As programs and programmers continue growing in number, complexity, and diversity, we as computer scientists might hope that traditional techniques are "good enough" for the deluge of new problems and new applications. But the opposite is true. In areas like carpentry, 3D design, floating point arithmetic, or deep learning, users need programming language (PL) tools like compilers, optimizers, and verifiers to understand and manipulate their programs. Unfortuately, **building a state-of-the-art compiler is a gargantuan task**, well outside the reach of all but PL experts. If we want a broader audience to build their own optimizers and verifiers, **we need a new framework in which to build PL tools**.

The difficulty in building traditional program optimizers and verifiers is rooted in their reliance on a technique called term rewriting, which analyzes and manipulates **one version of program at a time**. This approach is well-studied and can be made very efficient, but it requires a litany of heuristics and assumptions to ensure that the compiler considers the right version of the program at the right time. What if we dropped this constraint, and allowed our PL tools to analyze and manipulate **many versions of a program simultaneously**? Doing so would eliminate many of the pitfalls that prevent domain experts from building their own compilers, optimizers, and verifiers.

My research is focused on **equalilty saturation** (EqSat), a technique that can analyze and manipulate many equivalent programs simultaneously, therefore freeing the developer from the oracular decision making required to build a traditional compiler. I see EqSat as the future of PL tools; both in "core" areas like general purpose programing and theorem provers, and for new domains that are reaching for PL tools. To achieve this end, I have adopted a broad research philosophy that advances EqSat and its adoption along three axes:

1. **Develop novel, core techniques** with solid foundations and broad applicability;
2. **Build systems and communities** to make these techniques available to users;
3. **Collaborate with domain experts** to apply these techniques to real-world problems.

During my PhD and postdoc, I have worked with and led teams of researchers to develop EqSat according to these principles. In short, I have **developed novel techniques and algorithms** that advance EqSat and connect the fields of compilers, relational databases, and formal methods; **built tools and community infrastructure** including software with industrial and academic users, and a new workshop at a top PL conference; and **collaborated with domain experts** to bring EqSat to bear on problems in graphics, fabrication, machine learning, and more.

1 Developing novel, core techniques

Most of my recent technical work is centered around **equality saturation** (EqSat), a technique that mitigates many of the challenges with the term rewriting approach used in most compilers, optimizers, and verifiers. In the classic term rewriting paradigm, users provide a set of rewrite rules (e.g. $x + x \to 2x$), and a system repeatedly applies them to transform an input program into an equivalent output program. The key question then becomes: **when to apply which rewrite?** Since term rewriting is destructive, an incorrect choice of rewrite can lead the system down a wrong path, missing optimization or verification opportunities. EqSat avoids this issue by simultaneously applying all rewrites, compactly representing **exponentially many equivalent versions of the program** in a data structure called an e-graph.

EqSat was invented by Tate et. al. in 2009 and promised to solve these issues with classical term rewriting by repurposing the e-graph data structure used in SMT solvers. But **issues with scale and flexibility** prevented the idea from taking off. My research agenda has focused on developing the technical advancements necessary to make EqSat a viable technique for PL researchers and practitioners. In particular, EqSat algorithms and implementations at the time did unnecessary work maintaining data structure invariants, and they lacked the flexibility to handle semantic program anaylses, forcing users to resort to ad-hoc solutions for even simple anaylses like constant folding. Our work in POPL 2021 [1] introduced the egg tool and was recognized as a **Distinguished Paper**. egg addresses the above issues with a new, amortized approach to invariant maintenance and a novel technique to lift of lattice-based program analyses to the e-graph level.

egg's introduction in 2021 revitalized research into e-graphs and EqSat. While egg's advances solved the invariant maintenance problem, **new applications exposed e-matching as another, more fundamental bottleneck**. E-matching is the process of searching the e-graph for terms that match a given pattern; for example, e-matching for the pattern $x + f(x)$ might yield terms like $2 + f(2)$, $g(10) + f(g(10))$, and so on. E-matching is the core operation that powers term rewriting in all EqSat tools, but existing e-matching algorithms were complex, slow, and lacked theoretical guarantees. This deficiency came despite the fact that e-matching has been well-studied for decades in the context of SMT solvers. I led a team of researchers to overcome these issues by **connecting e-matching to relational joins**, i.e., connecting a fundamental problem in PL and formal methods to one in databases. Our POPL 2022 work [2] does just that, making the observation that e-graphs can be viewed as relational databases, and e-matching can be viewed as a relational join. This connection seems obvious in hindsight, but this work resulted in **simple, asymptotically faster, and provably worst-case optimal e-matching algorithms**.

