

# Non-attribute property improvements in MLIR

Krzysztof Drewniak<sup>1</sup>

Advanced Micro Devices

Oct 28, 2025

---

<sup>1</sup>Krzysztof.Drewniak@amd.com

## History and context

# What's a non-attribute property?

We can attach constants and data to operations

Usual way: stored as Attribute

```
def MyOp : ... {  
    let arguments = (ins ...  
        I64Attr:$length,  
        UnitAttr:$isBar  
    );  
}
```

- ▶ Immutable
- ▶ Unique
- ▶ Live forever in context

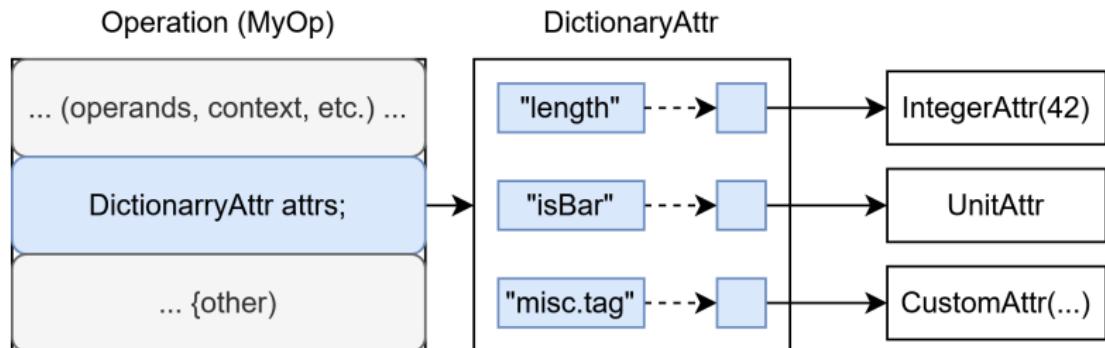
New: Non-attribute property

```
def MyOp : ... {  
    let arguments = (ins ...  
        I64Prop:$length,  
        UnitProp:$isBar  
    );  
}
```

