Yan Wang

Computer Science & Engineering Department   Email: wangy@cs.ucr.edu   Mobile: (+1)513-680-9612
University of California, Riverside   http://www.cs.ucr.edu/~wangy/

INTERESTS

Dynamic/static program analysis with its applications on Debugging, Compiler Optimization, Testing, and Security for Parallel, Large-scale Computing, and Mobile Systems

EDUCATION& HONORS

Ph.D. in Computer Science (GPA: **3.87/4.0**) **Sep. 2009-Aug. 2014(Expected)**

University of California, Riverside, CA, USA Advisor: Prof. Rajiv Gupta and Prof. Iulian Neamtiu

Dean's Fellowship Award, UC Riverside, 2009-2010

M.S. in Computer Science (GPA: **90/100**) **Aug. 2006-March 2009**

Xidian University (exempted from Graduate Entrance Exam), Xi'an, China Advisor: Prof. Xiyang Liu

IBM Fellowship, IBM, 2008; Grade Scholarship (TOP 1%), Xidian University, 2007

B.S. in Computer Science (GPA: **93/100**) **Ranked 7 out of 420** **Xidian University**, Xi'an, China **Aug. 2002-July 2006**

Honorable Mention of The ACM Asia Programming Contest Beijing Site, Peking University, 2004

Second Prize and Best Female Player Prize of the 3rd Programming Contest, Xidian University, 2004

First Grade Lenovo Scholarship, 2004; Top Grade Scholarship (TOP 1%), Xidian University, 2003

PUBLICATIONS

Yan Wang, Harish Patil, Cristiano Pereira, Gregory Lueck, Rajiv Gupta, and Iulian Neamtiu, "DrDebug: Deterministic Replay based Cyclic Debugging with Dynamic Slicing", **CGO'14**, 11 pages, Orlando, 2014.

Yan Wang, Rajiv Gupta, and Iulian Neamtiu, "Relevant Inputs Analysis and its Applications", **ISSRE'13**.

Yan Wang, Min Feng, Rajiv Gupta, and Iulian Neamtiu, "A State Alteration and Inspection-based Interactive Debugger", **SCAM'13**, Pages 84-93, Eindhoven, Netherlands, September 2013.

Yan Wang, Iulian Neamtiu, and Rajiv Gupta, "Generating Sound and Effective Memory Debuggers",**ISMM'13**.

Dennis Jeffrey, **Yan Wang**, Chen Tian, and Rajiv Gupta, "Isolating Bugs in Multithreaded Programs Using Execution Suppression", *Software Practice & Experience* (**SP&E'11**).

Yan Wang, Zhiwen Bai, Miao Zhang, Wen Du, Ying Qin, Xiyang Liu, "Fitness Calculation Approach for the Switch-Case Construct in Evolutionary Testing", **GECCO'08**, **Nominated for Best Paper Awards**.

Xiyang Liu, Miao Zhang, Zhiwen Bai, Lei Wang, Wen Du, and **Yan Wang**. 2007. Function Call Flow based Fitness Function Design in Evolutionary Testing. **APSEC '07a**.

Xiyang Liu, Tao Liu, Zhiwen Bai, **Yan Wang,** Haoying Mu, Chunxiang Li, PORD: a Reversible Debugging Tool using Dynamic Binary Translation. **APSEC '07b**, poster.

PROJECT EXPERIENCE

Research Assistant Intel Corporation & University of California Riverside

DrDebug: Deterministic Replay based Cyclic Debugging with Dynamic Slicing **June 2012- Now**

- Developed an effective deterministic replay based cyclic, interactive debugger for multi-threaded programs, called DrDebug. With DrDebug, a precise dynamic slice can be computed and browsed by navigating the dynamic dependence edges with our GUI. Then the slice is used to compute an *execution slice* whose replay can be performed efficiently as execution of code segments that do not belong to the slice is skipped. The user can step from the execution of one statement in the slice to the next while examining the values of variables in a live debugging session **[CGO'14]**.

- Implemented in C++ with *Pin*, PinADX, PinPlay, *KDbg, and GDB* on the Linux environment (will be available on GitHub soon).

<u>**Research Assistant University of California Riverside**</u>

A Dynamic Analysis Framework and its Applications for Android System Nov. 2013- Now

- Constructing a dynamic program analysis framework for Android and then build two applications on top of it---dynamic slicing and relevant input analysis. The dynamic analysis framework supports efficient def-use tracing and online control dependence detection.
- Implemented in Java/C++ with *Dalvik VM, and Redexer* on the Android environment.

Relevant Inputs Analysis and its Applications June 2012- June 2013

- Developed *relevant input analysis*, which characterizes the *role* and *strength* of inputs in the computation of different values during a program execution. The *role* indicates whether a computed value is derived from an input value or is influenced by an input. The *strength* indicates if role relied upon the precise value of the input or is just among one of many values that can play a similar role. We leveraged the analysis to speedup the delta debugging algorithm **[ISSRE'13]**.
- Implemented in C++ with *Pin* on the Linux environment.

