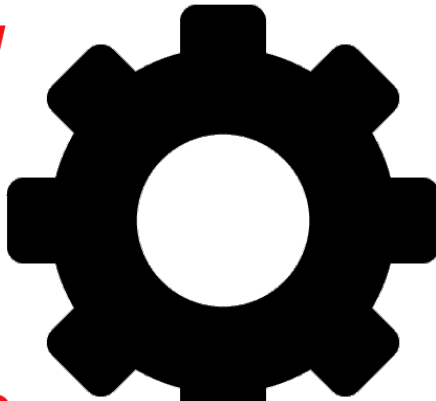


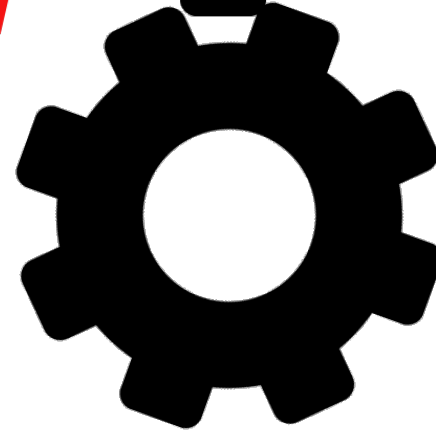
Advanced

T



pics

in



C

mpilers

Profiling



Simone Campanoni

simone.campanoni@northwestern.edu



Call graph in NOELLE

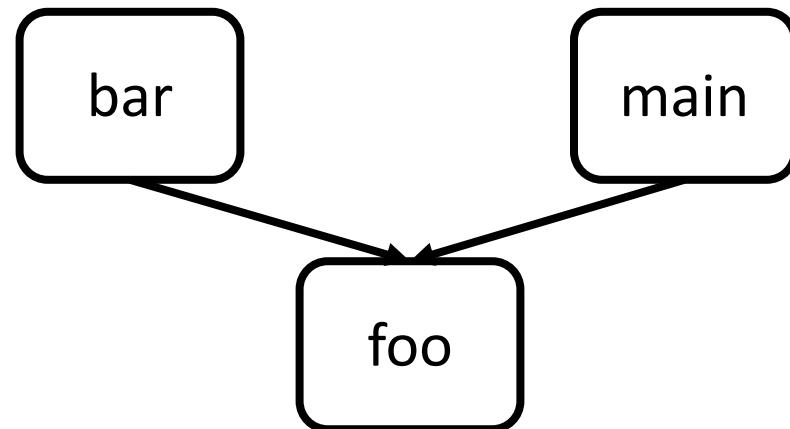
- Sources:
src/core/hotprofiler
- Main headers:
install/noelle/core/Hot.hpp
- Examples of passes using the abstraction:
examples/passes/profile

Outline

- How to profile with NOELLE
- Accessing profile information
- Loops and profiles

Profiles available

- Number of instructions of a given code region that has been executed
- Cumulative between all invocations of a code region



Normalize the code

Code must be normalized before you use NOELLE

- `noelle-norm MYIR.bc -o IR.bc`
or
- `noelle-simplification MYIR.bc -o IR.bc`

Generate, run, embed

- **Step 0: Generate** a binary that will be run to collect the profile
noelle-prof-coverage IR.bc standalone_binary -lm -lstdc++

The IR you want to profile

Generate, run, embed

- **Step 0: Generate** a binary that will be run to collect the profile
noelle-prof-coverage IR.bc standalone_binary -lm -lstdc++

