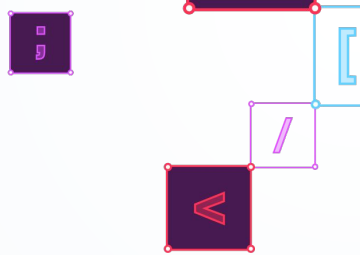
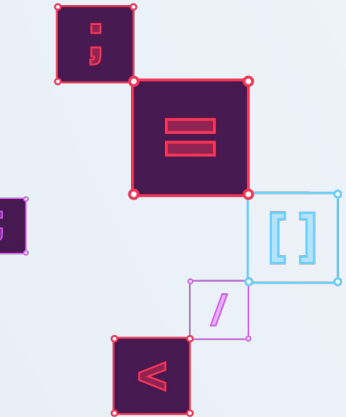


Incremental CSA



Philipp Dominik Schubert, Balázs Benics

sonar

Agenda

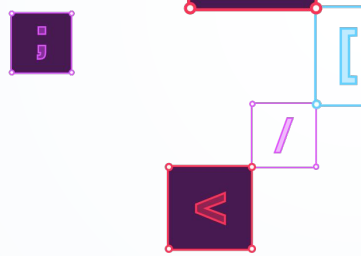
Motivation

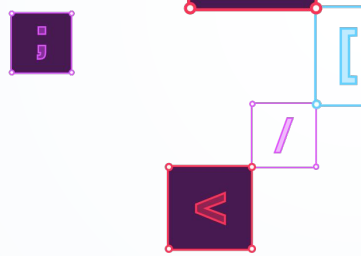
The engine as of today

The prototype

Preliminary results

Questions





Agenda

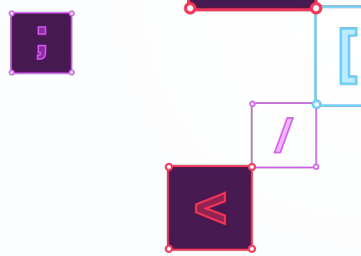
Motivation

The engine as of today

The prototype

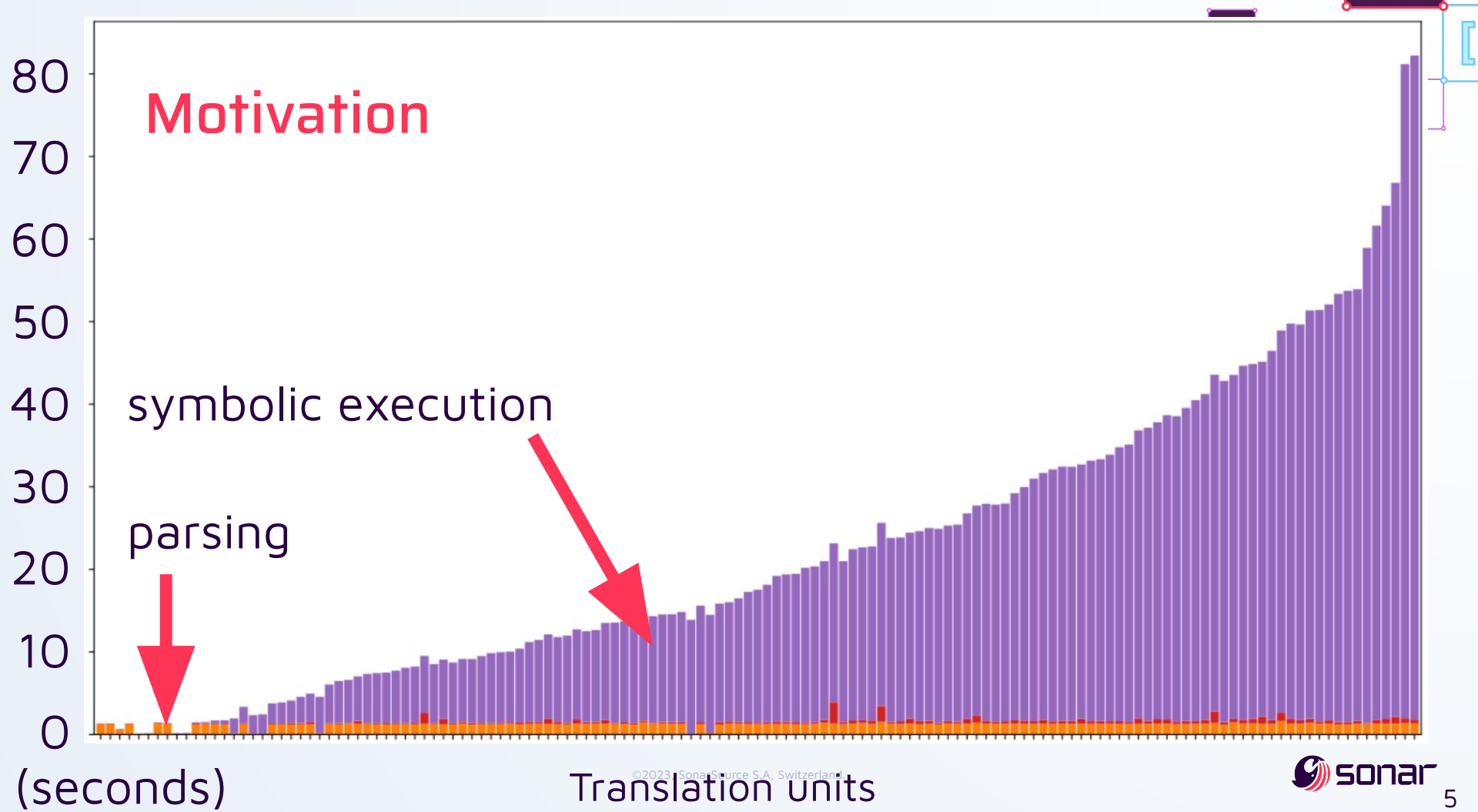
Preliminary results

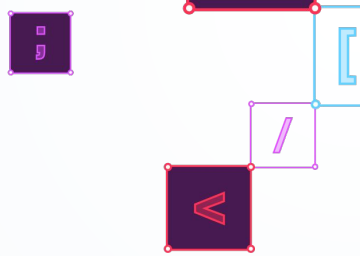
Questions



Motivation

Motivation