Connecting e-matching to relational joins was a major step forward for e-matching itself, but it also opened the door for my interest database theory and the relational model. Much of my postdoctoral work has continued to explore the **connection between EqSat and databases**. In particular, if e-graphs are databases and e-matching is a join, can we view EqSat as Datalog? Datalog is a declarative logic programming language; it is the subset of Prolog that roughly corresponds to recursive SQL. Connecting EqSat to Datalog lets us tap into decades of work on the latter, including query optimization, incremental computation, and theoretical understanding. I am currently leading a team of PhD students to take this idea even further, **completely rethinking e-graphs and EqSat from a relational perspective** in a way that will benefit both domains. Our prototype implementation (under submission to PLDI 2023 [3]) beats both EqSat tools and Datalog implementations, and it **unifies several disparate techniques across modern Datalog systems, SMT solvers, and EqSat tools**. This work will have a major impact on EqSat tools and will also likely change the way authors of SMT solvers approach their data structures; we are already in contact with the authors of CVC5 to integrate some of our previous work into their SMT solver [4]. There is much work left to do, however. EqSat still lacks the solid theoretical foundation that its cousin techniques, Datalog and term rewriting, both have; and SMT solvers' requirement for backtracking makes it difficult for these insights to apply to these widely used tools. I am excited to continue this work with students across PL, databases, and formal methods, and **I am Co-PI on a submitted NSF Medium grant on this topic** with Prof. Zachary Tatlock and Prof. Dan Suciu.

2 Building systems and communities

Building systems and communities to support the research of others is essential to reliable, continued scientific progress. This goes far beyond making the code open-source, writing some documentation, and getting the "artifact badges". In the case of egg, it means a **years-long effort** to create packaged software with over 100,000 downloads, documentation, full-length tutorials, and examples; to maintain a discussion and chat forum for users and like-minded researchers; and organize a workshop at PLDI 2022—a top PL conference. **I am Co-PI on a submitted NSF CCRI grant** to continue growing and supporting the community around egg and EqSat; this grant is also with Prof. Tatlock and Prof. Suciu. While the NSF Medium grant mentioned above proposes novel research connecting EqSat and databases, the CCRI (Community Research Infrastructure) grant focuses on building a sustainable community around egg and EqSat, including building a Community Advisory Board, creating datasets, and growing the open-source community.

Upon publishing our first paper egg in POPL 2021, I saw that egg could meet a real need in the PL community for a usable, flexible EqSat tool. However, I know first-hand the challenges of building on top of research artifacts; there is a gulf between the "reusable" badge on a paper and a system that others can understand, build upon, and contribute to. For the past three years, I have been working to bridge this gap by building **technical and community infrastructure** around egg. This includes the obvious technical aspects like the software itself, documentation, and tutorials, as well as follow-on technical work like our OOPLSA 2021 paper [5] (also a **Distinguished Paper**) to aid EqSat users in coming up with rewrite rules. But perhaps most important is the time spent teaching, training, and collaborating with students, users, and other researchers in the egg and EqSat community. On the student side, this includes **training students** to build and maintain quality technical infrastructure and documentation; several PhD students I have trained are now co-maintainers of egg. Externally, the result of this effort has been **wide-spread technical transfer to both industry and academia**. To give a few industrial examples: Intel uses egg to optimize arithmetic circuits, Fastly's new WebAssembly JIT compiler is directly inspired by egg, Certora relies on egg for checking logical equivalence of smart contracts, and Amazon Web Services uses egg together with SMT solvers as part of their automated reasoning pipeline. In academia, this level of support has led to dozens of projects not just citing our work, but making egg an essential component in their research.

Perhaps the most visible component of EqSat community infrastructure is the **EGRAPHS workshop at PLDI**, at top PL conference. I co-organized and chaired the program committee for the first edition in 2022, and I will do so again for the upcoming edition in 2023. We developed the workshop in response to the growing interest in EqSat and egg, including alternative implementations, new applications, and theoretical work. EGRAPHS is one-day workshop with a focus on work-in-progress, and is not limited to egg users, but targeted to the broader e-graph and EqSat community. This breadth led to a **diverse set of presenters including PhD students, undergrads, industry researchers, and hobbyists**. The first edition was an enormous success with 80 registered attendees, some of whom went on to publish the work presented at EGRAPHS. I look forward to future editions especially as a faculty member, as they should be great venue to mentor and recruit students as well as foster new collaborations.