The name of the binary

that will be generated with instrumentation code

Generate, run, embed

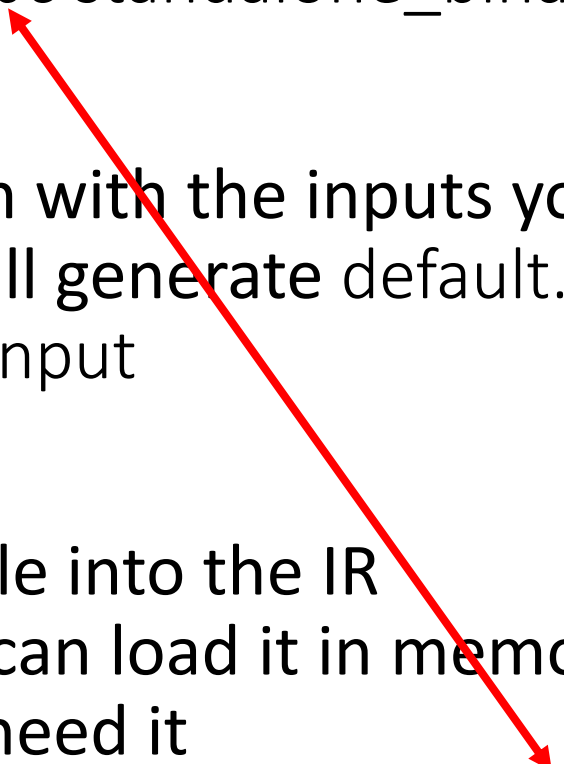
- **Step 0: Generate** a binary that will be run to collect the profile
noelle-prof-coverage IR.bc standalone_binary -lm -lstdc++

*Compilation options to use
to translate the input IR into binary
(e.g., libraries to link)*

Generate, run, embed

- **Step 0: Generate** a binary that will be run to collect the profile
noelle-prof-coverage IR.bc standalone_binary -lm -lstdc++
- **Step 1: Run** the program with the inputs you want
The execution will generate default.profrw
./standalone_binary myInput
./standalone_binary 10 20 30
./standalone_binary input_to_process.txt

Generate, run, embed

- **Step 0: Generate** a binary that will be run to collect the profile
noelle-prof-coverage IR.bc standalone_binary -lm -lstdc++
 - **Step 1: Run** the program with the inputs you want
The execution will generate default.profrw
./standalone_binary myInput
 - **Step 2: Embed** the profile into the IR
so that NOELLE can load it in memory automatically
every time you need it
noelle-meta-prof-embed default.profrw IR.bc -o IR_with_profile.bc
- 

Accessing the profile from your pass

- Every time you load NOELLE, the profile will be available and accessible via NOELLE's APIs

```
noelle-load -load ~/CAT/lib/CAT.so -CAT IR_with_profile.bc  
-disable-output
```

Outline

- How to profile with NOELLE
- Accessing profile information
- Loops and profiles

Fetching the profiles

```
/*  
 * Fetch NOELLE  
 */  
auto& noelle = getAnalysis<Noelle>();
```

```
auto hot = noelle.getProfiles();
```

```
if (!hot->isAvailable()){  
    return false;  
}  
errs() << "The profiler is available\n";
```

noelle/core/Hot.hpp

Profiles

- Queries you can do:
 - Has X executed?
(X = instruction, loop, function, basic block, SCC)
 - The number of times X is executed
 - Number of static instructions that compose X
 - How often a branch is taken

Self, total, static

- Static = number of static instructions that compose X
- Self = number of dynamic instructions executed within X for the whole program execution without counting instructions executed by callees
- Total = number of dynamic instructions executed within X for the whole program execution counting instructions executed by callees

APIs for all X

```
auto executed = hot->hasBeenExecuted(&F);
```

```
hot->getSelfInstructions(&F)  
hot->getTotalInstructions(&F)
```

```
hot->getStaticInstructions(&F)
```

```
hot->getDynamicTotalInstructionCoverage(&F)
```

Any pointer to any X



APIs for all X but SCC

`hot->getInvocations(&F)`

Any pointer to any X



Each X has extra X-specific APIs

```
hot->getAverageLoopIterationsPerInvocation(LS)
```

Outline

- How to profile with NOELLE
- Accessing profile information
- **Loops and profiles**

APIs

- NOELLE provides API to sort loops by their profile

```
noelle.sortByHotness(*loops);
```

```
auto loop = (*loops)[0];
```

Hottest loop of a program



Always have faith in your ability

Success will come your way eventually

Best of luck!