Yin Wang

yinwang0@gmail.com

(415) 246-9859

1 Bayside Village Pl Suite 404, San Francisco, CA 94107

LinkedIn profile: http://www.linkedin.com/in/yinwang

Technical blog: http://yinwang0.wordpress.com

GitHub: http://github.com/yinwang0

SUMMARY

I graduated from Indiana University, where I worked with its Programming Languages Group (Dan Friedman, Kent Dybvig and Amr Sabry). They taught me how to design programming languages and compilers, and how to write beautiful programs.

I implemented interpreters for a wide variety of semantics, an optimizing compiler from Scheme to X64, a logic programming language and various type systems (Hindley-Milner system, intersection types etc). I (re)invented various things including a CPS transformer without administrative redexes and a fast register allocation algorithm. I extensively researched over 400 papers in the fields of programming languages and logics.

I interned twice at Google, where I built a static analyzer for Python which infers types from Python programs and builds semantic indexes. The system was built from almost complete scratch and is still in daily use processing Google's Python code. The first version of the analyzer was opensourced and merged into Jython.

My recent work at Coverity was to improve the Java Webapp Security product, an automatic tool for finding security bugs (SQLi, XSS etc) in web applications. I improved the product's accuracy and cut the memory usage by half on some benchmarks.

In my spare time I built ydiff, a language-aware program comparison tool. I share my thoughts on my blog. I'm also designing a programming language containing all the best things that I have learned.

SKILLS

- *Programming Languages*

  Assembly, C, C++, Java, JavaScript, Python, Scheme, Haskell, Ocaml, Common Lisp, Emacs Lisp, MATLAB, Mathematica, SQL, Perl, shell
- *Functional Programming (10 years of experience, implemented an optimizing Scheme compiler to X64, various type systems including Hindley-Milner and intersection types)*

- *Logic Programming (implemented logic programming language "miniKanren" and a constraint-based negation operator)*
- *Object-Oriented Programming (deeply understand design patterns and know how NOT to use most of them)*
- *Databases (SQL, NoSQL)*
- *Programming language theory (semantics, first-order logic, lambda calculus, combinatory logic, type systems, type theory, linear logic, Hoare logic)*
- *compiler construction and optimization, partial evaluation*
- *mechanical theorem proving (Coq, ACL2)*
- *parsing (built a parser combinator library, wrote parsers for C++, JavaScript and Lisp)*
- *Kernel level programming (built Linux and Windows device drivers)*
- *Linux development environment (15 years of experience)*
- *Graphics (built physically-based ray tracer and animation system)*
- *Computational geometry (implemented scan line algorithms for Voronoi diagram in Euclidean and L1 metrics)*
- *Documentation: TeX/LaTeX, Illustrator, Flash, PowerPoint, Keynote, HTML, CSS*
- *Others in cirruculum: computer architecture, data structures, algorithms, operating systems, databases, networks, AI, graphics, matrix computation, probability theory, applied logic, theory of computation, computational geometry, combinatorics, analog/digital circuits, linear/non-linear programming*

PORTFOLIO

Some of my best personal code (in Scheme, JavaScript, Python, Coq):

http://github.com/yinwang0

First version of the Python analyzer that I built at Google:

http://hg.python.org/jython/file/11776cd9765b/src/org/python/indexer

Some of my thoughts:

http://yinwang0.wordpress.com

WORK EXPERIENCE

Software Engineer, Coverity (Dec 2012 – June, 2013)

Java WebApp Security

Java WebApp Security is a fully automatic code analysis system that finds security problems such as SQLi, XSS etc.

- Worked on global dataflow graph and abstract interpretation
- Reduced false negatives and false positives by making the dataflow graph closerly model the semantics of the language
- Reduced memory usage by over 50% on some benchmarks by correcting the termination criteria of the abstract interpreter

Software Engineer Intern, Google (May 2010 – Aug 2010)

type inference for Python

- Improved and simplified the Python static analyzer that I built in 2009 (see below)
- Enhanced it with inter-procedural analysis