3 Collaborating with domain experts

I have extensive experience collaborating with domain experts to apply PL, systems, and formal methods techniques to problems in areas like deep learning [6], 3D design [7], carpentry [8] and biological automation [9, 10]. In many cases, these collaborations (in addition to being great fun and learning opportunities) have **led to advances in the state-of-the-art in both the application domain and in the PL techniques themselves**.

As an example, I worked with experts in 3D computer-aided design to build a program synthesis tool to help users make large, structural edits to their models—like changing the number of spokes in a wheel [7]. The core of this technique is "re-rolling" flat, unstructured 3D models into loops and other programming constructs that capture the repetitive structure in the model; there are many ways to do so, so EqSat was a natural fit. However, EqSat at the time did not have a way to perform semantic analysis (required for loop re-rolling) on e-graphs. Solving this problem not only led to a PLDI 2021 paper [7] demonstrating program systhesis for reconstructing 3D models, but it also motivated the e-class analysis technique published the egg POPL 2021 paper [1].

My collaboration with deep learning experts at Google had a similar outcome. Recent work had used various graph rewriting techniques to optimize the compute graphs that represent deep learning programs. However, these graphs rewriting techniques suffered from the same problem as conventional term rewriting: the choice of when to apply rewrite rules is ad-hoc and can lead to suboptimal results. We developed an EqSat based optimizer for these compute graphs that can do cost-based optimization without the expensive backtracking that previous techniques required. Our superoptimizer yielded **better compile- and run-time performance than previous state-of-the-art**; this work was published at MLSys 2021 [6]. Critical to this project's success was the support for so-called "multi-patterns" that allow for searching for multiple patterns that share variables simultaneously. At the time, `egg` did not support them at all, and even SMT solvers support was (and still is) relatively ad-hoc. This MLSys 2021 work, required adding rudimentary support for multi-patterns in `egg`, and it motivated solving the problem once and for all in our POPL 2022 work [2] that reduced multi-patterns to relational queries.

I have also collaborated with biologists, chemists, and electrical engineers to bring PL and systems ideas to the area of biological automation. This included building new digital microfluidic hardware [11], designing an operating system-like abstraction for programming these kinds of devices [9, 12], and designing and analyzing new applications for these hybrid electronic-molecular systems [13, 14]. While most of this work was done earlier in my PhD, I continue to look for opportunities to collaborate with this area. For example, I have recently mentored undergraduate students on follow-on work [10], and I have worked with students using `egg` and EqSat in course projects in molecular programming.

4 Future Directions

E-graphs, EqSat, and their connection to relational databases have much more to give as research areas. From a theoretical perspective, EqSat is relatively poorly understood. We do not sufficiently understand either its termination criteria or even the guarantees that can be made when it *does* terminate. From a practical perspective, there are many open questions about how to use EqSat in a variety of settings. Languages with variable binding (like most general-purpose programming) are difficult to work with in EqSat, since the notion of equivalence becomes contextual. If we can extend e-graphs with a notion of context, that opens up a whole new set of applications. For example, using EqSat to do inlining in a general-purpose compiler could prove very fruitful, as the e-graph can represent both the inlined and non-inlined versions of a function. Another interesting direction is to use EqSat inside theorem provers or dependently typed languages. One challenge in these languages is that type equality (an important part of type checking) can now involve arbitrary computation; this makes typical approaches like canonicalization less feasible, but e-graphs are well-suited to these kinds of equivalence queries.

As we push e-graphs and EqSat closer to the relational model, I am also interested in working with colleagues in databases. In fact, I have already begun doing so; we have a paper under submission to SIGMOD [15] that presents a new join algorithm that combines best of both tried-and-true binary joins and newer, worst-case-optimal join algorithms. Similar to my work with EqSat, I am also interesting in connecting relational techniques to unconventional domains. For example, video game engines have lately begun moving to a data-oriented architecture called Entity-Component-System (ECS) that resembles a very limited relational database. Can we use the full power of relational databases and still meet the performance goals of modern interactive systems? Could a relational interface be a good fit for teaching programming via game development?

In summary—whether in PL with EqSat, relational databases, or in entirely new areas—I will continue to **develop novel, core techniques** like relational e-matching or e-class analysis, **build systems and communities** like `egg` or the EGRAPHS workshop, and **collaborate with domain experts** to apply and learn from diverse application areas.

