# Xilun Wu

Lawson CS Building, 305 N. University Street, West Lafayette, IN-47907 USA
wu636@purdue.edu, https://xilunwu.github.io/, +1 (765)430-2553

## RESEARCH INTEREST

Xilun is a sixth-year PhD student in Computer Science at Purdue University. His main research interest sits at exploring the intersection of **programming languages** and multiple application domains including **data management** and **machine learning**. Most of his projects are aimed at bringing high-level language features such as code reuse, abstractions, and modularity to system programming without losing performance with compiler techniques. He is also interested in type theory and category theory.

## EDUCATION

- **Purdue University, West Lafayette, USA**
  PhD Candidate, Computer Science 2015-2022 Expected
- **Purdue University, West Lafayette, USA**
  MS, Computer Science 2020
- **Zhejiang University, Hangzhou, China**
  BS, Computer Science and Technology 2014

## INDUSTRY EXPERIENCE

- **Software Engineer Intern, Facebook, Menlo Park, USA** **Summer 2019**
  - **Goal:** To add self-types in Hack language for two benefits:
    1. Supporting Cake Pattern for the construction of reusable components
    2. Replacing the existing facility for dependency injection in Hack language (i.e. trait and interface requirements) with self-types
  - **Job Description:** Involved in the technical design of adding self-types in Hack language. Modified the Hack type checker in all phases to support self-types. Removed the ad-hoc type checking for trait and interface requirements. Wrote a code mod for legacy code compatibility and tested the type check on the whole codebase for correctness and performance.
  - **Result:** Implemented a prototype that was able to correctly type check self-types. Significant speedup in type checking was observed on certain test cases due to the removal of the ad-hoc code.

## ACADEMIC EXPERIENCE

- **PhD Candidate, Purdue University, West Lafayette, USA**

  **2015-2020**

  Have been a member of Tiark Rompf research group. Conducted multiple research projects in team. They include:

  - **Lantern:** an expressive and performant **deep learning** framework
    - **Goal:** To develop a deep learning framework that combines the expressiveness of PyTorch and the execution performance of TensorFlow, and reveal the intrinsic relationship between backpropagation and the PL concept continuation.
    - **Job Description:** Implemented a machine learning framework in Scala powered by fusing delimited continuation and multi-staging together, to correspondingly achieve PyTorch-style expressiveness and TensorFlow-style graph reification.
    - **Result:** Benchmarked the performance of training a Recursive Neural Network model **TreeLSTM** in Lantern framework and it is more than **4x faster than** that in PyTorch (performant), while this model is **hard to code** in TensorFlow (expressive).
  - **LMS-NPRR:** native SQL **query compilation** with an efficient NPRR join algorithm implementation in single-core in-memory setting
    - **Goal:** To add Leapfrog Trie Join (a worst-case optimal join algorithm) in a single-core in-memory SQL query engine written in **500 LOC of Scala** by Tiark Rompf, and optimize the data representation to reach competitive performance as other state-of-the-art query engines on **graph queries**.
    - **Job Description:** Extended the mini SQL query engine with Leapfrog Trie Join algorithm as a special join operator. Adopted multiple data representations and searching algorithms and tested the performance of their combinations on TPC-H benchmark.
    - **Result:**
      1. The prototype was completed in less than **1K LOC of Scala** using the Light-weight Modular Staging library.
      2. Achieved competitive performance on TPC-H benchmark as EmptyHeaded, a state-of-the-art in-memory database system embedding NPRR join algorithm.
  - **Extending DOT:** extending the mechanized foundations for Scala and similar languages with full dependent types

- **Goal:** To extend DOT calculus the theoretical fundament of Scala language with full dependent types.
- **Job Description:** Investigated the strong normalization proof methods for Calculus of Construction and the possibility of adopting them for the extended DOT calculus.
- **Result:** Currently investigating into the possibility of encoding abstract type members as strong sigma types while not violating logic consistency by limiting the elimination rule.