For technical details, see

http://yinwang0.wordpress.com/2010/09/12/pysonar

Software Engineer Intern, Google (May 2009 – Aug 2009)

semantic indexing for Python

- Built a static analyzer for Python which infers types from Python programs and builds semantic indexes
- Still in daily use processing all of Google's Python code
- Open-sourced and merged into Jython. Code available at:

  http://hg.python.org/jython/file/11776cd9765b/src/org/python/indexer

Software Engineer, UOneNet (Feb 2006 – Aug 2006)

large-scale multi-user online game engine

I helped with setting up the code and tools for this early-stage startup

EDUCATION

Indiana University

M.S. Computer Science (entered as a PhD student), Dec 2012, GPA 3.75

Research direction: programming languages

Advisor: Amr Sabry Committee: Amr Sabry, Dan Friedman, Kent Dybvig, Larry Moss

Studied: programming languages theory, compilers, databases, probability theory, algebra, applied logic, natural language processing

Taught: Java, databases, theory of computation (graduate)

Cornell University

M.S., Computer Science (entered as PhD student), May 2008, GPA 3.7

Studied: graphics rendering, matrix computation, programming languages

Taught: Java, algorithms (graduate)

Tsinghua University

Computer Science (entered as PhD student), 2001-2005, GPA 3.7

Research direction: EDA

Worked on VLSI routing algorithms. Won a *best paper award* at ASP-DAC.

Studied: VLSI design, computational geometry, algorithms, combinatorics, machine learning, linear/non-linear programming

Sichuan University

B.E., Computer Science, May 2001, GPA 3.6

Studied: data structures, algorithms, networks, operating systems, AI, analog/digital

circuits, discrete math, calculus, linear algebra, complex functions, physics, ...
Final project: unification of Linux and Windows device driver interfaces for an ADPCM card

Oregon Programming Languages Summer School
Participation, Logic, Languages, Compilation, and Verification, 2010

LIST OF SIGNIFICANT PROGRAMS

1. *optimizing compiler* from the Scheme programming language to X64
2. *type inferencer* for Python
3. one-pass *CPS and ANF transformers* with no administrative redexes
4. *online partial evaluator* for Scheme
5. *Linux kernel device driver* and *Windows device driver*
6. *parsers* for C++, JavaScript and Scheme
7. *structural differencer* which compare program by parse trees
8. advanced *type systems* (Hindley-Milner system, intersection types)
9. physically-based *ray tracer*
10. physically-based *animation engine*
11. logic programming language with constraint logic programming
12. interpreters for various programming languages

RESEARCH EXPERIENCE

Intersection types, session types and delimited continuations
Spring 2012 (with Amr Sabry)

Researched the area of type systems and type theory. Implemented a polar intersection type inference system

Non-graph-coloring register allocation methods
Fall 2011 (with R. Kent Dybvig)

Designed a semantic-based register allocation method, implemented in a Scheme compiler, wrote a paper draft

Reversible computing
Spring 2009 (with Amr Sabry)

Designed a reversible CEK abstract machine

Logic programming language extension
Fall 2008 (with Dan Friedman)

Reimplemented the logic language miniKanren, implemented universal quantification and constraint-based negation

PUBLICATIONS

- Most of my recent findings are written into technical blog posts and animated slides instead of academic papers*: http://yinwang0.wordpress.com*

- *Yin Wang, R. Kent Dybvig, Register Allocation By Model Transformer Semantics, draft, 2011. (arxiv:1202.5539)*
- *Yin Wang, Xianlong Hong, Tong Jing, Yang Yang, Xiaodong Hu, Guiying Yan: The polygonal contraction heuristic for rectilinear Steiner tree construction. ASP-DAC 2005: 1-6 (best paper award)*
- *Yin Wang, Xianlong Hong, Tong Jing, Yang Yang, Xiaodong Hu, Guiying Yan: An Efficient Low-Degree RMST Algorithm for VLSI/ULSI Physical Design. PATMOS 2004: 442-452*
- *Yang Yang, Qi Zhu, Tong Jing, Xianlong Hong and Yin Wang, "Rectilinear Steiner Minimal Tree among Obstacles", 5th IEEE International Conference on ASIC (ASICON'03), Beijing, China, October, 2003.*

AWARDS

Fellowship, Indiana University, 2006 (declined)
Best Paper Award, ASP-DAC 2005

LANGUAGES

English (proficient)
Mandarin Chinese (native speaker)