Dialects as a Dialect

Bringing native C++ registration to IRDL



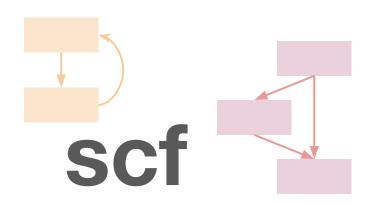
Ivan Ho

PhD Student Supervised by Tobias Grosser

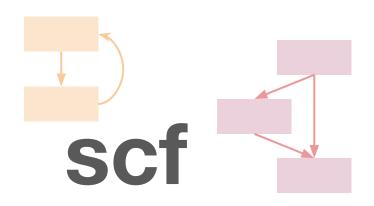






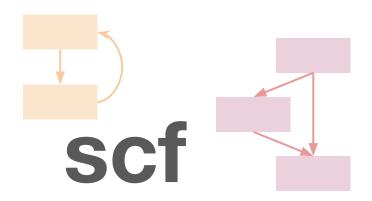










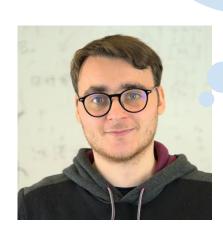




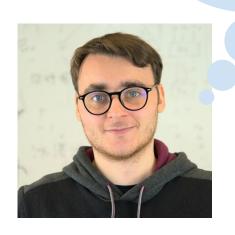
transform



What if we had dialects as a dialect?

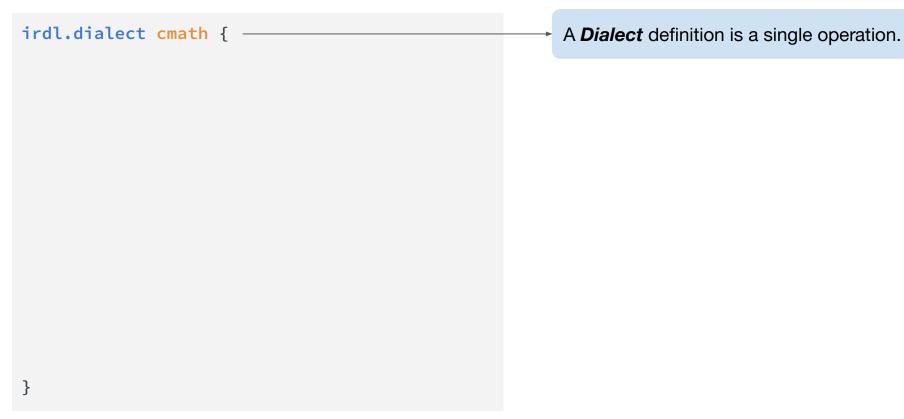


What if we had dialects as a dialect?



IRDL
\\((a)^(a)^///

```
irdl.dialect cmath {
  irdl.type complex {
    %0 = irdl.is f32
    %1 = irdl.is f64
    %2 = irdl.any_of(%0, %1)
    irdl.parameters (elem: %2)
  irdl.operation norm {
    %0 = irdl.any
    %1 = irdl.parametric @cmath::@complex(%0)
    irdl.operands (in: %1)
    irdl.results (res: %0)
```



```
irdl.dialect cmath {
                                                        A Dialect definition is a single operation.
  irdl.type complex {
    %0 = irdl.is f32
    %1 = irdl.is f64
    %2 = irdl.any_of(%0, %1)
                                                         Types may have constraints.
    irdl.parameters (elem: %2)
```

```
irdl.dialect cmath {
                                                        A Dialect definition is a single operation.
  irdl.type complex {
    %0 = irdl.is f32
    %1 = irdl.is f64
    %2 = irdl.any_of(%0, %1)
                                                        Types may have constraints.
    irdl.parameters (elem: %2)
  irdl.operation norm {
    %0 = irdl.any
    %1 = irdl.parametric @cmath::@complex(%0)
                                                        An Operation is defined similarly.
    irdl.operands (in: %1)
    irdl.results (res: %0)
```

```
irdl.dialect cmath {
  irdl.type complex {
   // ...
  irdl.operation norm {
    %0 = irdl.any
    %1 = irdl.parametric @cmath::@complex(%0)
    irdl.operands (in: %1)
    irdl.results (res: %0)
```

```
$ mlir-opt --irdl-file=cmath.irdl.mlir example.cmath.mlir
```

```
irdl.dialect cmath {
  irdl.type complex {
   // ...
  irdl.operation norm {
    %0 = irdl.any
    %1 = irdl.parametric @cmath::@complex(%0)
    irdl.operands (in: %1)
    irdl.results (res: %0)
```