Agenda

Motivation

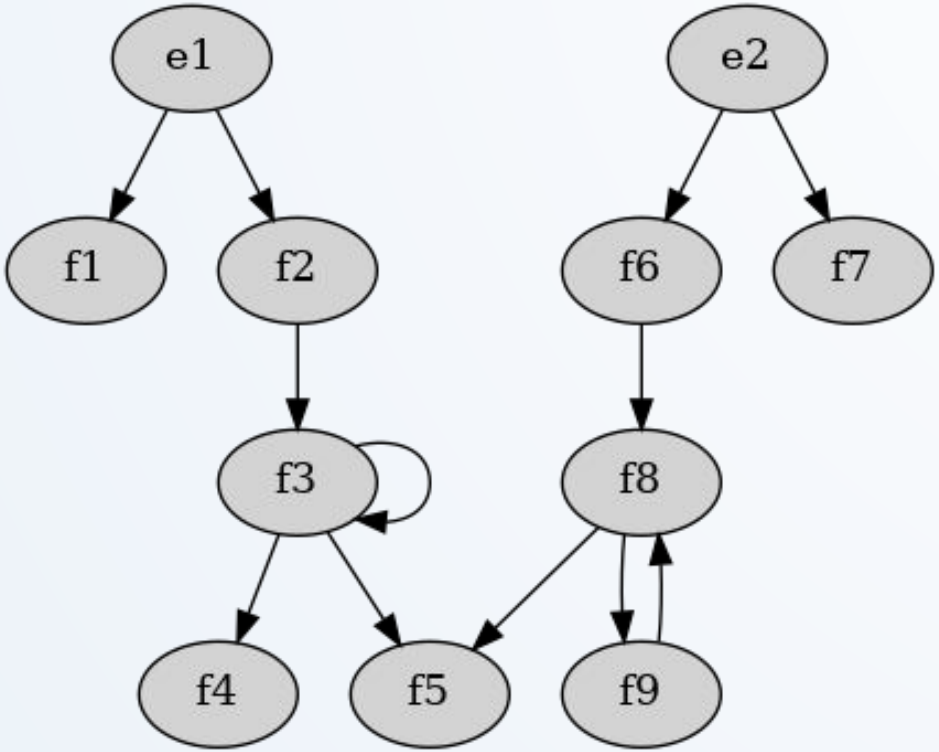
The engine as of today

The prototype

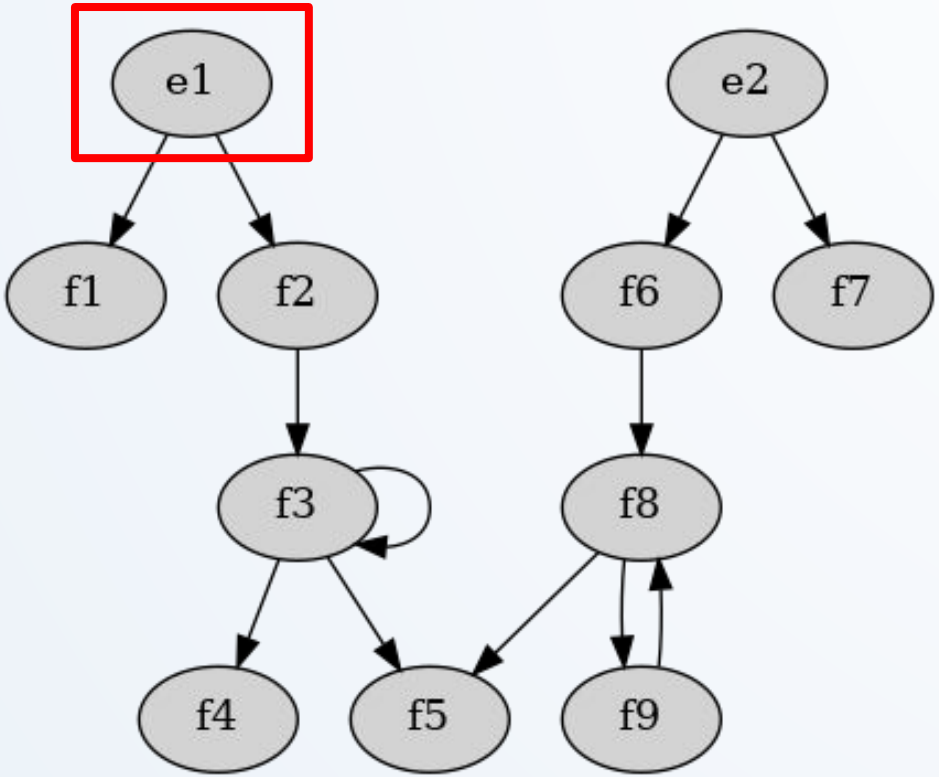
Preliminary results

Questions

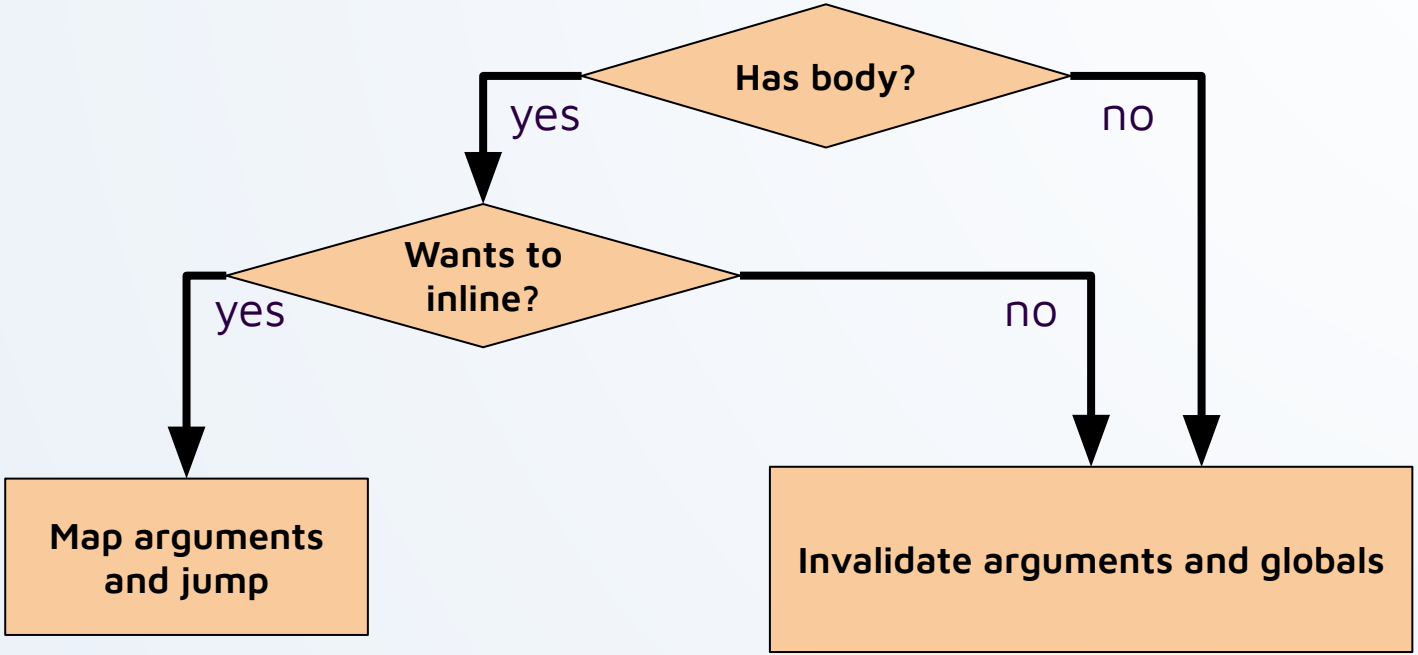
Call graph



Call graph



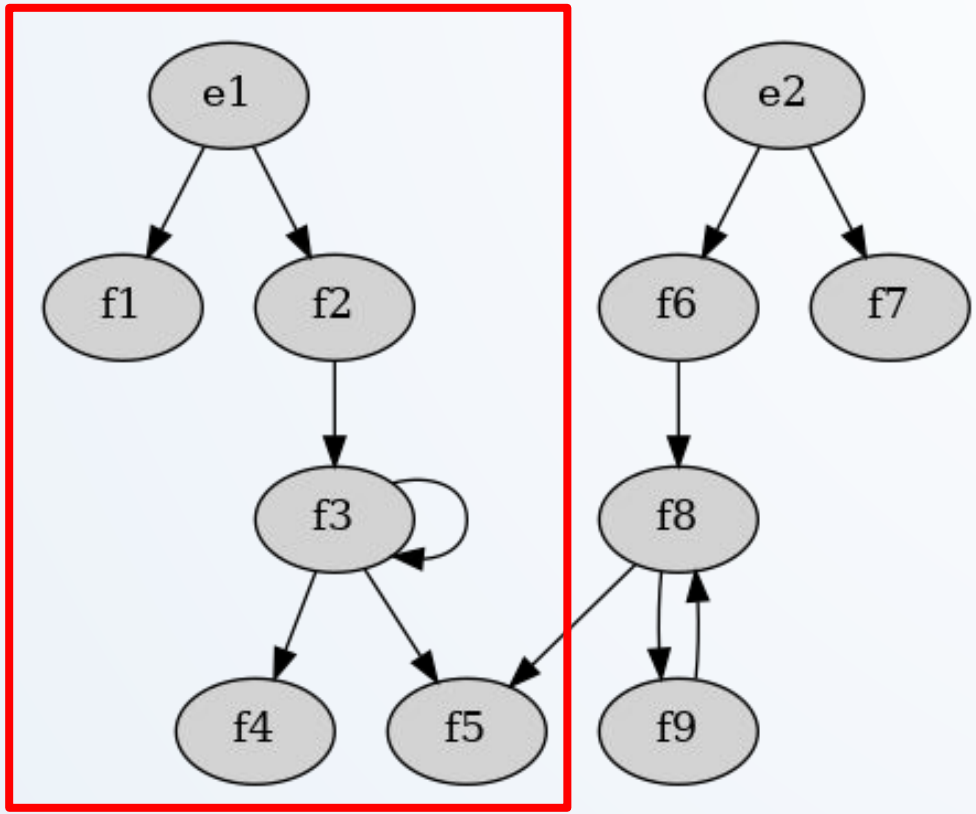
Inlining heuristic side effects



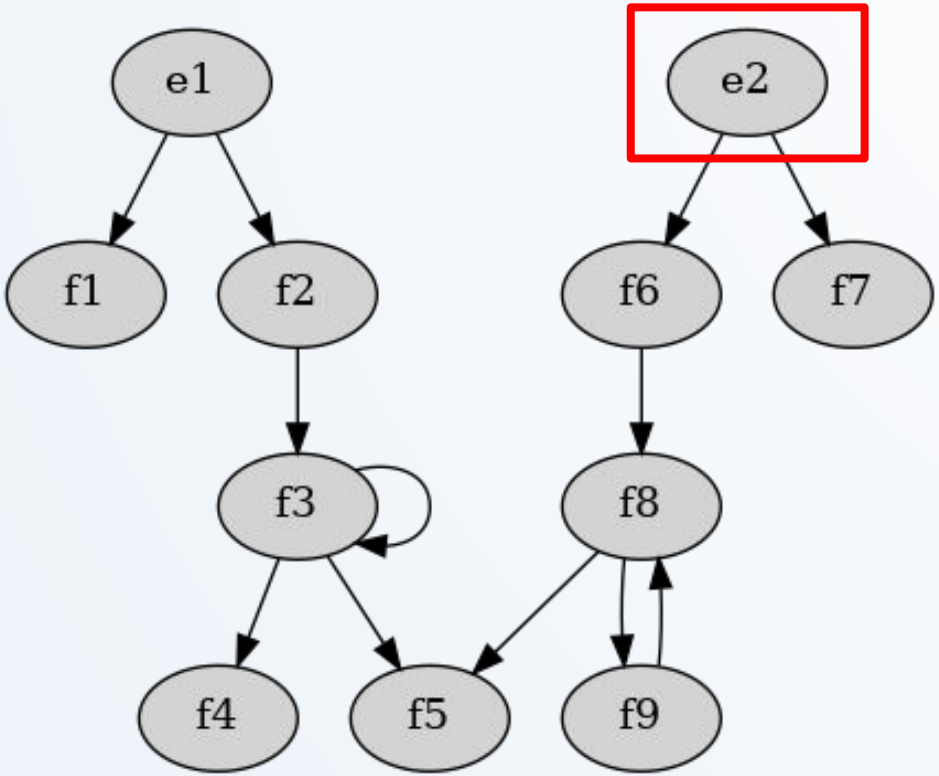
Inlining heuristic side effects

- Can decide not to inline, even if we could.
- Once inlined ⇉ always inlined.
- Inlined functions considered “covered”
- Might remember to never inline something again.

