xDSL: Prototyping MLIR in Python

Sasha Lopoukhine, Mathieu Fehr, and many more!
Research

Open-Source Development

Teaching
(150 students / year)
SSA and Operations
Blocks and Regions
Dialects as abstractions
Peephole rewrites
TableGen
CMake
C++
Templates
Compilation Time
Core Concepts...
How to teach MLIR?

MLIR is hard to get started with!
What if we reimplemented MLIR in Python?
Operation definition in xDSL

```python
@irdl_op_definition
class AddOp(IRDLOperation):
    name = "toy.add"
    lhs: Annotated[Operand, AnyTensorTypeF64]
    rhs: Annotated[Operand, AnyTensorTypeF64]
    res: Annotated[OpResult, AnyTensorTypeF64]
```
Module builders in xDSL

```python
@Builder.implicit_region
def main() -> None:
    x = toy.ConstantOp.from_list([1, 2, 3, 4, 5, 6], [2, 3]).res
    y = toy.AddOp(x, x).res
    toy.PrintOp(y)
    toy.ReturnOp()

toy.FuncOp("main", main_type, main)
```
Reshape definition in xDSL

class ReshapeReshapeOpPattern(RewritePattern):
    @op_type_rewrite_pattern
    def match_and_rewrite(self, op: ReshapeOp, rewriter: PatternRewriter):
        """Reshape(Reshape(x)) = Reshape(x)"""
        if isinstance(op.arg, OpResult) and isinstance(op.arg.op, ReshapeOp):
            t = cast(TensorType[Float64Type], op.res.typ)
            new_op = ReshapeOp(op.arg.op.arg, t)
            rewriter.replace_matched_op(new_op)
Easy to get started

```
import xdsl
from xdsl.printer import Printer

@example
def main():
    n = 5
    x = 0
    while x < n:
        print(x)
        x = x + 1
```

pip install xdsl

Fast iteration loop

Fully hackable

xdsl.dev/xdsl
xDSL: A Python framework replicating MLIR
xDSL: A Python framework replicating MLIR
xDSL: A Python **Sidekick** for MLIR
Prototyping MLIR dialects in xDSL
Prototyping MLIR dialects in xDSL

xDSL pass reducing accessed columns

Improvement over database compiler DuckDB
How do we connect with MLIR?
How do we connect with MLIR?

Textual IR

%0 = arith.constant 42 : i32
%1 = arith.constant 12 : i32
%2 = arith.addi %0, %1 : i32
How do we connect with MLIR?

Textual IR

%0 = arith.constant 42 : i32
%1 = arith.constant 12 : i32
%2 = arith.addi %0, %1 : i32

What is arith?
How do we connect with MLIR? (WIP)

```mlir
irdl.operation @arith.addi {
  %0 = irdl.is_type : f32
  %1 = irdl.is_type : f64
  %2 = irdl.any_of(%0, %1)
  irdl.operands(%2, %2)
  irdl.results(%2)
}
```
Why not Python Bindings?
The Pyramid of MLIR Compatibility
The Pyramid of MLIR Compatibility
The Pyramid of MLIR Compatibility

- IR and textual format
- IRDL dialect

- Monthly (?)
- Yearly
The Pyramid of MLIR Compatibility

- **Dialect definitions**: Weekly
- **IRDL dialect**: Monthly (?)
- **IR and textual format**: Yearly
The Pyramid of MLIR Compatibility

- IR and textual format
  Generates
- IRDL dialect
  Weekly
- Dialect definitions
  Monthly (?)
  Yearly
The Pyramid of MLIR Compatibility

- **API** (Daily)
- **Dialect definitions** (Weekly)
- **IRDL dialect** (Monthly ?)
- **IR and textual format** (Yearly)

Generates
Other *Sidekick* Frameworks?
xDSL: A Compiler Infrastructure for Python DSLs

xdsl.dev

github.com/xdslproject/xdsl/
Backup slides
xDSDL has reasonable overheads compared to MLIR