IRDL dialects are loaded **dynamically**

```
$ mlir-opt --irdl-file=cmath.irdl.mlir example.cmath.mlir
```

```
irdl.dialect cmath {
  irdl.type complex {
    // ...
  irdl.operation norm {
    %0 = irdl.any
    %1 = irdl.parametric @cmath::@complex(%0)
    irdl.operands (in: %1)
    irdl.results (res: %0)
```

```
$ mlir-opt --irdl-file=cmath.irdl.mlir
example.cmath.mlir
```

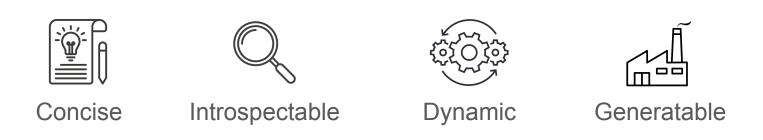
```
irdl.dialect cmath {
  irdl.type complex {
    // ...
  irdl.operation norm {
    %0 = irdl.any
    %1 = irdl.parametric @cmath::@complex(%0)
    irdl.operands (in: %1)
    irdl.results (res: %0)
```

```
$ mlir-opt --irdl-file=cmath.irdl.mlir example.cmath.mlir
```

```
irdl.dialect cmath {
  irdl.type complex {
    // ...
  irdl.operation norm {
    %0 = irdl.any
    %1 = irdl.parametric @cmath::@complex(%0)
    irdl.operands (in: %1)
    irdl.results (res: %0)
```

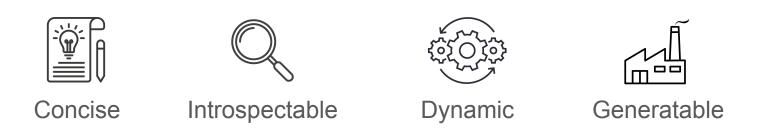
X unsatisfied constraint

IRDL: IR Definition Language



And it *just* works...

IRDL: IR Definition Language

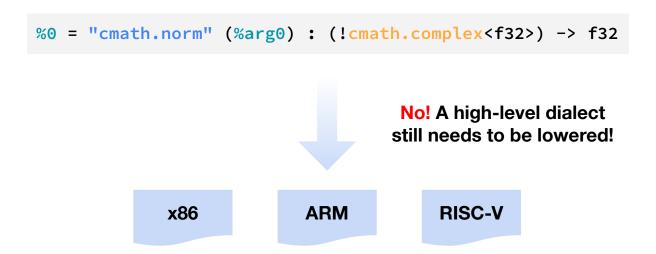


And it just works... kind of

Is parsing this program enough?

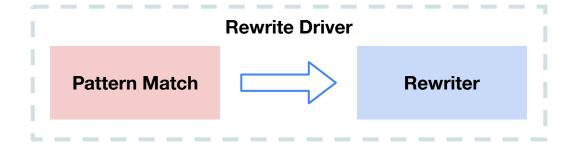
```
%0 = "cmath.norm" (%arg0) : (!cmath.complex<f32>) -> f32
```

Is parsing this program enough?



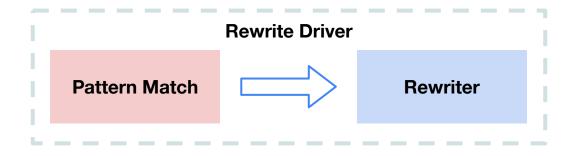
How to Rewrite IR in MLIR

(the abridged version)



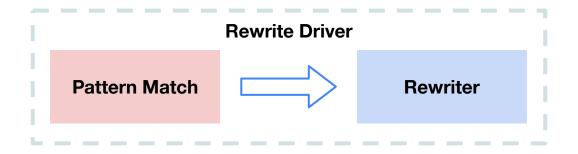
How to Rewrite IR in MLIR

(the abridged version)



