Equality Saturation for MLIR with Egglog

—

Executive Summary

- Build a tool that applies equality saturation to MLIR using egglog.
- Compiler backend will use xDSL.
- Evaluation will compare program runtime against other optimisations.

Phase-Ordering Problem (1)

If we have a term rewrite: $x \times 2 \rightarrow x \ll 1$

Then $(a \times 2)/2 \rightarrow (a \ll 1)/2$

But should be $(a \times 2)/2 \rightarrow a$

Phase-Ordering Problem (2)

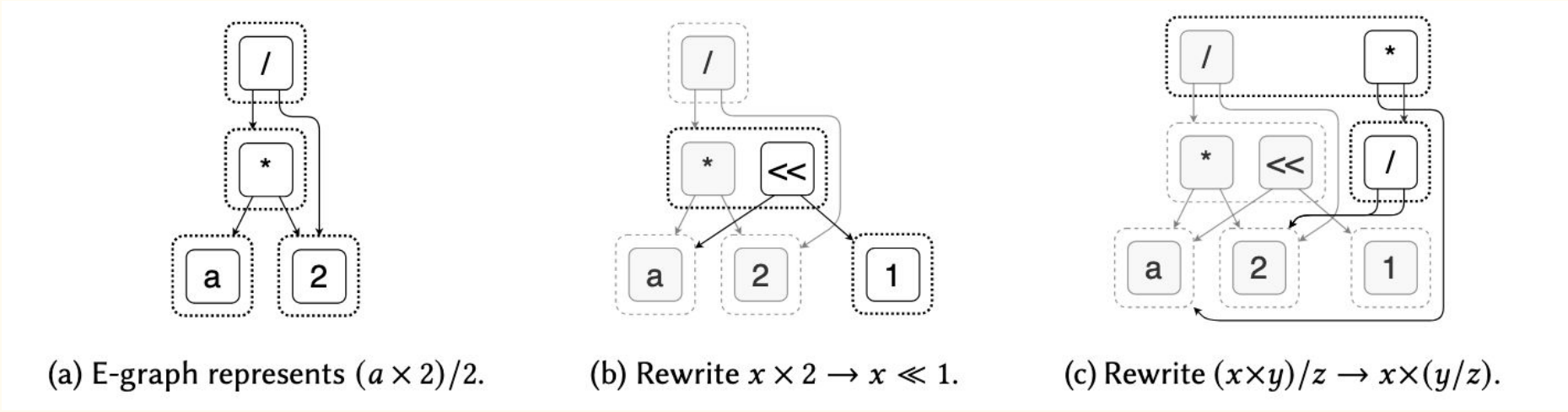
- Term rewriting is destructive.

- Which rewrite? When?

Equality Saturation

- Apply all the rewrites all the time!
- Addresses phase ordering problem using an e-graph.
- Explore then extract.

E-Graph



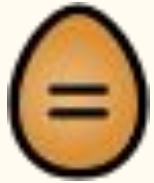
(a) E-graph represents $(a \times 2)/2$.

(b) Rewrite $x \times 2 \rightarrow x \ll 1$.

(c) Rewrite $(x \times y)/z \rightarrow x \times (y/z)$.

Figure from Willsey et al. “egg: Fast and Extensible Equality Saturation”, 2021.

Egg(log)



- Egg: Rust library for equality saturation.

- Datalog: Recursive database query language.
  - Supports incremental execution.

- Egglog = Egg + Datalog
  - Python library binds to Rust implementation.

MLIR: Multi-level Intermediate Representation

- Modular compiler infrastructure for domain-specific compilation.

- Allows abstraction levels to be modelled as *dialects*.
  - tensor, linalg, affine, scf, etc

- Provides reusable dialect transformations.
  - Canonicalization: Constant folding, dead code elimination, etc.

MLIR - Toy Dialect

```
# User defined generic function that operates on unknown shaped arguments
def multiply_transpose(a, b) {
  return transpose(a) * transpose(b);
}
```

```
func @multiply_transpose(%arg0: tensor<*xf64>, %arg1: tensor<*xf64>)
    -> tensor<*xf64> {
  %0 = "toy.transpose"(%arg0) : (tensor<*xf64>) -> tensor<*xf64>
  %1 = "toy.transpose"(%arg1) : (tensor<*xf64>) -> tensor<*xf64>
  %2 = "toy.mul"(%0, %1) : (tensor<*xf64>, tensor<*xf64>) -> tensor<*xf64>
  "toy.return"(%2) : (tensor<*xf64>) -> ()
}
```

Figure from “Building a Compiler with MLIR” presentation at LLVM Dev Mtg, 2020 by Mehdi Amini and River Riddle

xDSL

- Python sidekick for MLIR.
- Compiler infrastructure for Python DSLs.
- Will provide backend for toy compiler.

Project Goals

- Explore a dialect-agnostic way to apply equality saturation to MLIR using egglog.

- Compare execution time of EqSat-optimised program with existing MLIR transformations on small programs.

Project Plan

- Preliminary research.
  - Equality Saturation, Egglog, MLIR, xDSL, etc.
- Implementation and Evaluation.
- Write up.

Questions?