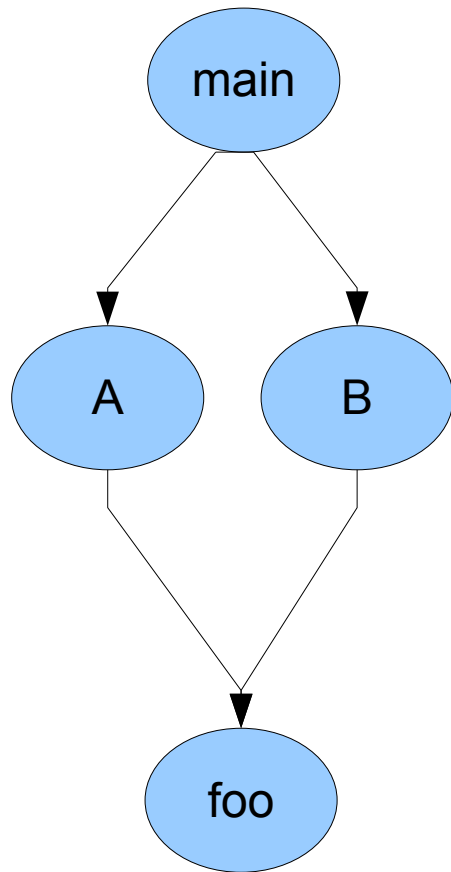


# Inferred Call Path Profiling (ICPP)

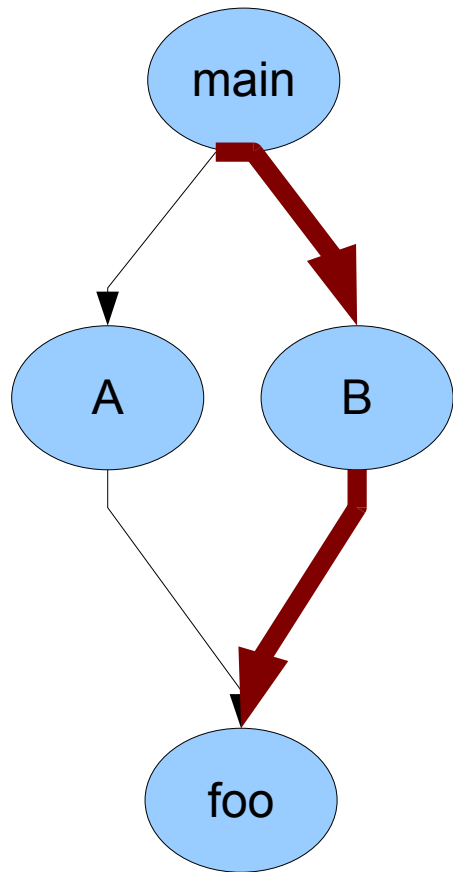
Todd Mytkowicz, Devin Coughlin, and Amer Diwan  
University of Colorado at Boulder