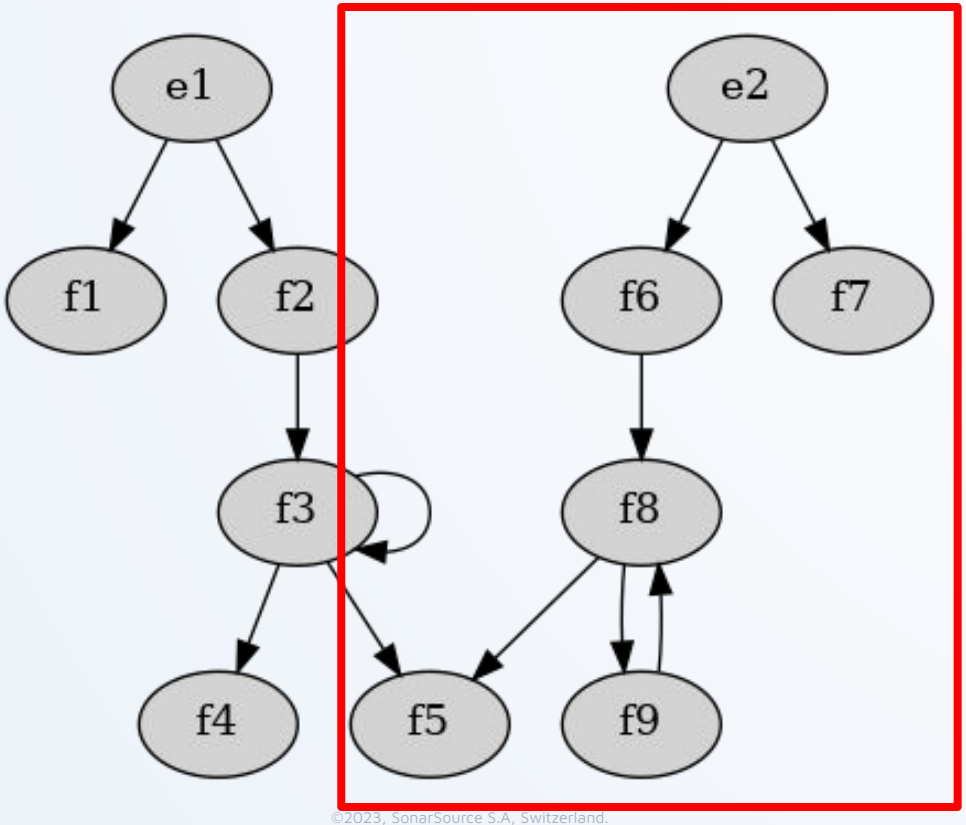
Call graph



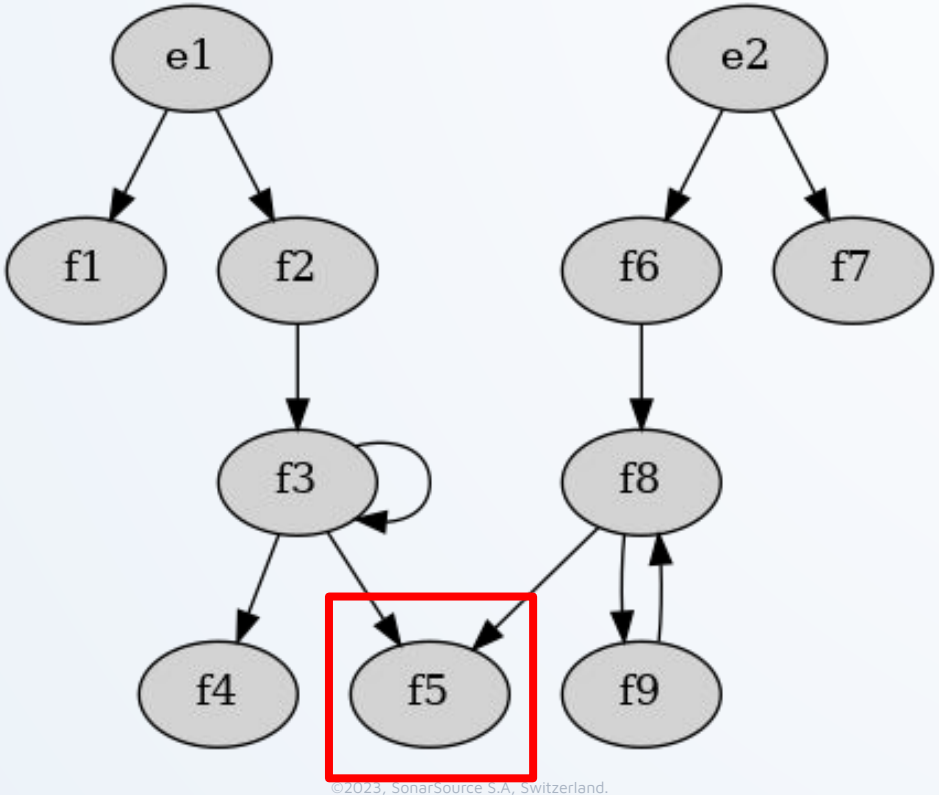
Call graph

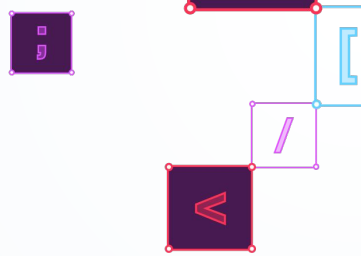


Call graph

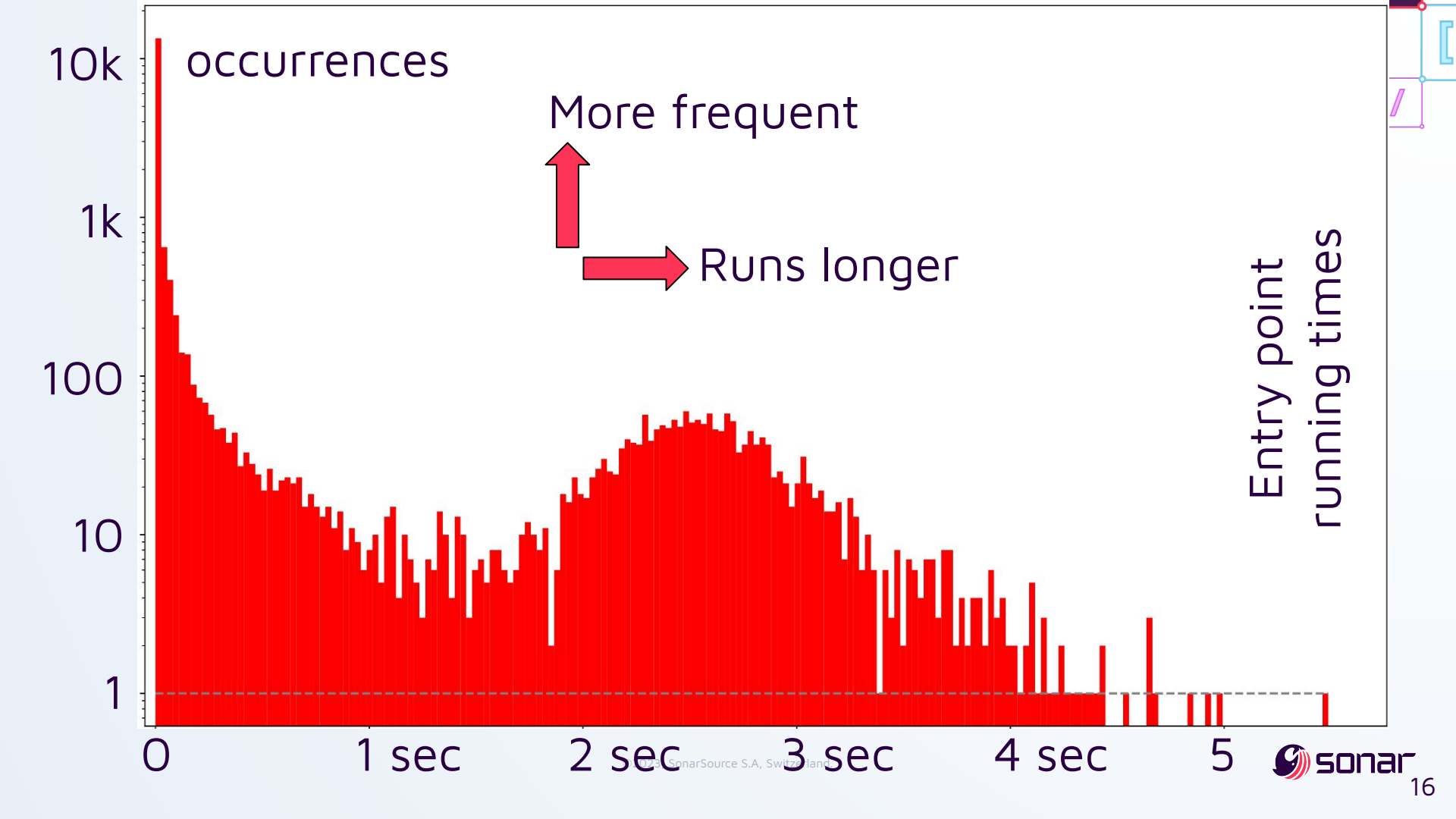


Call graph



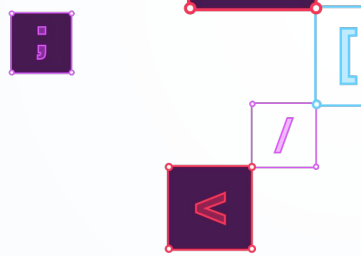


Analysis times per entry point



Let’s fix long analysis times…

- Serious engineering