[1] **Max Willsey**, Chandrakana Nandi, Yisu Remy Wang, Oliver Flatt, Zachary Tatlock, and Pavel Panchekha. egg: Fast and extensible equality saturation. In **POPL**, 2021. URL https://doi.org/10.1145/3434304. Distinguished Paper.

[2] Yihong Zhang, Yisu Remy Wang, **Max Willsey**, and Zachary Tatlock. Relational e-matching. In **POPL**, 2022. URL https://doi.org/10.1145/3498696.

[3] Yihong Zhang, Yisu Remy Wang, Oliver Flatt, David Cao, Philip Zucker, Eli Rosenthal, **Max Willsey**, and Zachary Tatlock. Better together: Unifying datalog and equality saturation. under submission to PLDI 2023.

[4] Oliver Flatt, Samuel Coward, **Max Willsey**, Zachary Tatlock, and Pavel Panchekha. Small proofs from congruence closure. In **FMCAD**, 2022. URL https://doi.org/10.34727/2021/isbn.978-3-85448-053-2_13.

[5] Chandrakana Nandi, **Max Willsey**, Amy Zhu, Yisu Remy Wang, Brett Saiki, Adam Anderson, Adriana Schulz, Dan Grossman, and Zachary Tatlock. Rewrite rule inference using equality saturation. In **OOPSLA**, 2021. URL https://doi.org/10.1145/3485496. Distinguished Paper.

[6] Yichen Yang, Phitchaya Mangpo Phothilimtha, Yisu Remy Wang, **Max Willsey**, Sudip Roy, and Jacques Pienaar. Equality saturation for tensor graph superoptimization. In **MLSys**, 2021. URL https://arxiv.org/abs/2101.01332.

[7] Chandrakana Nandi, **Max Willsey**, Adam Anderson, James R. Wilcox, Eva Darulova, Dan Grossman, and Zachary Tatlock. Synthesizing structured CAD models with equality saturation and inverse transformations. In **PLDI**, 2020. URL https://doi.org/10.1145/3385412.3386012.

[8] Haisen Zhao, **Max Willsey**, Amy Zhu, Chandrakana Nandi, Zachary Tatlock, Justin Solomon, and Adriana Schulz. Co-optimization of design and fabrication plans for carpentry. **ACM Transactions on Graphics (TOG)**, 2022. URL https://dl.acm.org/doi/10.1145/3508499.

[9] **Max Willsey**, Ashley P. Stephenson, Chris Takahashi, Pranav Vaid, Bichlien H. Nguyen, Michal Piszczek, Christine Betts, Sharon Newman, Sarang Joshi, Karin Strauss, and Luis Ceze. Puddle: A dynamic, error-correcting, full-stack microfluidics platform. In **ASPLOS**, 2019. URL https://doi.org/10.1145/3297858.3304027.

[10] Caleb Winston, **Max Willsey**, and Luis Ceze. Virtualizing existing fluidic programs. In **Journal on Emerging Technologies in Computing Systems**, 2022. URL https://doi.org/10.1145/3558550.

[11] Ashley Stephenson, **Max Willsey**, Jeff McBride, Sharon Newman, Bichlien Nguyen, Christopher Takahashi, Karin Strauss, and Luis Ceze. PurpleDrop: A digital microfluidics-based platform for hybrid molecular-electronics applications. **IEEE Micro**, 2020. URL https://ieeexplore.ieee.org/document/9130165.

[12] **Max Willsey**, Ashley P. Stephenson, Chris Takahashi, Bichlien H. Nguyen, Sarang Joshi, Karin Strauss, and Luis Ceze. Scaling microfluidics to complex, dynamic protocols. In **ICCAD**, 2019. URL https://ieeexplore.ieee.org/document/8942070. Invited Paper.

[13] Douglas Carmean, Luis Ceze, Georg Seelig, Callie Bee, Karin Strauss, and **Max Willsey**. Dna data storage and hybrid molecular-electronic computing. **Proceedings of the IEEE**, 2019. URL https://ieeexplore.ieee.org/abstract/document/8556046. Invited Paper.

[14] Sharon Newman, Ashley P Stephenson, **Max Willsey**, Bichlien H Nguyen, Christopher N Takahashi, Karin Strauss, and Luis Ceze. High density dna data storage library via dehydration with digital microfluidic retrieval. **Nature Communications**, 2019. URL https://www.nature.com/articles/s41467-019-09517-y.

[15] Yisu Remy Wang, **Max Willsey**, and Dan Suciu. Free join: Unifying worst-cast optimal and traditional joins. under submission to SIGMOD 2023.