. 1–12.

[ICSE Demo 14] John Jacobellis, Na Meng, and Miryung Kim. "Cookbook: In Situ Code Completion Using Edit Recipes Learned from Examples". In: *Companion Proceedings of the 36th International Conference on Software Engineering*. ICSE Companion 2014. Hyderabad, India: ACM, 2014, pp. 584–587.

[SIGMOD Demo 17] Muhammad Ali Gulzar, Matteo Interlandi, Tyson Condie, and Miryung Kim. "Debugging Big Data Analytics in Spark with *BigDebug*". In: *Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference 2017, Chicago, IL, USA, May 14-19, 2017*. 2017, pp. 1627–1630.

[SoCC 16] Matteo Interlandi, Sai Deep Tetali, Muhammad Ali Gulzar, Joseph Noor, Tyson Condie, Miryung Kim, and Todd D. Millstein. "Optimizing Interactive Development of Data-Intensive Applications". In: *Proceedings of the Seventh ACM Symposium on Cloud Computing, Santa Clara, CA, USA, October 5-7, 2016*. 2016, pp. 510–522.

[SoCC 17] Muhammad Ali Gulzar, Matteo Interlandi, Xueyuan Han, Mingda Li, Tyson Condie, and Miryung Kim. "Automated debugging in data-intensive scalable computing". In: *Proceedings of the 2017 Symposium on Cloud Computing, SoCC 2017, Santa Clara, CA, USA, September 24 - 27, 2017*. 2017, pp. 520–534.

[Strata] Miryung Kim. *Who are we? The largest-scale study of professional data scientists*. `https://conferences.oreilly.com/strata/strata-ca/public/schedule/detail/65460`.

[TSE 18] M. Kim, T. Zimmermann, R. DeLine, and A. Begel. "Data Scientists in Software Teams: State of the Art and Challenges". In: *IEEE Transactions on Software Engineering* (2017), pp. 1–17.

[VLDB 15] Matteo Interlandi, Kshitij Shah, Sai Deep Tetali, Muhammad Ali Gulzar, Seunghyun Yoo, Miryung Kim, Todd D. Millstein, and Tyson Condie. "Titian: Data Provenance Support in Spark". In: *PVLDB* 9.3 (2015), pp. 216–227.

[VLDBJ 17] Matteo Interlandi, Ari Ekmekji, Kshitij Shah, Muhammad Ali Gulzar, Sai Deep Tetali, Miryung Kim, Todd Millstein, and Tyson Condie. "Adding data provenance support to Apache Spark". In: *The VLDB Journal* (2017).