- Unchanged since the dawn
- Multiple stakeholders
- High risk



Agenda

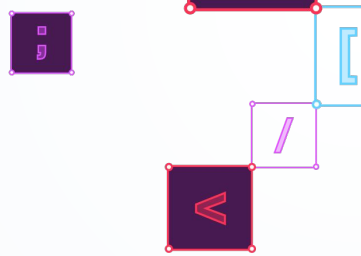
Motivation

The engine as of today

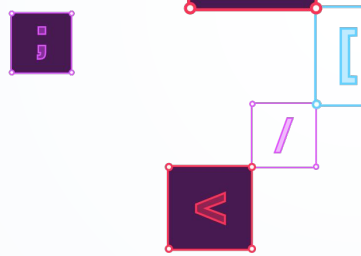
The prototype

Preliminary results

Questions



Guiding principles



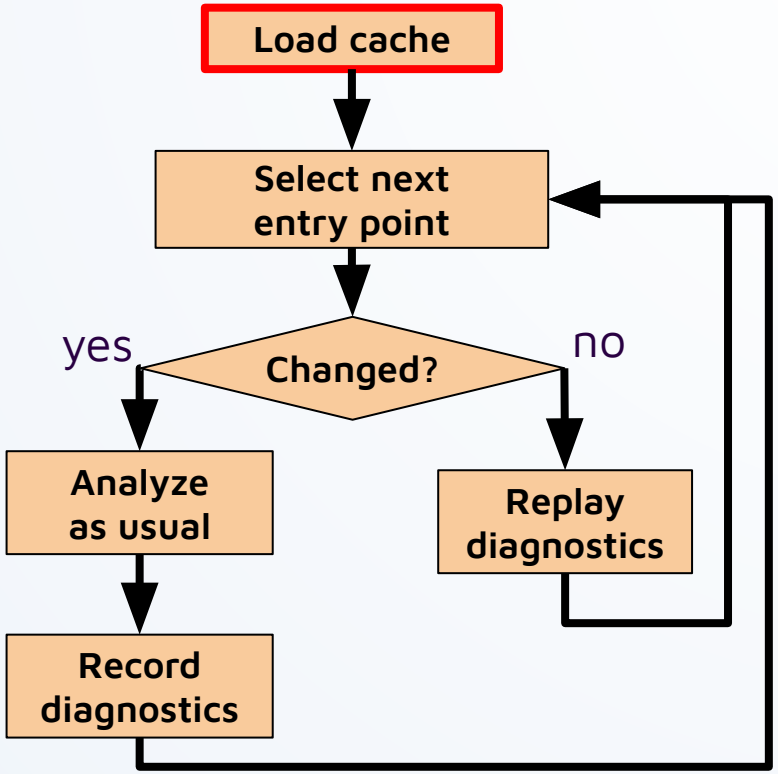
- Minimal
- Self-contained
- Significant speedup for the “usual” cases

