In-IR Optimizer Utility Testing

Nate Chandler

October 11, 2023

The problem

```
// Utility definition
void replaceAllUsesWith(
    Value old,
    Value new
) { ... }
```

```
// IR function
sil @handoff : $() -> () {

  %c = apply @getC() : $() -> C
  %c2 = copy %c : $C
  apply @takeC(%c2) : $(C) -> ()
  return
}
```

Constraints

- Specify elements
- Usual FileCheck test

```
// CHECK: sil @handoff
// CHECK: [[C:%[\d+]]] = apply
// CHECK: apply @takeC([[C]])

// IR function
sil @handoff : $() -> () {

  %c = apply @getC() : $() -> C
  %c2 = copy %c : $C
  apply @takeC(%c2) : $(C) -> ()
  return
}
```

Alternatives considered

- Utility passes.
  - ~~Choose arguments.~~
  - FileCheck.
- Unit tests.
  - Choose arguments.
  - ~~FileCheck.~~

```
// CHECK: sil @handoff
// CHECK: [[C:%[\d+]]] = apply
// CHECK: apply @takeC([[C]])

// IR function
sil @handoff : $() -> () {

  %c = apply @getC() : $() -> C
  %c2 = copy %c : $C
  apply @takeC(%c2) : $(C) -> ()
  return
}
```

The solution

```
// Utility definition
void replaceAllUsesWith(
    Value old,
    Value new
) { ... }
```

```
FunctionTest RAUWTest(
  "rauw",
  [](auto &args) {
    auto old = args.takeValue();
    auto new = args.takeValue();
    auto res = replaceAllUsesWith(old, new);
    function.dump();
  }
);
```

```
// CHECK: sil @handoff
// CHECK: [[C:%[\d+]]] = apply
// CHECK: apply @takeC([[C]])

// IR function
sil @handoff : $() -> () {
  specify_test "rauw %c2 %c"
  %c = apply @getC() : $() -> C
  %c2 = copy %c : $C
  apply @takeC(%c2) : $(C) -> ()
  return
}
```

The solution

In a bit more detail

```
// Utility definition
void replaceAllUsesWith(
    Value old,
    Value new
) { ... }

FunctionTest RAUWTest(
  "rauw",
  [](auto &args) {
    auto old = args.takeValue();
    auto new = args.takeValue();
    auto res = replaceAllUsesWith(old, new);
    function.dump();
  }
);
```

```
// CHECK: sil @handoff
// CHECK: [[C:%[\d+]]] = apply
// CHECK: apply @takeC([[C]])

// IR function
sil @handoff : $() -> () {
  specify_test "rauw %c2 %c"
  %c = apply @getC() : $() -> C
  %c2 = copy %c : $C
  apply @takeC(%c2) : $(C) -> ()
  return
}
```

Benefits

- Write test code once.
- Run on many functions.
- Specify arguments.

Conveniences

- Multiple specify_test instructions.
- Other function elements.
  - Operands
  - Instructions
  - Blocks
- Contextual notation.
  - @instruction[-1].operand[0]
  - @block
  - @function[+1]

Status

- Swift: In use.
- LLVM: In progress.