- **Research Assistant, Purdue University, West Lafayette, USA Summer 2018**
  - Implemented compiler features in the framework of LLVM which primarily are adding optimization annotations/directives in a certain category of high-level programs (grid-based programs in C/C++) to guide the compiler backend to perform auto-parallelization as source-to-source code transformations.
- **Research Assistant, Purdue University, West Lafayette, USA 2015-2017**
  - Worked with Professor Rompf on LMS-NPRR.
- **Teaching Assistant, Purdue University, West Lafayette, USA 2017-2020**
  - CS35400 Operating Systems Fall 2020
  - CS52500 Parallel Computing Spring 2020
  - CS35400 Operating Systems Fall 2019
  - CS25200 Systems Programming June 2017 -May 2019
  - CS35200 Compilers: Principles And Practice Spring 2017
- **Master Student, Purdue University, West Lafayette, USA 2014-2015**

## PAST PROJECTS

- **Lantern:** an expressive and performant deep learning framework
  - Demystifying differentiable programming: Shift/reset the penultimate backpropagator. **ICFP'19**
  - Backpropagation with Callbacks: Foundations for Efficient and Expressive Differentiable Programming. **NeurIPS'18**
- **LMS-NPRR:** native query compilation with an efficient NPRR join algorithm implementation in single-core in-memory setting
  - Compiling Graph Queries in Relational Engines. **DBPL'19**
- **Extending DOT:** investigating the practicality of extending the mechanized foundations for Scala and similar languages with full dependent types

- Towards Full Dependent Types in DOT/Scala. Poster presentation on PurPL Grad Visit Day 2020.

## PRESENTATIONS

1. Poster presentation. Towards Full Dependent Types in DOT/Scala. PurPL Grad Visit Day 2020, West Lafayette IN, USA.
2. Poster presentation. Backpropagation with Callbacks: Foundations for Efficient and Expressive Differentiable Programming. **NeurIPS'18**, Montreal, Canada.
3. Poster presentation. A Language and Compiler View on Differentiable Programming. Huawei Midwest Research Summit 2018, Champaign IL, USA.

## AWARDS

- **Distinguished Artifact Award** at OOPSLA 2016 (international conference), Leo Osvald, Grégory Essertel, **Xilun Wu**, Lilliam I. Gonzalez Alayon, Tiark Rompf.

## PUBLICATIONS

1. Fei Wang, Daniel Zheng, James Decker, **Xilun Wu**, Grégory M Essertel, Tiark Rompf. Demystifying differentiable programming: Shift/reset the penultimate backpropagator. ICFP 2019, 31 pages.
2. Ruby Tahboub, **Xilun Wu**, Grégory Essertel, Tiark Rompf. Compiling Graph Queries in Relational Engines. DBPL 2019, 12 pages.
3. Fei Wang, James Decker, **Xilun Wu**, Grégory Essertel, Tiark Rompf. Backpropagation with Continuation Callbacks: Towards Efficient and Expressive Differentiable Programming. NeurIPS 2018, 12 pages.
4. Leo Osvald, Grégory Essertel, **Xilun Wu**, Lilliam I. Gonzalez Alayon, Tiark Rompf. Gentrification gone too far? affordable 2nd-class values for fun and (co-)effect. OOPSLA 2016, 18 page. **(Distinguished Artifact Award)**

## TEACHING

Xilun has rich experience in instructing undergrad and graduate level courses in Computer Science. The tasks performed as Teaching Assistant include organizing course materials, designing coding assignments, providing appropriate assistance and evaluating students' performance. The courses he has instructed as Teaching Assistant include:

- Parallel Systems (Graduate)
- Operating Systems (Undergrad)

- Systems Programming (Undergrad)
- Compilers: Principles And Practice (Undergrad)

## COURSES TAKEN

**Theory Courses:**

- Algorithm Design Analytics and Implementation CS58000
- Linear Algebra MA51100
- Numerical Linear Algebra CS51500

**Systems Courses:**

- Compiler & Programming Systems CS50200
- Operating Systems CS50300
- Parallel Computing CS52500
- Database Systems CS54100
- Programming Languages CS56500

**Statistical CS Courses:**

- Data Mining CS57300

**Research Seminars:**

- Graph Data Management CS59000
- Fault Tolerant Computer Systems Design CS59000
- Language-Based Systems Security CS59000
- Computer-aided Program Reasoning CS59000
- Deep Learning and System Reasoning CS59000

**Communication:**

- Effective Teaching In CS CS59100
- Classroom Communication English as a Second Language for TAs ENGL62000
- Written Communication English as a Second Language Graduates ENGL62100