Prototype architecture

- Analysis cache
- Oracle
- Report replayer
- Report recorder

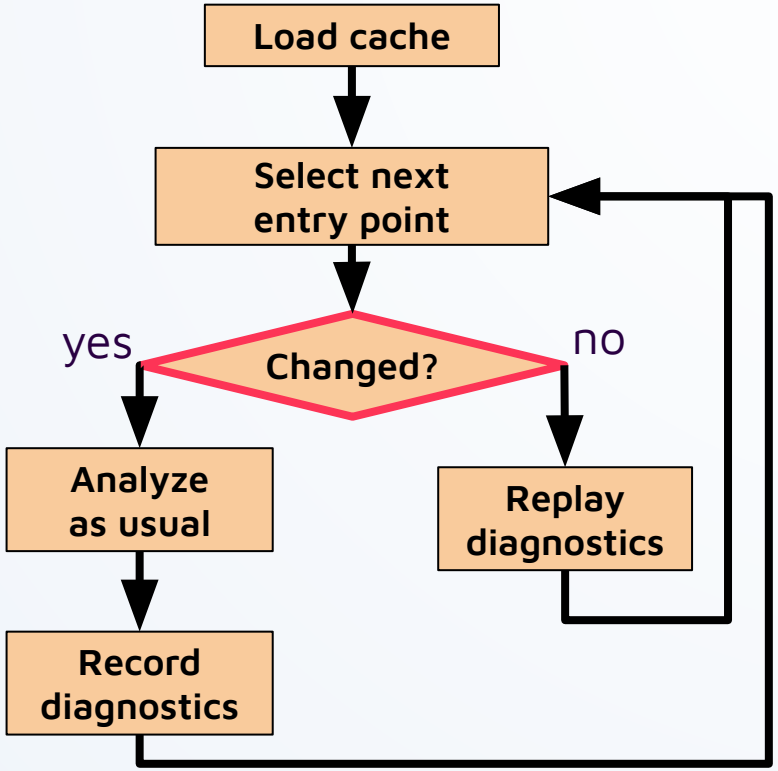
Prototype architecture

- Analysis cache
- Oracle
- Report replayer
- Report recorder



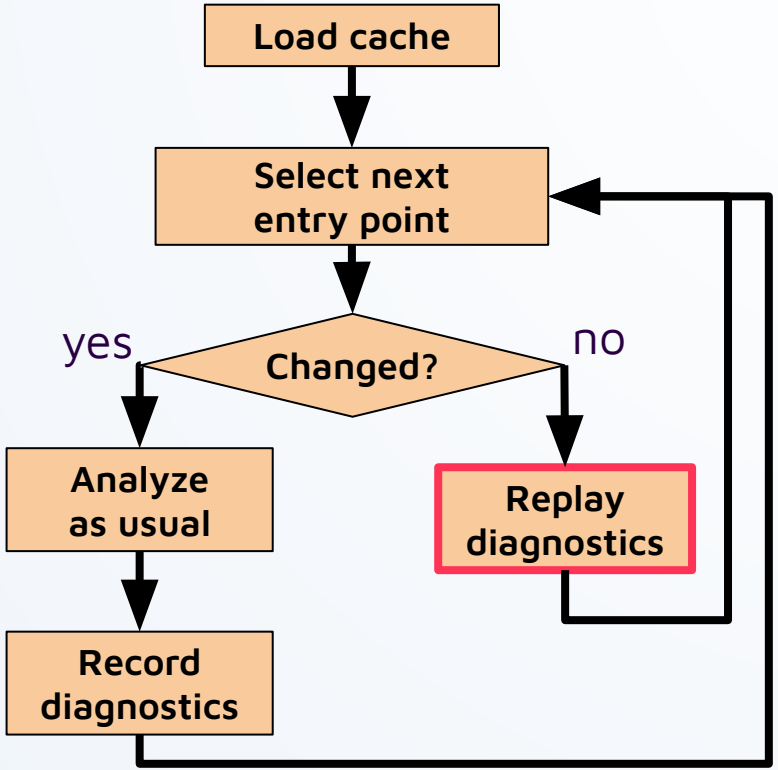
Prototype architecture

- Analysis cache
- Oracle
- Report replayer
- Report recorder



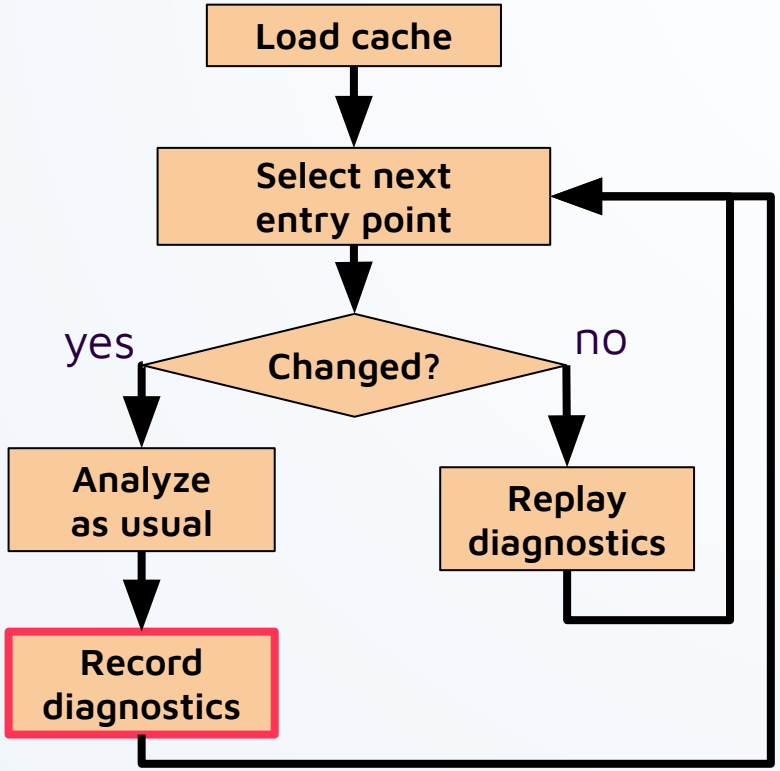
Prototype architecture

- Analysis cache
- Oracle
- Report replayer
- Report recorder



Prototype architecture

- Analysis cache
- Oracle
- Report replayer
- Report recorder



Anchor decls, relative references

c:@N@num@F@add#I#I#

```
FunctionDecl add 'int (int, int)'
|-ParmVarDecl used x 'int'
|-ParmVarDecl used y 'int'
`-CompoundStmt
  `-ReturnStmt
    `-BinaryOperator 'int' '+'
      |-ImplicitCastExpr 'int' <LValueToRValue>
      | `-DeclRefExpr 'int' lvalue ParmVar 'x' 'int'
      `-ImplicitCastExpr 'int' <LValueToRValue>
        `-DeclRefExpr 'int' lvalue ParmVar 'y' 'int'
```

```
namespace num {
   void add(int x, int y) {
      return x + y;
   }
}
```

Anchor decls, relative references

c:@N@num@F@add#I#I#

```
FunctionDecl add 'int (int, int)'
|-ParmVarDecl used x 'int'
|-ParmVarDecl used y 'int'
`-CompoundStmt
  `-ReturnStmt
    `-BinaryOperator 'int' '+'
      |-ImplicitCastExpr 'int' <LValueToRValue>
      | `-DeclRefExpr 'int' lvalue ParmVar 'x' 'int'
      `-ImplicitCastExpr 'int' <LValueToRValue>
        `-DeclRefExpr 'int' lvalue ParmVar 'y' 'int'
```

```
namespace num {
   void add(int x, int y) {
      return x + y;
   }
}
```

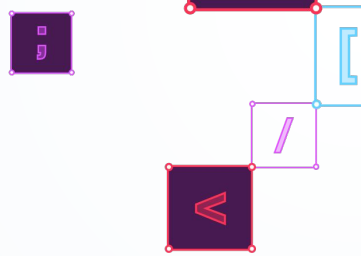
seq{2,0,0,1,0}

Relocatable diagnostics

- Anchor decl
- AST index sequence
- Getter function

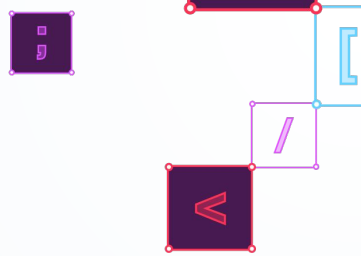
} source location

- Message



Oracle

- Preprocessor token watcher
- Hash:
  - source text
  - call dependencies
  - type dependencies



Diagnostic relocation

- Relocate diagnostics eagerly
- Might have absolute line refs
  - *"Control jumps to line 80"*
  - *"[...] call to alloca() on line 55 returned to caller"*
  - *"Loop condition is false. Execution continues on line 44"*