How to Rewrite IR in MLIR

(the abridged version)



mlir-opt --irdl-file=cmath.irdl.mlir example.cmath.mlir

IRDL dialects are loaded dynamically

MLIR infrastructure uses **static C++ types**!

mlir-opt --irdl-file=cmath.irdl.mlir example.cmath.mlir

IRDL dialects are loaded dynamically

IRDL doesn't work with the existing MLIR C++ infrastructure

MLIR infrastructure uses **static C++ types**!

mlir-opt --irdl-file=cmath.irdl.mlir example.cmath.mlir

IRDL dialects are loaded dynamically

IRDL didn't work with the existing MLIR C++ infrastructure

MLIR infrastructure uses **static C++ types**!

\$ mlir-irdl-to-cpp example.irdl.mlir

static constemp:::llvm::StringLiteral getOperationName() return :: [lym::StringLiteral("test ind) to cap.bar") closs Smalladdanter : minite Sandinianarioldeniarus ettir: Valundannas static ::llvm::ArrayRef<::llvm::StringRef> getOperandNames() { return operandNames; DOLLEC:

voice Boscotte

voice static ::llwn::StringRef getOperandName(unsigned index) (class Beeffg : public ::stir::dp:Beeffgo (public: using designition using designition using designition total designition - seedigadaguser; totalist ctyporous Raughis using Desiricidation - Seedigadem-indicator-Dangells; using Desiricidation - Seedigadem-indicator-D return get@perandHames()[index]; static ::llwn::ArrayRef<::llwn::StringRef> getResultNames() (wbsit:
 wbsitrettetgetiatect!) over136g
 static caratrage :(Unno:StringLiteral.getifalactinnungace)) {
 return :(Unno:StringLiteral.getifalactinnungace)) {
 return :(Unno:StringLiteral.getifalactinnungace)) }
} static constempr ::llvs::StringLiteral getOperationName() {
 return ::llvs::StringLiteral("test_ind_to_opp-hosh"); static ::llwn::StringRef getResultName(unsigned index) { religitype paratype (religiblest DistactionFerent Aparter) test override /// Print a type registered to this diatect.
vaid printippelicalization type type,
realization value and count overtime std::pair<unsigned, unsigned- getStructuredOperandIndexAndLength(unsigned index) { static ::[lvm::StringBef gotOperandSume(ursigned index) {
 assert(index = 2 %5 "invalid attribute index");
 return getOperandSumes()[index];
} N.TH. DECLARE_EMPLICET_TYPE_DD:::orlin::text_inst_to_cop::TextInstToCoppliatect static ::llwn:/ArreyMefs::llwn:StringMefs getMesultNames() {
 static ::llwn:StringNef resaltNames() = ("res");
 orbarn rosultNames; ::mlir::Operation::operand_range_getStructuredOperands(unsigned_index) {
 auto-valueRange = getStructuredOperandIndexAndLength(index); return (std::mxt(getOperation()->operand_begin(), valueRange.first), std::mext(getOperation()->operand_begin(), valueRange.first + valueRange.second)); static ::llum:StringDef getRessltName(urs:good index) {
 assert(!ndex < 1.55 "insatid attribute index");
 return getRessltName()[index];
} cless TestEndIToCopType | public ||mlir:|Type |mblic: | UKING Type:|Type: std::pair-unsigned, unsigned- petStructuredtesultIndexAndLength(unsigned index) (/// Return the unatractared operand index of a atractared operand along with the answet of ometra sidelpakenus/syred, ansigned-petitractured/perandIndexIndEngth/unsigned index) { return (index, 1)} intlr:(Operation:result_range getStructuredResults(unsigned index) (
 auto valueMange = getStructuredResultIndexIndiangth index);
 return (std::mext[getOperation()->result_begin(), valueMange.first), /// Gat the 2-th structured operand [Single value, voriable or optional]
:##If::Operation:operand_mange_patStructuredOperands(unsigned incom) (HAR RECLAR EMPLICET TWO ID:::eldr::test drift to com::fretfreffedaphyse sub-valueForce = getfrictured@graedindowled.cogthicres()
return (std):centigetOperation(:-operand_begin(), valueForce, first),
return (std):centigetOperation(:-operand_begin(), valueForce, first - reducionos or med()); std::next(metOperation()->result begin(), valueRange.first + valueRange.second)); ::mlir::Value getRes() { return ::llvm::cast<::mlir::Value>(getStructuredResults(0).front()); } atdispairsusspeet, unsigned- getStructuredMesultImdexAndLength(unsigned index) (orders (Nooks, 1); tions healyne - motic (anticityen/ignehosenhealyne, heitfollodefine, inticityedforeigne (olong home Home; olong home Home; olong home Home; olong anticityen olong disparat was a "mort, infl., is, igno for"; (antic consider ittless disparated ballethealyne) (anticityen olong home; orange ("teol") ("mort disparated by phenometry ("mort disparated by infl.) static void build(:mlir:Openiler-Gopbulder, ::mlir:OpenilerState dopState, ::mlir:Type res, ::llve::ArrayNefc::mlir::NewedStributes attributes = {});
static void build(::mlir::Openiler-GopSpulder, ::mlir::OpenilerState GopState, ::mlir::TypeRaspe resultTypes, ::mlir::ValueRaspe operands, ::llve::ArrayNefc::mlir:

**The Company of the /// Det the new structured result (single value, variable or optional).
Indir/injuration/result_rusage_gettrectrosfments(prostype: Index) {
 and value/maps = patterctrosfments(betterface) (refer)
 return (set/:rest(petforeid)) -=result_page(s), value/maps.frest),
 sin/:rest(petforeids) -=result_page(s), value/maps.frest = value/maps.scond(s);
 sin/:rest(petforeids) -=result_page(s), value/maps.frest = value/maps.scond(s);
 sin/:rest(petforeids). /// Set the c-to structured result (single value, veriadic or sections).

individuantion (result yangs gettiresturesbesits)(no.god (sobs))

and voluburen; postporturesbesitsticknoinsbesits)(risks)

coturn (std:(next)getiperation) -result.bgs(n), voluburen; first),

coturn (std:(next)getiperation) -result.bgs(n), voluburen; first) -palartinge.second)) \$ mlir-irdl-to-cpp example.irdl.mlir ::mlir::OperationState GogState, ::mlir::Type res, ::mlir::Value Un; ::mlir:: , ::mlir::OperationState SadSState, ::mlir::TypeRampe resultTypes, ::mlir::Value class BeefOpGemericAdagtorBase (struct Properties (namespace will (namespace test inst to opp-(1005 MartyCenericAdayter) mbltc: stract Properties (| BeefQpGenericAdaptorBase(::mlir::Operation =co) : odsAttrs(co--getRawDictionaryAttrs()), odsDpName(co--getRamOions(co--getRagions(co--getRagions()) {} /// Return the anstructured operand index of a structured operand along with the amount of unstructured operands it contains std::pair-unsigned, unsigned operands:izel { bool TestInditotppType::classof(Type type) {
 return liver:ime=TestInditotppTimlect>(Type.getDimlect(T); No. ICC: | const Properties SpetProperties() (static malitrosphinestheratemit quartefulphinese(malitrosphinese(paint, millionitrisphinesemit, millionitys) and under millionitrisphinese (paints) and under millionitrisphinese (paints) and under millionitrisphinese (malitrosphinesemit), (Millionitrisphin, Millionitrisphin, Millionitrisphin, Millionitrisphin, Millionitrisphin, Millionitrisphin, Millionitrisphin, Millionitrisphin, Millionitrisphinesemit, Millio ::std::notional:::mlir::DoerationName:.odsOcName: Properties properties; ::mlir::RegisWierge edsRegists; - Opforts (A) (Then of Striegled Arrests, There (Stige) if template <typename RangeT>
class BeefOpGenericAdaptor : public detail::BeefOpGenericAdaptorGase {
 using WalueT = ::llvm::detail::ValueOfRange-RangeT>; tomplate «typonamo Remyal» class HankSphemerichlepter | public detail:|RenkSphemerichlepterNese | using Nabau" = :llaw:|detail:|RahadPHamperAmpel*) uxing Nabau = dorall:!RahdphomerichleptorHamp; BeefőpSenericAdaptor(RangeT values, ::mlir::DictionaryAttr attrs, ::mlir::OpaqueFraperties properties, ::mlir::RegionRange regions = ()) : BeefőpSenericAdaptor(values Property organism Lindows access access and proposed in access and a property organism Lindows access and a proposed by the property of the proposed by the property of the pr Heshby Generic Liberton (Respet volumes, const. Reshby Generic Liberta Liberton Gauss) + General Liberton General Liberton Gaussian (Liberton Gaussian Liberton Gaussian Gau// The Comparing parameter action will place be to the control later.