- ▶ Mutable
- ▶ Live inline on operation

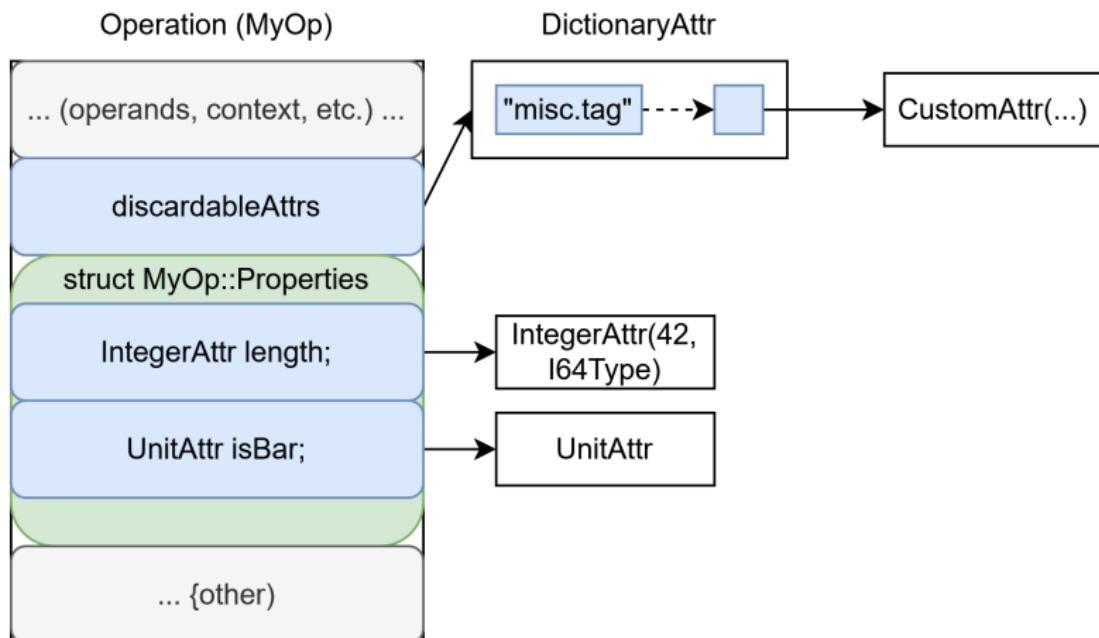
# The very old days

All attached data uniqued into uniques dictionary



# Properties structs

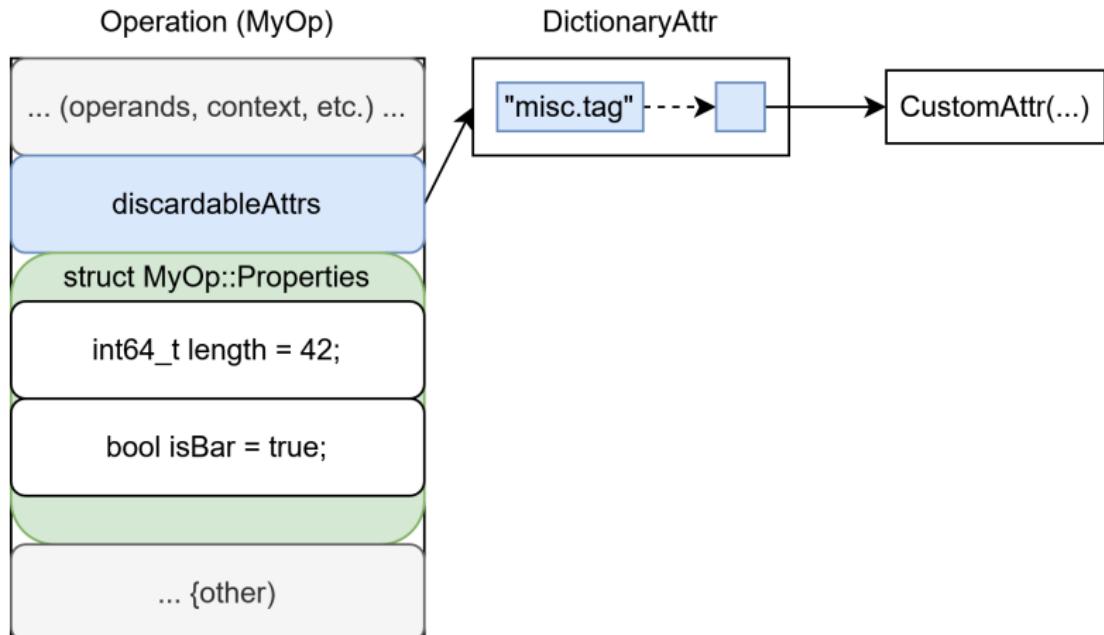
Data inherent to operation moves inline — still uniqued



What we do today

# Non-attribute properties

The struct **could** store anything



Rarely used feature — missing infrastructure

New infrastructure

## Parsing / printing

```
def MyOp : ... {  
    let arguments = (ins ...  
        I64Prop:$length,  
        UnitProp:$isBar  
    );  
}
```

```
let assemblyFormat =  
    "$length (`bar` $isBar)?";
```

Now you can have

```
my_op 5 bar
```

without custom parsers/printers

# Builders

```
def MyOp : ... {  
    let arguments = (ins ...  
        I64Prop:$length,  
        UnitProp:$isBar  
    );  
}
```

We now generate

```
MyOp::create(Location loc,  
             ...  
             int64_t length,  
             bool isBar = false);
```

automatically

## Other new features

- ▶ Combinators
  - ▶ OptionalProp
  - ▶ ArrayProp
  - ▶ ConfinedProp and PropConstraint
- ▶ Enum generation != attribute generation - I\*Enum and EnumProp
- ▶ Tablegen rewrite patterns handle properties now

## Future plans

- ▶ Generic builders taking properties structs directly
- ▶ PropRef and OwningPropRef
- ▶ Generic parsing/printing of properties — quick and easy stringly-typed FFI

Ongoing discussion and improvement — ideas welcome!