Agenda

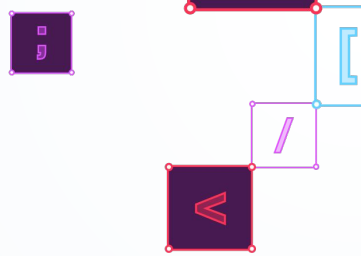
Motivation

The engine as of today

The prototype

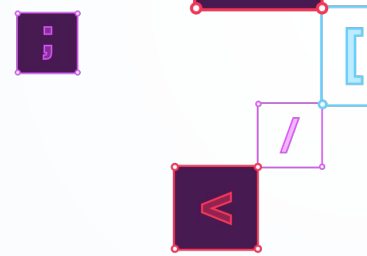
Preliminary results

Questions



Preliminary results

Whitespace, comment changes



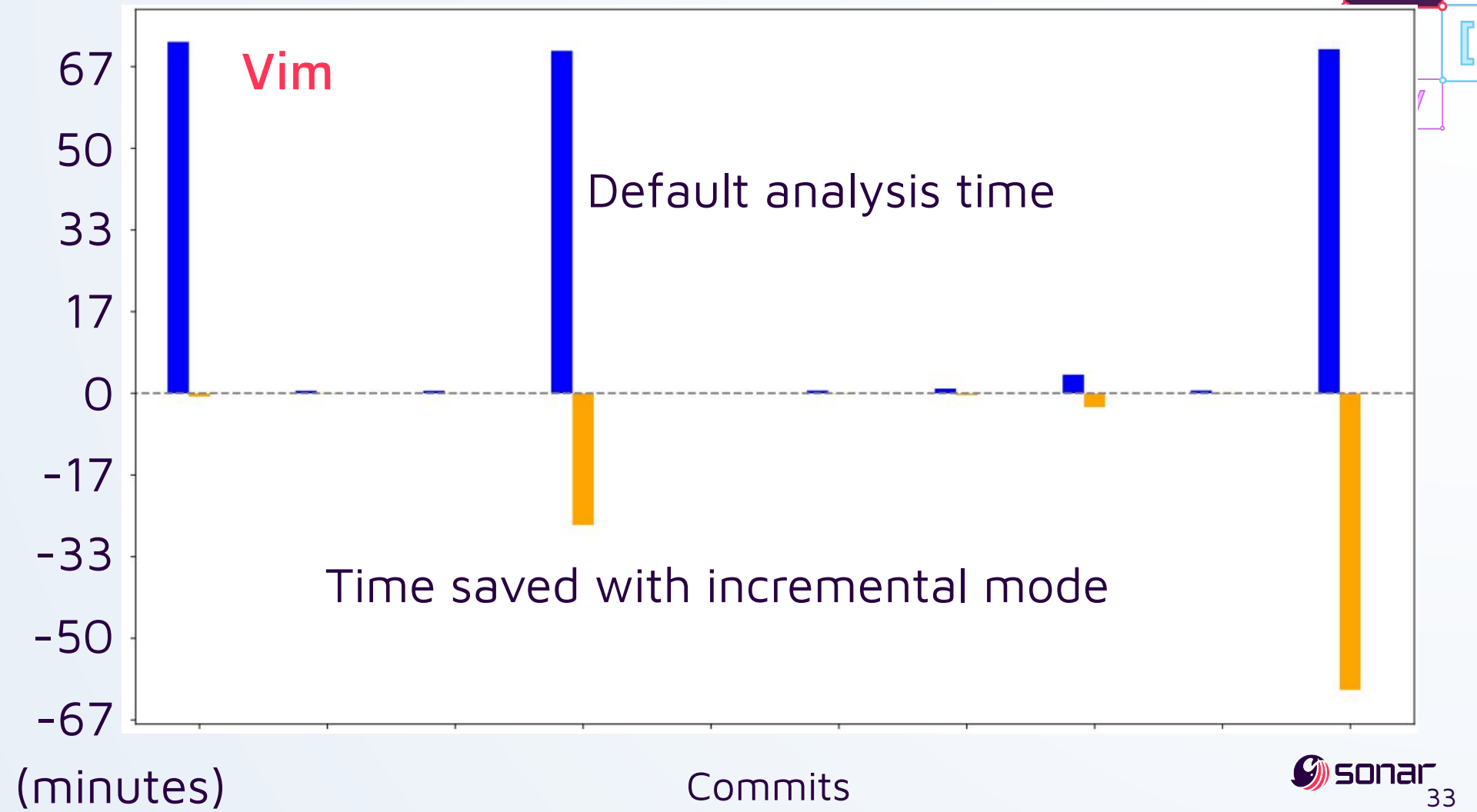
Default Analysis

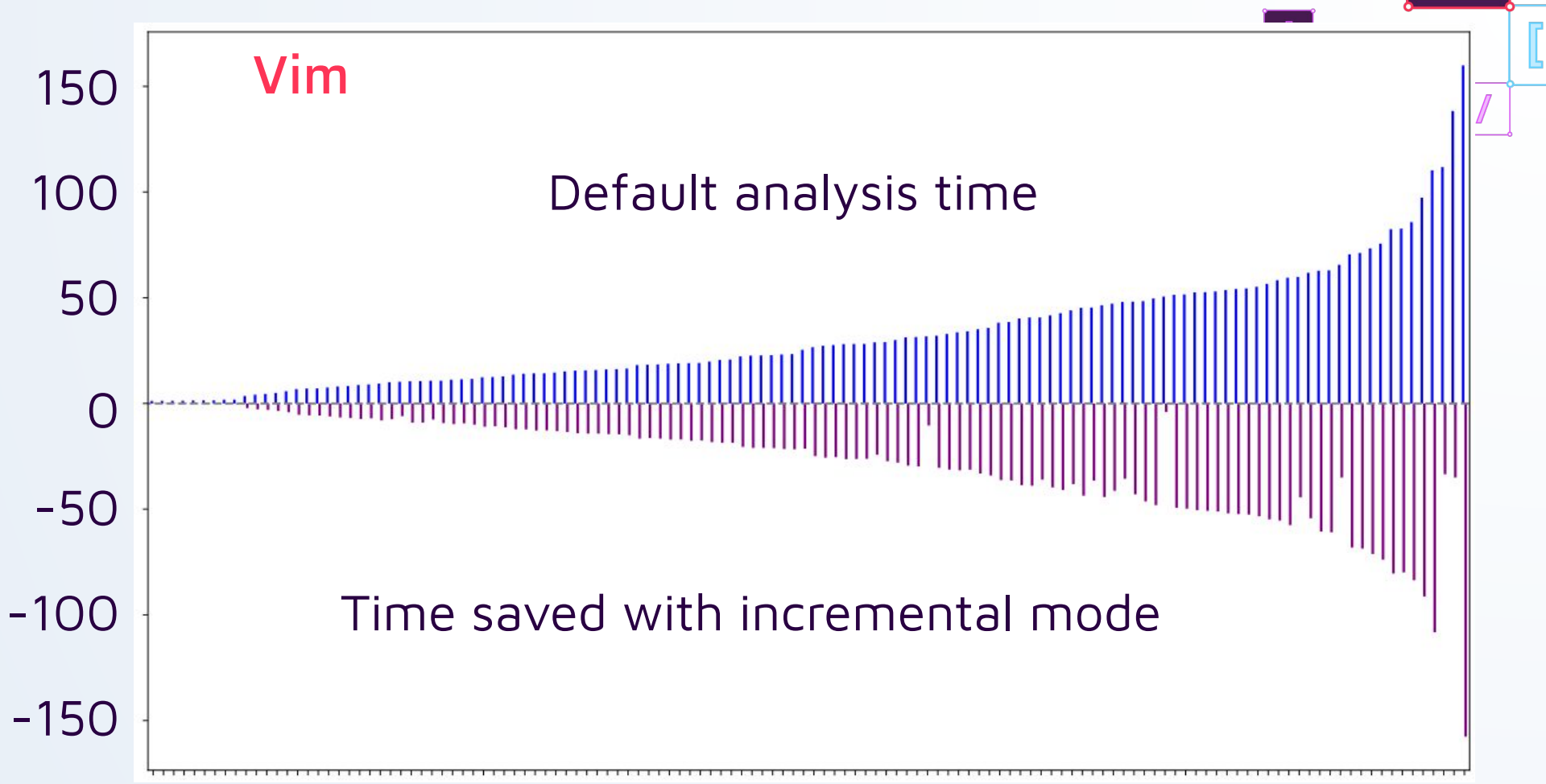
```
parsing: 48 ms
symbolicExecution: 1984 ms
```

