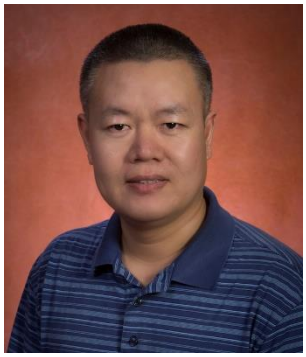
Dr. Weikuan Yu



Yu has broad research interests spanning structured and unstructured database systems, data mining and analytics of big data and social networks, parallel computer systems and architecture, high speed interconnects, cloud computing, computational biology and climate modeling, storage and I/O systems. Dr. Yu has graduated many doctoral and master students who upon graduation joined prestigious organizations such as Oak Ridge National Laboratory, Yahoo, Intel, Boeing, IBM T.J. Watson Research Center, Amazon, and other industry positions. He is currently working with the PASL group of two postdoctoral researchers, 8 Ph.D. students and 3 master students. His group's research has been supported in part by NASA, DOE (ORNL and LLNL), NSF, Alabama Department of Commerce, Mellanox, Intel, NVIDIA, and Scitor, along with equipment donations from Mellanox, NVIDIA, and Solarflare. Yu's research won the 2012 Alabama Innovation Award and the First Prize of the 2012 ACM Student Research Competition Grand Finals. Yu is a senior member of IEEE and member of ACM and USENIX.

Project Opportunities:

1. Explore different indexing techniques and log-structured storage platforms and evaluate the efficacy of data analytics programs to process complex datasets on large-scale computer systems.

2. Collect a variety of I/O traces from several common benchmarks and applications and evaluate the performance of different storage tools.

3. Evaluate the communication and I/O management techniques in deep neural networks to identify contentious bottlenecks and temporary hotspots.

Requirements:

1. Interest in computer systems research on data analytics and protection.
2. Junior or lower with good coding skills, and willingness to tackle new problems.
3. GPA > 3.4; can work up to a maximum of 10 hours per week during academic semesters, and 20 hours per week during summer.
4. Weekly reports to justify the hours claimed per week.

Contact: yuw@cs.fsu.edu

Dr. Grigory Fedyukovich

Dr. Fedyukovich's research interests are:

- Synthesis of inductive invariants for verification of program safety and termination
- Relational verification and its applications to security analysis and automated parallelization
- Functional (Skolem) synthesis via lazy quantifier elimination and programming by example
- Incremental verification using function summaries and simulation relations.

Project Opportunities:

Project 1

Wouldn't it be great if programs could be written automatically? For instance, if you could somehow describe what program should produce, then the actual code of the program would be magically generated? At FSU we develop a tool that does a little bit of magic and can synthesize small fragments of code (namely, bodies of functions without loops) completely automatically. Our tool is called AE-VAL and its source code is publicly available at https://github.com/grigoryfedyukovich/aeval/tree/master. We have several projects to make the tool more robust. All of them require some software engineering skills and experience with C/C++.

1) get familiar with the baseline algorithm for program synthesis (VMCAI'19 paper by Grigory Fedyukovich et al) and the tool

2) implement proper support for divisibility constraints

3) extend the technique to arrays

4) extend the technique to the compositional case (i.e., several functions at a time)

Project 2

Finding and repairing bugs in programs is notoriously hard and time-consuming. For this reason, research on automated program repair has grown intensively in recent years. One way to do a repair is to iteratively search for small mutations. For instance, if we replace '+' with '-', or '>' with '>=' in a buggy program we may find a correct program. Unfortunately, a naive enumeration of mutants does not guarantee success. We thus wish to discover efficient ways of performing such search and implement it on top of existing LLVM infrastructure (https://llvm.org/). The project requires good programming skills and experience with UNIX and C/C++.

1) get familiar with the baseline tools for checking program correctness and bug finding

2) get familiar with the LLVM infrastructure

3) implement various mutation strategies as LLVM passes

4) experiment with different buggy programs and mutation strategies

Project 3

To understand better what a computer program is doing, we often want to represent it as a mathematical formula. Interestingly, we can then use our mathematical knowledge to simplify/solve/prove such formula and even decode the result back to the program level. To help in this process, there are a lot of open-source tools available, including one developed at FSU. This project aims at improving the mathematical reasoning at the backend of the tool. The project requires good programming skills and experience with UNIX and C/C++.

1) get familiar with algorithms for proving program correctness (so-called CHC solvers)

2) get familiar with the FreqHorn tool (developed at FSU) to solve CHC

3) design and implement new algorithms for CHC-solving

4) experiment with different programs using old and new algorithms and find cases on which new algorithms outperform old algorithms

Website: http://www.cs.fsu.edu/~grigory

Contact: grigory@cs.fsu.edu

Dr. Chris Mills

Chris' broad research interests include software engineering and machine learning. Specifically, his research focuses on applications of artificial intelligence to software engineering automation, pre-compilation code optimization, natural language processing (NLP), manifold learning, and the construction of explainable modeling frameworks targeted at providing the power of machine learning to technical laypersons in industrial settings.

Project Opportunities:

Project 1

One large challenge facing digital assistants and chatbots is the unstructured nature of human conversation. Given a sufficiently complex conversational context, even a cutting-edge bot can experience failures that result in damaging user confidence. This project focuses on two aspects of this problem:

1. Identifying when conversational context has become sufficiently complex to begin questioning model fidelity.
2. Deploying conversational tactics to steer the conversation into a more predictable path by either asking for clarification or deflecting nonsensical requests.

Project 2

A core distinction between chatbots and digital assistants is that rather than providing only text-based responses, assistants are also typically capable of taking actions on behalf of a user. For example, not only can Siri respond to a request, she can also play a requested song or provide a map to a specified destination. Unfortunately, it is difficult to convey to a user what functionality a digital assistant has in practice. As a result, users can attempt to use a feature that is not yet implemented or invoke a feature in an unexpected way, which leads to frustration and declining adoption. This project focuses on using user feedback in chat logs to address these issues by:

1. Using unsupervised learning and text summarization to automatically identify high-demand, unsupported user intents.
2. Using online learning and model checkpointing to dynamically adjust to user-specific invocations that deviate from those used to train the base model.

Project 3

Firms in various industries such as finance and law have decades of data that is invaluable for making operational decisions. While large firms have budgets that allow for dedicated data science teams and staff statisticians, many small and medium size firms do not have such a luxury. Instead, they depend on accountants and other professionals that deeply understand the data in question but lack training in the statistical foundations of data analytics. Further, even software packages specifically marketed to these users fail to recognize this educational gap. This project focuses on constructing a natural language interface for technical laypersons to operationalize data by:

1. Providing direct access to relational data via a natural language interface (NLI). The NLI is housed within a digital assistant that answers arbitrary user questions by translating the user request into an intermediate form of SQL.
2. Providing an NLI for arbitrary model construction that uses user utterances for data extraction and an asynchronous, multi-objective genetic algorithm for hyperparameter selection to build a user-specific, predictive model with post-process visualizations to aid with decision making.

Contact: cmills@cs.fsu.edu