# Context helps developers



\$> ./a.out  
segfault

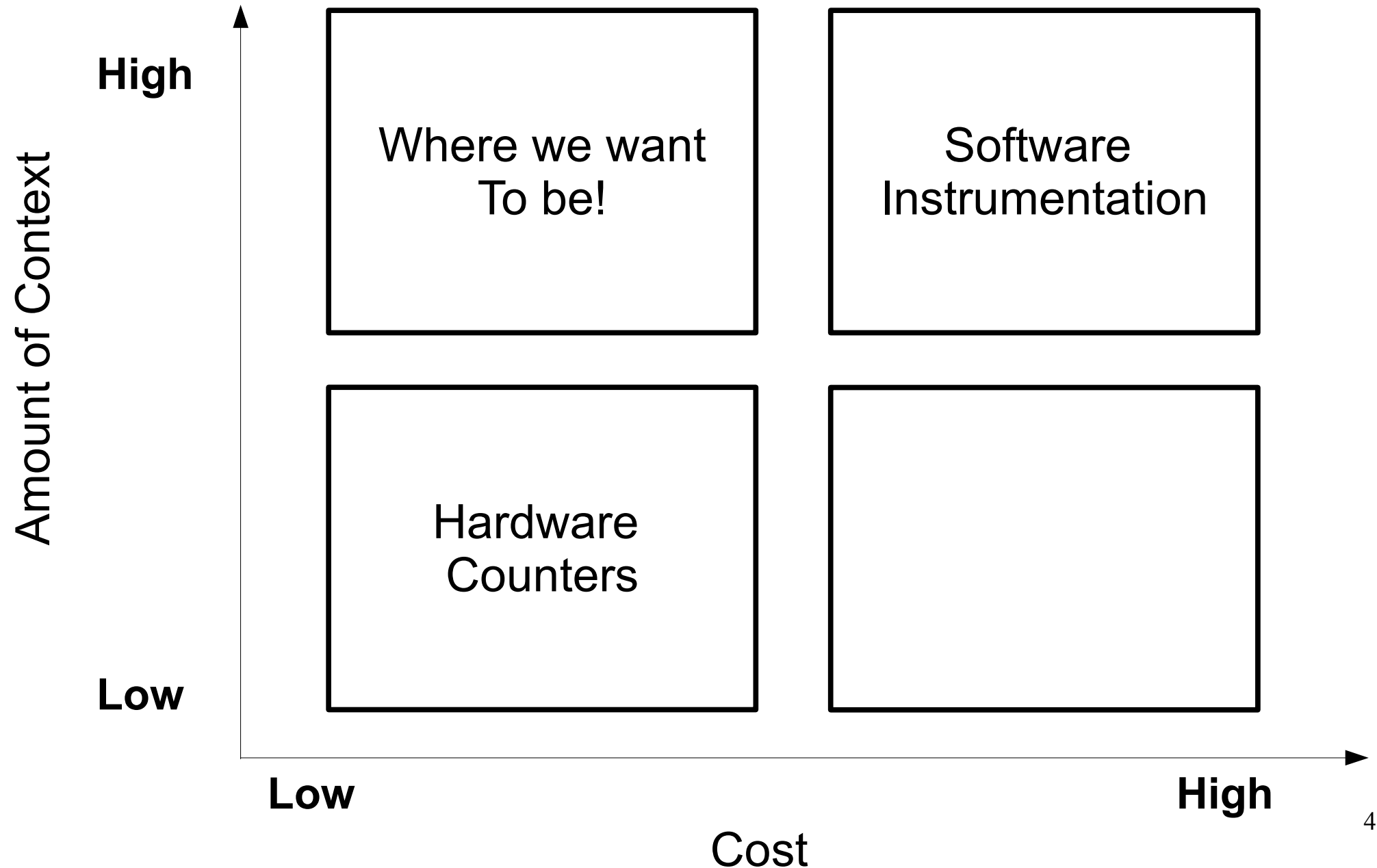
# Context helps developers



```
$> ./a.out  
segfault @ foo  
@ B  
@ main
```