Incremental Analysis

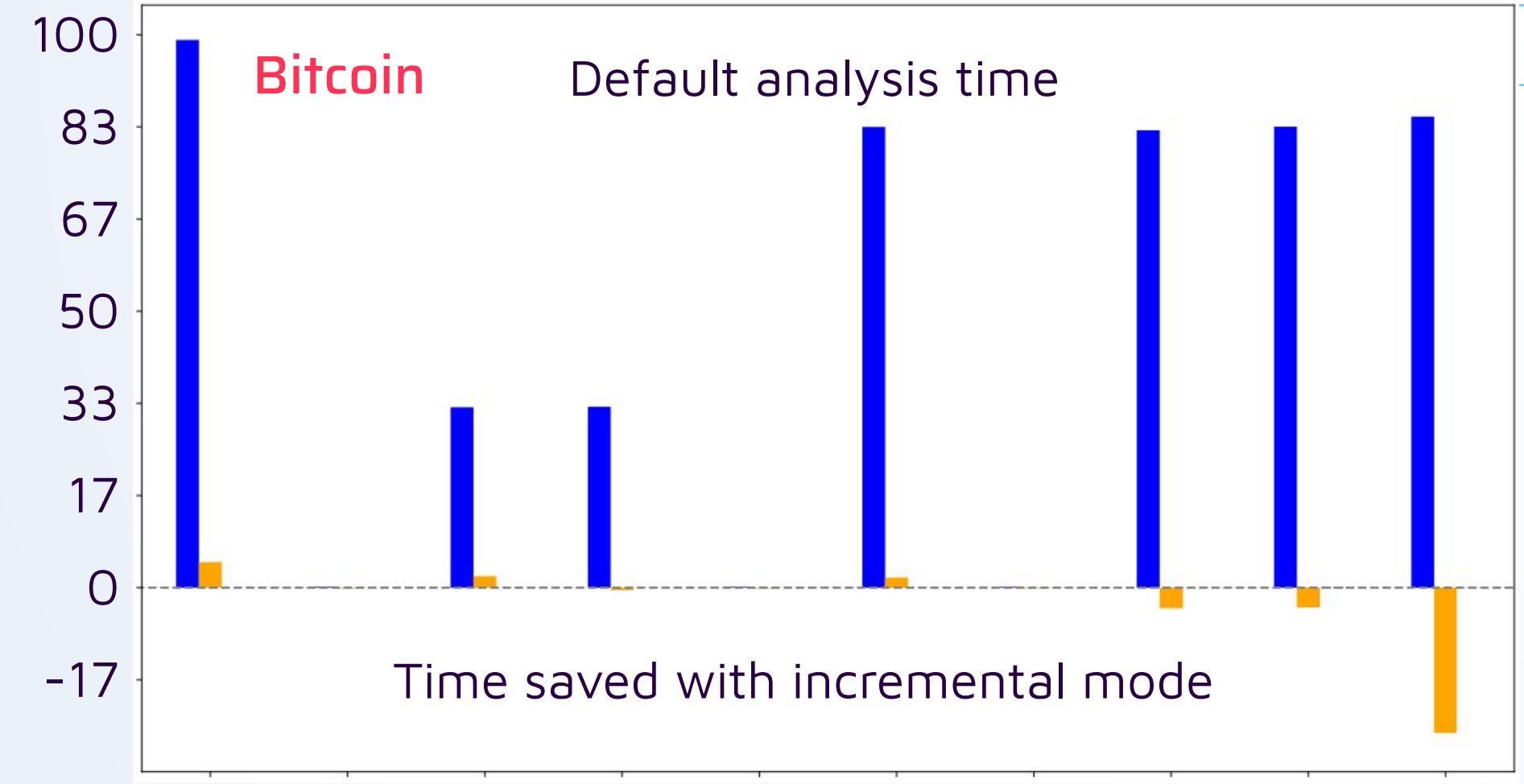
```
parsing: 62 ms
symbolicExecution: 42 ms
```

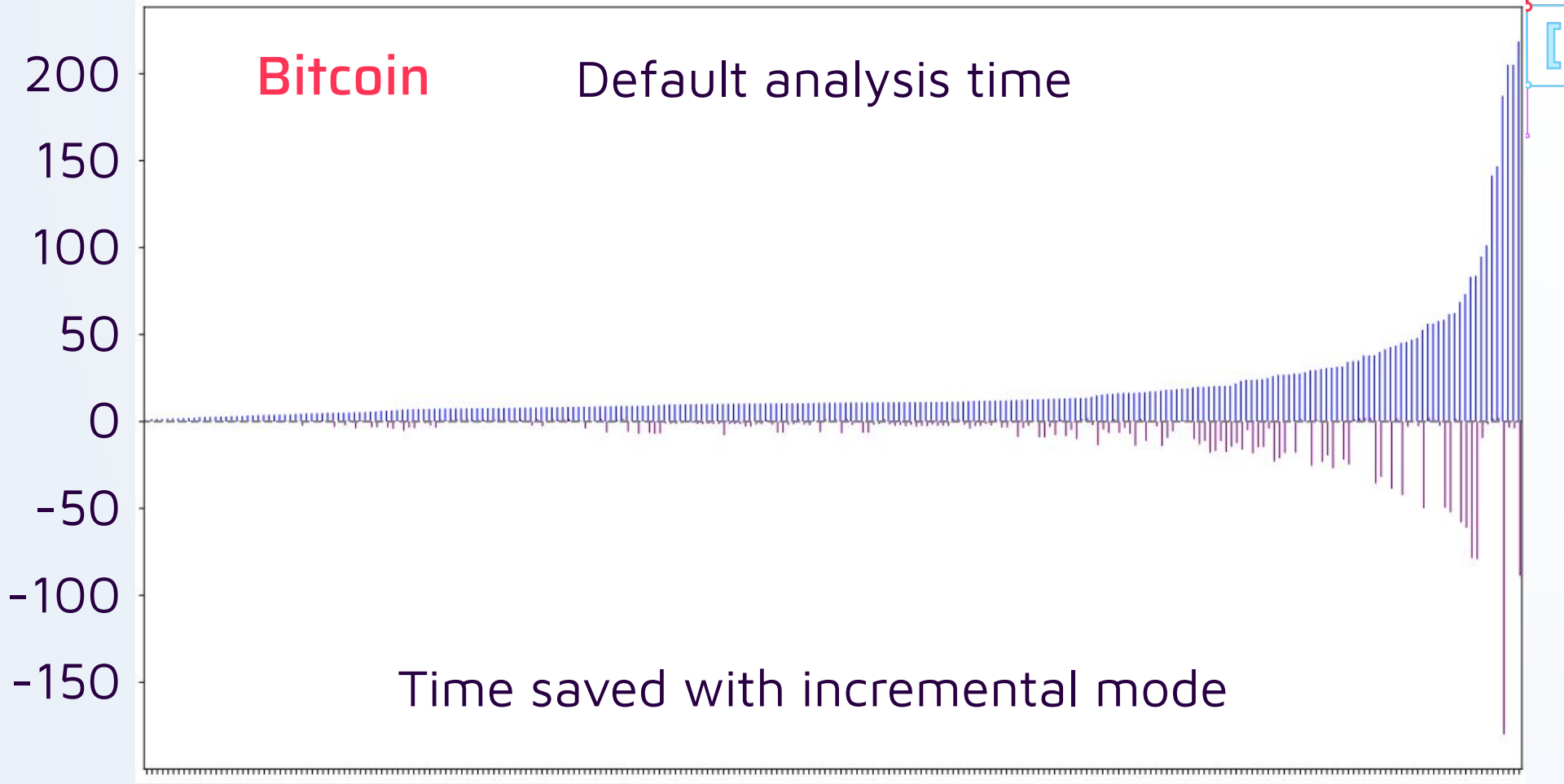
gzip:inflate.c


Vim
67
50
33
17
0
-17
-33
-50
-67
Default analysis time
Time saved with incremental mode
(minutes)
Commits


Vim
150
100
50
0
-50
-100
-150
Default analysis time
Time saved with incremental mode
(seconds)
Translation units

(commit 9)


100
83
67
50
33
17
0
-17
Bitcoin
Default analysis time
Time saved with incremental mode
(minutes)
Commits