Impolate Spreame Latelant = Mainly, (spreame = 5th control LTL tradition among action to the control later.

MainlyScenticAdoptor(Respet volume, Latelant of : Baselog), addignmands(volume) () | ReefOptomerizedaptor(RampeT values, LateInst op): Rame(op), odsOperands(values) {} ::Rir::Twee TestIndlToCopSistect::parseType(::glir::Dialect#ssFarser &parser) const (inthicrips TestInditophalactingsreDypolinitringhalactingAnner i Illus 1986 bypate "Borner_getCarmention(i); itMostSringAn record; itMostSringAn record; itMostSringAn record; itMostSringAn recording and recording and recording and into annexes(vi. - generateflypeParser(perser, Amendia, gentyse); items gentyles, equae()) std::pair<unsigned, unsigned> petStructuredOperandIndexAndLength(unsigned index) { return Base::getStructuredOperandIndexAndLength(index, odsOperands.size()); /// Set the s-to structured operand (single volse, variable or option Rangel getStructureSperands\uniqued index) { auto velucRange = petStructureSperandIndexAndLength(index); /// Get the s-th structured operand (single value, variadic or aptional).
RangeT getStructuredOperands(unsigned index) {
 auto valueRange = getStructuredOperandIndexAndLength(index); std::mext(odsOperands.begin(), valueRange.first + valueRange.second)); /// Frant a type requestered to this similar.

would TestInd(Totappinietti)printType (intlinitype type, RangeT petOperands() (



add_irdl_to_cpp_target(TestIRDLToCppGen test_irdl_to_cpp.irdl.mlir)

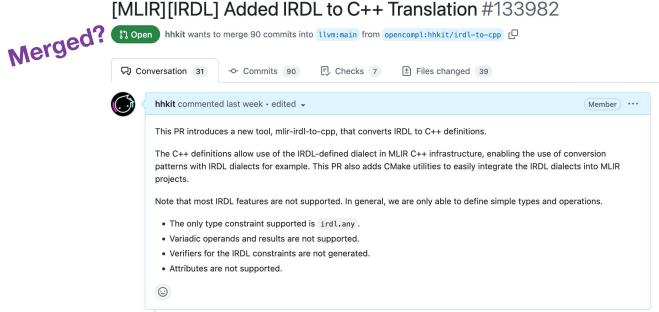
```
// CHECK: func.func @test() {
// CHECK: %[[v0:[^]*]] = "test_irdl_to_cpp.bar"() : () -> i32
// CHECK: %[[v1:[^]*]] = "test_irdl_to_cpp.bar"() : () -> i32
// CHECK: %[[v2:[^]*]] = "test_irdl_to_cpp.hash"(%[[v0]], %[[v0]]) : (i32, i32) -> i32
// CHECK: return
// CHECK: }
func.func @test() {
    %0 = "test_irdl_to_cpp.bar"() : () -> i32
    %1 = "test_irdl_to_cpp.beef"(%0, %0) : (i32, i32) -> i32
    return
}
```

test_conversion.testd.mlir





[MLIR][IRDL] Added IRDL to C++ Translation #133982









IRDL: IR Definition Language





```
irdl.dialect cmath {
 irdl.type complex {
   %2 = irdl.any_of(%0, %1)
   irdl.parameters (elem: %2)
 irdl.operation norm {
   %0 = irdl.any
   %1 = irdl.parametric @cmath::@complex(%0)
   irdl.operands (in: %1)
   irdl.results (res: %0)
```





IRDL Devs



Mathieu Fehr email: mathieufehr@gmail.com github: math-fehr



Théo Degioanni github: Moxinilian discord: moxinilian



Ivan Ho email: ivan@hhkit.dev github: hhkit

Resources



https://godbolt.org/z/Ya54Whvd8