Google

StableHLO Compatibility

OpenXLA Community Meeting

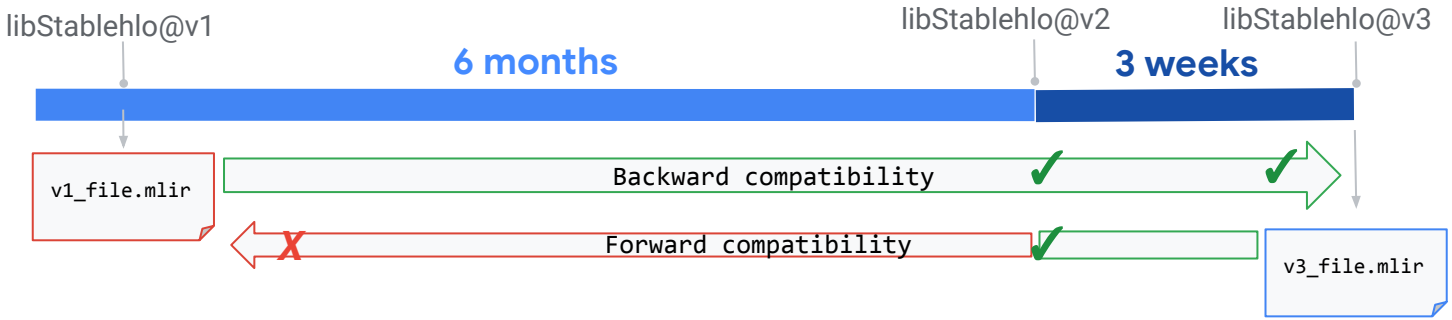
Kevin Gleason - 10/12/22

Agenda

- Compatibility Requirements
- Upgrade and Downgrade Hooks
- Compatibility Prototype
- Next Steps

Requirements *(feedback wanted!)*

R1	libStablehlo provides 6 months of backward compatibility
R2	libStablehlo provides 3 weeks of forward compatibility
R3	Source compatibility for C, C++ and Python APIs within libStablehlo is an aspirational goal.



Feedback: openxla/stablehlo#115

Supported Dialects and Operations, Attributes, Types

Dialect	Supported Ops, Attributes and Types
Arith	`AddIOp, CmpIOp, CmpIPredicateAttr, ConstantOp, DivSIOp, ExtSIOp, IndexCastOp, MaxSIOp, MinSIOp, MulIOp, SelectOp, SubIOp, TruncIOp`
Builtin	No ops, but all attributes and types.
CHLO	All ops, attributes and types.
Func	`CallOp`
MLProgram	All ops, attributes and types.
SparseTensor	Aspirational (pending the sparsity RFC which is expected in Q4 2022).
StableHLO	All ops, attributes and types except `CustomCallOp` whose semantics is implementation-defined.
Tensor	`CastOp, DimOp, FromElementsOp`

Feedback: [openxla/stablehlo#115]

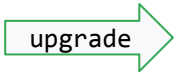
IR Upgrades

- Provides backward compatibility
- For 6 months, support dialect upgrades
- Payloads have version number, allowing tooling to error if out of compatibility window

Rename `stablehlo.sub` → `stablehlo.subtract`

v39

```
func.func @upgrade(%arg0: tensor<2xi1>) -> () {
  %0 = stablehlo.sub %arg0, %arg0 : tensor<2xi1>
  return
}
```



v40

```
func.func @upgrade(%arg0: tensor<2xi1>) -> () {
  %0 = stablehlo.subtract %arg0, %arg0 : tensor<2xi1>
  return
}
```

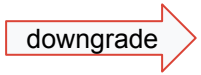
IR Downgrades

- Provides forwared compatibility when possible, else warn / error
- For 3 weeks after dialect changes, target the previous version
- Can be processed in v39, and can be upgraded by v40

Rename `stablehlo.sub` → `stablehlo.subtract`

v40

```
func.func @upgrade(%arg0: tensor<2xi1>) -> () {
  %0 = stablehlo.subtract %arg0, %arg0 : tensor<2xi1>
  return
}
```



v39

```
func.func @upgrade(%arg0: tensor<2xi1>) -> () {
  %0 = stablehlo.sub %arg0, %arg0 : tensor<2xi1>
  return
}
```

Compatibility Prototype Tool / API

Compatibility Tool

```
stablehlo-translate --compat [flags] file.mlir
    --target=<version>    Target version to generate (Default to minimum supported)
    --emit-asm            Output textual assembly (Default generate bytecode)
```

Compatibility APIs

```
// namespace mlir::stablehlo
OwningOpRef<Operation *> parseWithCompat(llvm::SourceMgr &sourceMgr,
                                         MLIRContext *context);

LogicalResult writeWithCompat(Operation *topLevelOperation,
                              int64_t targetVersion,
                              bool emitAssembly,
                              llvm::raw_ostream &output);
```

Feedback: openxla/stablehlo#278

Compatibility Prototype Example

```
void registerSubOpChanges() {
 // Change log:
 //   Version 39: SubOp<"stablehlo.sub"> exists
 //   Version 40: SubOp<"stablehlo.sub"> -> SubtractOp<"stablehlo.subtract">
 // Backward compatibility: Support v39 and after.
 // Forward compatibility: Target v39 for printing.

 // Upgrade <v40 -> 40: [sub --> subtract]
 addUpgrade("stablehlo.sub", 40,
             [&](Operation *op, int64_t fromVer) -> LogicalResult {
               renameOperation(op, "stablehlo.subtract");
               return success();
             });

 // Downgrade v40 -> 39: [subtract --> sub]
 addDowngrade("stablehlo.subtract", 39,
               [&](Operation *op, int64_t fromVer) -> LogicalResult {
                 renameOperation(op, "stablehlo.sub");
                 return success();
               });
}
```

Compatibility Prototype Example

```
gleasonk@gleasonk-linux:~/compat$ cat sub.mlir
func.func private @test_sub(%arg0: tensor<2xi1>) -> () attributes {stablehlo.compat_version = 37 : i32} {
  %0 = "stablehlo.sub"(%arg0, %arg0) : (tensor<2xi1>, tensor<2xi1>) -> tensor<2xi1>
  func.return
}
gleasonk@gleasonk-linux:~/compat$ stablehlo-translate --compat --target=40 sub.mlir
func.func private @test_sub(%arg0: tensor<2xi1>) attributes {stablehlo.compat_version = 40 : i64} {
  %0 = stablehlo.subtract %arg0, %arg0 : tensor<2xi1>
  return
}
gleasonk@gleasonk-linux:~/compat$ stablehlo-translate --compat --target=38 sub.mlir
"func.func"() ({
^bb0(%arg0: tensor<2xi1>):
  %0 = "stablehlo.sub"(%arg0, %arg0) : (tensor<2xi1>, tensor<2xi1>) -> tensor<2xi1>
  "func.return"() : () -> ()
}) {function_type = (tensor<2xi1>) -> (), stablehlo.compat_version = 38 : i64, sym_name = "test_sub", sym_
) -> ()
```

Next Steps

- In Q4 2022:
  - Propose an MLIR RFC about stability of the bytecode format.
  - Start a discussion about stability guarantees for the Builtin dialect.
  - Finish the compatibility tool, hold a design review for the StableHLO Compatibility RFC
- We aim to ship a working solution by the end of the year.