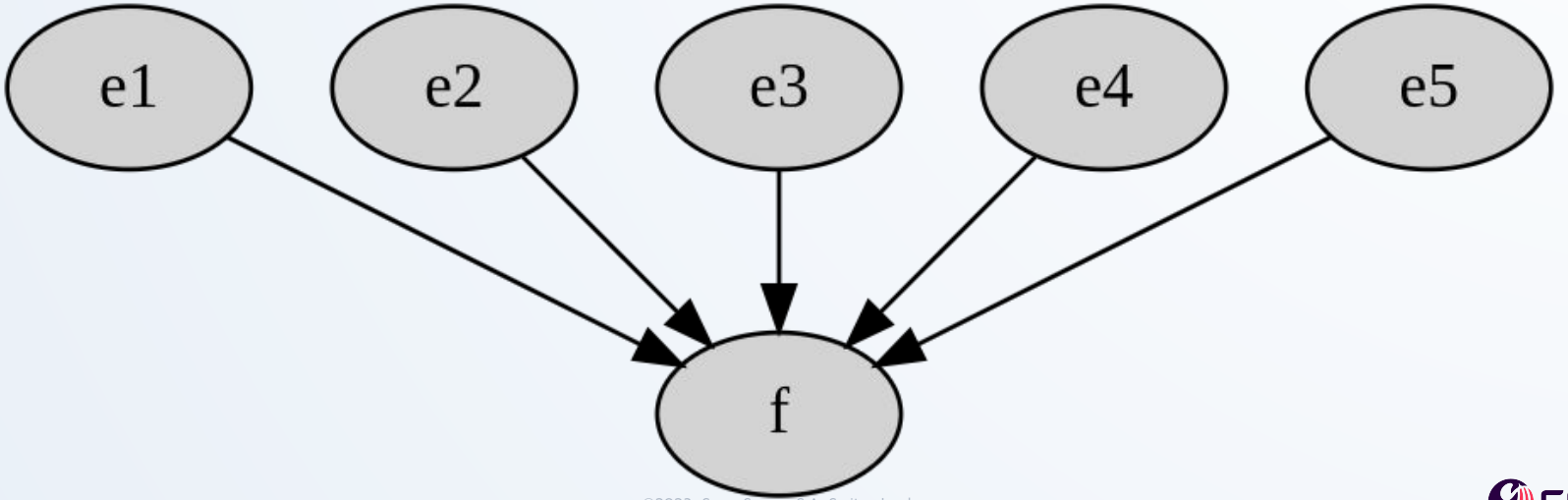
(seconds)

Translation units

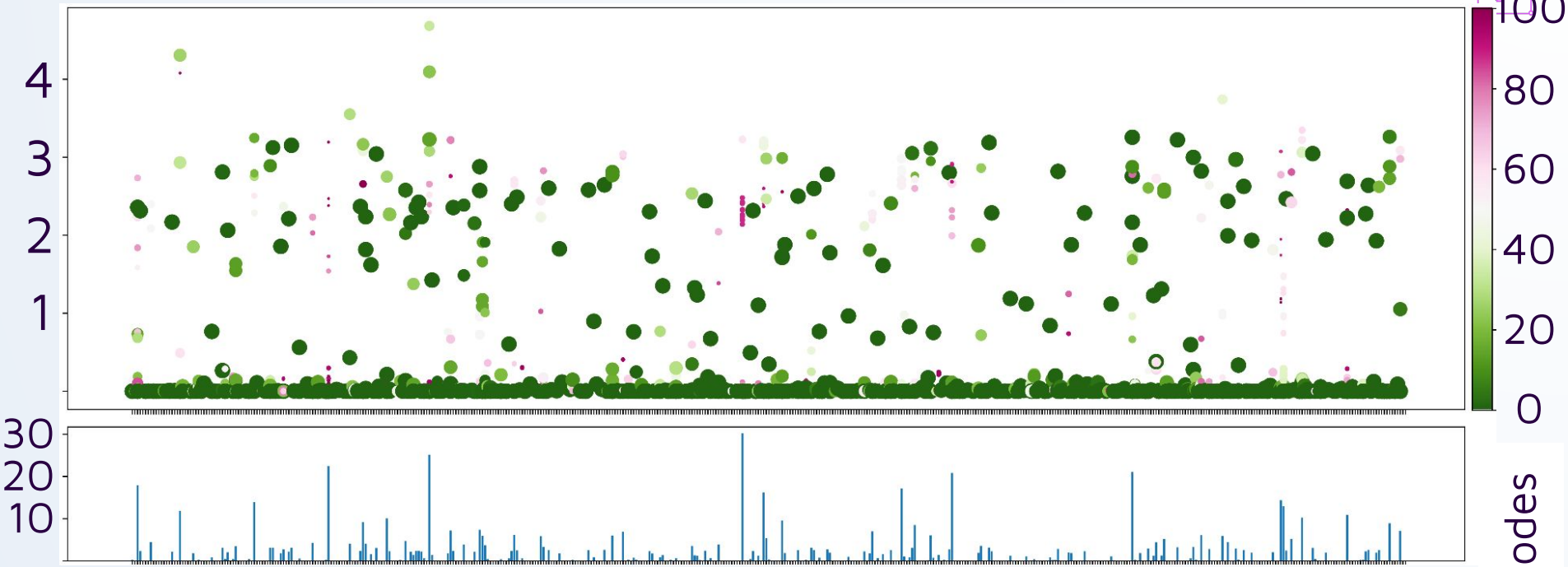
(commit 9)

Weaknesses

Changes in commonly inlined function (“sharing”)



Entry point running times and sharing

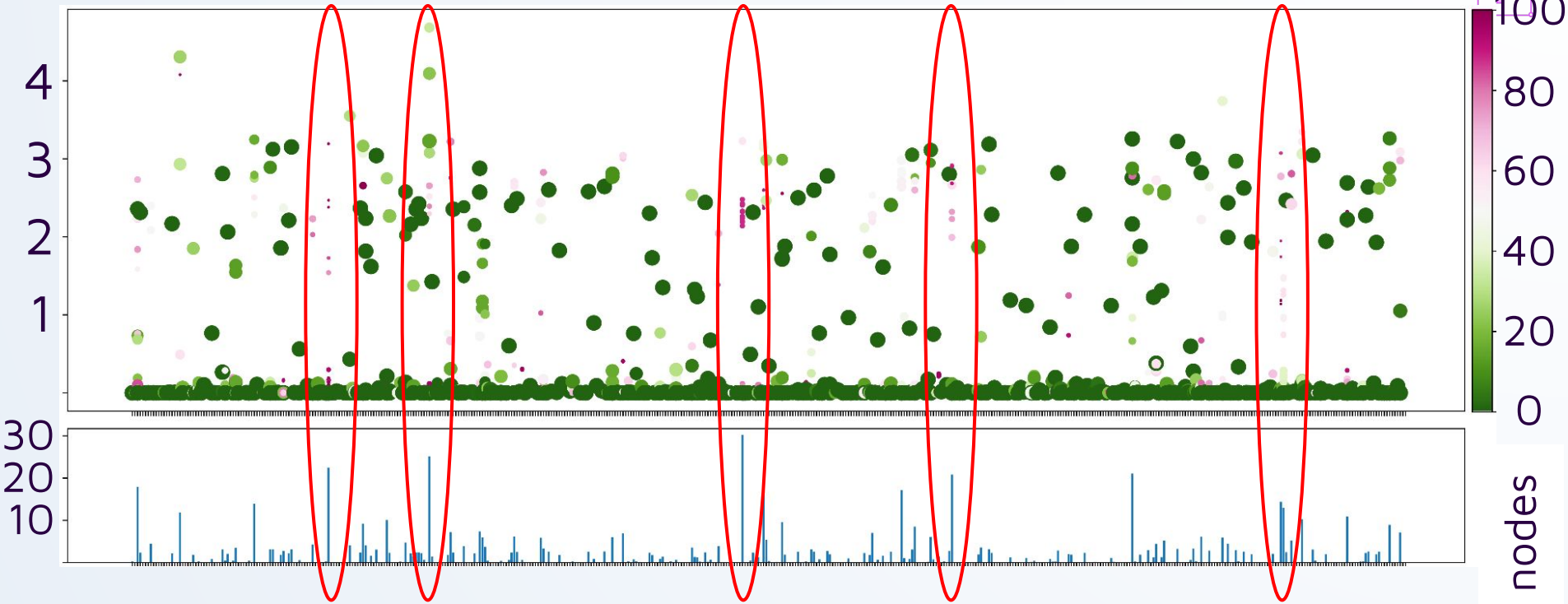


(seconds)

Translation units

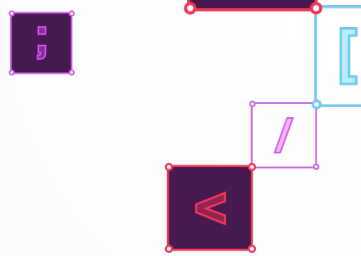
(default run)

Entry point running times and sharing



Conclusion

- Moderate improvements overall
- Only a few cache-hits for C++
- A lot of potential
- Works well for trivial, narrow changes



Agenda

Motivation

The engine as of today

The prototype

Preliminary results

Questions