Generating Sound and Effective Memory Debuggers Oct. 2011- May 2012

- Developed a new approach for automatically constructing debuggers based on declarative specification of bug conditions and root causes. To facilitate bug localization, we introduced value propagation chain, which captures the propagation for erroneous value from its origin to the manifest point **[ISMM'13]**.
- Implemented in C++ with *Pin*, *Flex*, and *Bison* on the Linux environment.

A State Alteration and Inspection-based Interactive Debugger Aug. 2010- Sep. 2011

- Raised the abstraction level of debuggers by introducing high-level commands: state alteration commands for dynamically switching the directions of conditional branches and suppressing the execution of statements; state inspection commands including navigating and pruning dynamic slices [**SCAM'13]**.
- Implemented in C++ with *Pin*, *KDbg*, and *GDB* on the Linux environment.

A Record/Replay System for Multithreaded Program May 2010- June 2010

- Developed a recording/replay system for multithreaded program. The scheduling decisions of the *Valgrind* scheduler are logged during recording phase. During the replay phase, execution is replayed using the logged scheduling decisions.
- Implemented in C with *Valgrind* on the Linux environment (available on http://wet.cs.ucr.edu/download.html).

Isolating Bugs in Multithreaded Programs Using Execution Suppression Oct. 2009- June 2010

- Developed a general framework that can isolate the root cause of any failure in a multithreaded program that involves memory. We handled three important types of concurrency bugs—data races, atomicity violations, and order violations—as well as other kinds of memory bugs. *Execution suppression* is leveraged to iteratively reveal memory corruption in a failing execution to isolate the true root cause of the failure **[SP&E’11]**.
- Implemented in C with *Valgrind* on the Linux environment.

<u>**Research Assistant Xidian University**</u>

PORD: a Reversible Debugging Tool using Dynamic Binary Translation June 2007- March 2009

- Developed PORD on the X86/Linux platform, which provides a cross-platform reversible debugger for embedded systems. It enables embedded systems in ARM, SPARC, PPC or MIPS instruction set architectures to be debugged on a general-purpose X86 host architecture. PORD also implements an efficient reversible debugger for the X86 target architecture, which enjoys a near native speed **[APSEC’07b]**.
- Implemented in C with Qemu, and GDB on the Linux environment.

Test automation framework and its application on the Altimeter Software **Feb. 2006- Feb.2008**

- Performed white-box branch-coverage automatic test for altimeter software using evolutionary testing. I designed and implemented the fitness calculation module and constructed program dependence graph for C program, which was based on the tree parser generated by the open source parser generator *ANTLR*.
- Designed a Flattened Control Dependence Graph for the switch-case construct, and proposed a unified fitness calculation approach based on Alternative Critical Branches for the switch-case and other constructs, which provides much better guidance to evolutionary testing **[GECCO'08]**.
- Implemented in C++/Java with *ANTLR* on the Windows environment.

PROFESSIONAL SKILLS

Programming Language:

	Language: C/C++, Java, C#, OCaml, Assembly language for IA32	**Script**: Linux Shell, Python
	Parallel Programming: pthread, OpenMP, MPI, CUDA, MapReduce	**GUI design**: Qt, KDE
Tools:	**Binary translation and instrumentation**: Pin, Valgrind, Qemu	**Record and Replay**: PinPlay
	Source code translation and instrumentation: ANTLR, Flex, Bison	**Debuggers**: KDbg, GDB, PinADX
	Distributed Computing: Hadoop	**Database**: MySQL, SQL Server

TALKS

- "DrDebug: Deterministic Replay based Cyclic Debugging with Dynamic Slicing", CGO’14, Orlando, Florida, USA, February 2014.
- "Relevant Inputs Analysis and its Applications", SoCal Programing Languages and Systems Workshop, Los Angeles, CA, USA, November 2013.
- "A State Alteration and Inspection-based Interactive Debugger", SCAM'13, Eindhoven, The Netherlands, Sep. 2013
- "Generating Sound and Effective Memory Debuggers", ISMM’13, Seattle, Washington, USA, June 2013.
- “QuickZoom: A State Alteration and Inspection-based Interactive Debugger", SoCal Programing Languages and Systems Workshop, San Diego, CA, USA, Dec. 2011.
- “Debugger Syntheis”, SoCal Programing Languages and Systems Workshop, Irvine, CA, USA, April 2012.

TEACHING EXPERIENCE

Teaching assistance in ***Compiler Design***	University of California, Riverside	Winter 2012 & Winter 2013 & Winter 2014
Teaching assistance in ***Compiler Construction***	University of California, Riverside	Spring 2011 & Spring 2012

PROFESSIONAL ACTIVITIES

I helped review papers in ***architecture, compiler, programming language and software engineering***:

- **Journal:** ACM TOPLAS, IEEE Transaction on Computers and IEEE Transaction on Reliability
- **Conference:** ASPLOS, PLDI, ISCA, MICRO, POPL, PPoPP, PACT, CGO, ICSM, APLAS, ISMM, ISPASS, CASES, RV, IPDPS, QSIC, LCTES, SAMOS, INTERACT, and EXADAPT