Calling context is expensive

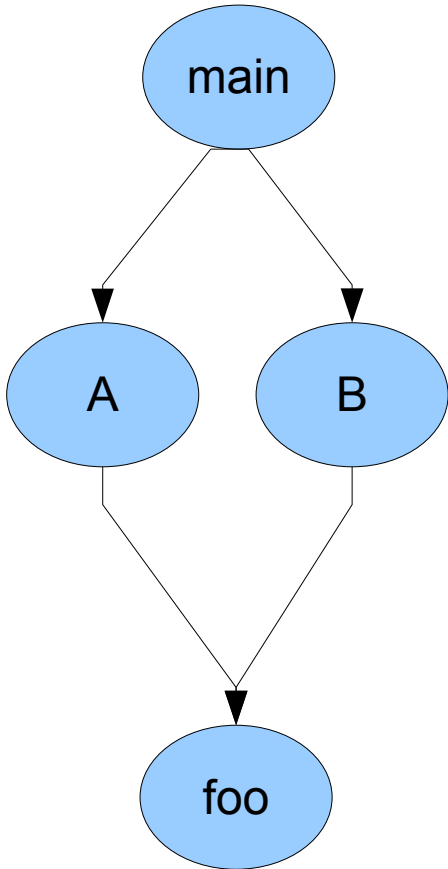
# Challenge: low cost, high context



# Approach

calling context: **48**

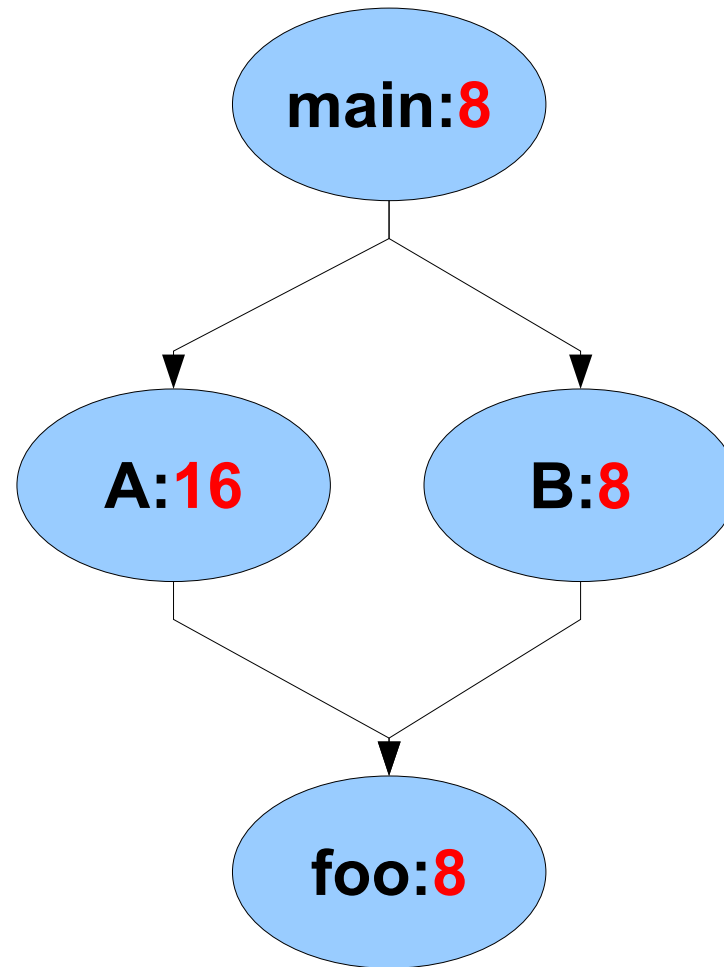
call path: **main – A – foo**



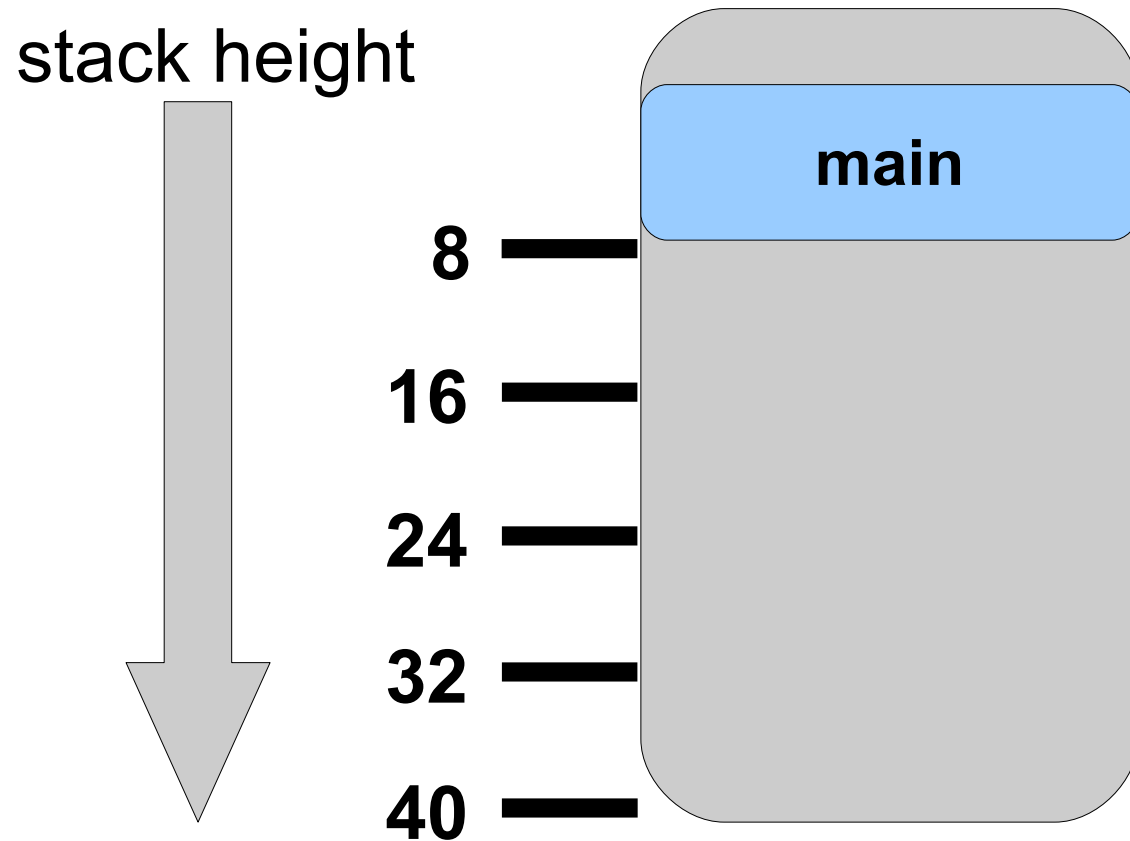
**Phase 1:** capture calling context

**Phase 2:** map calling context to call path

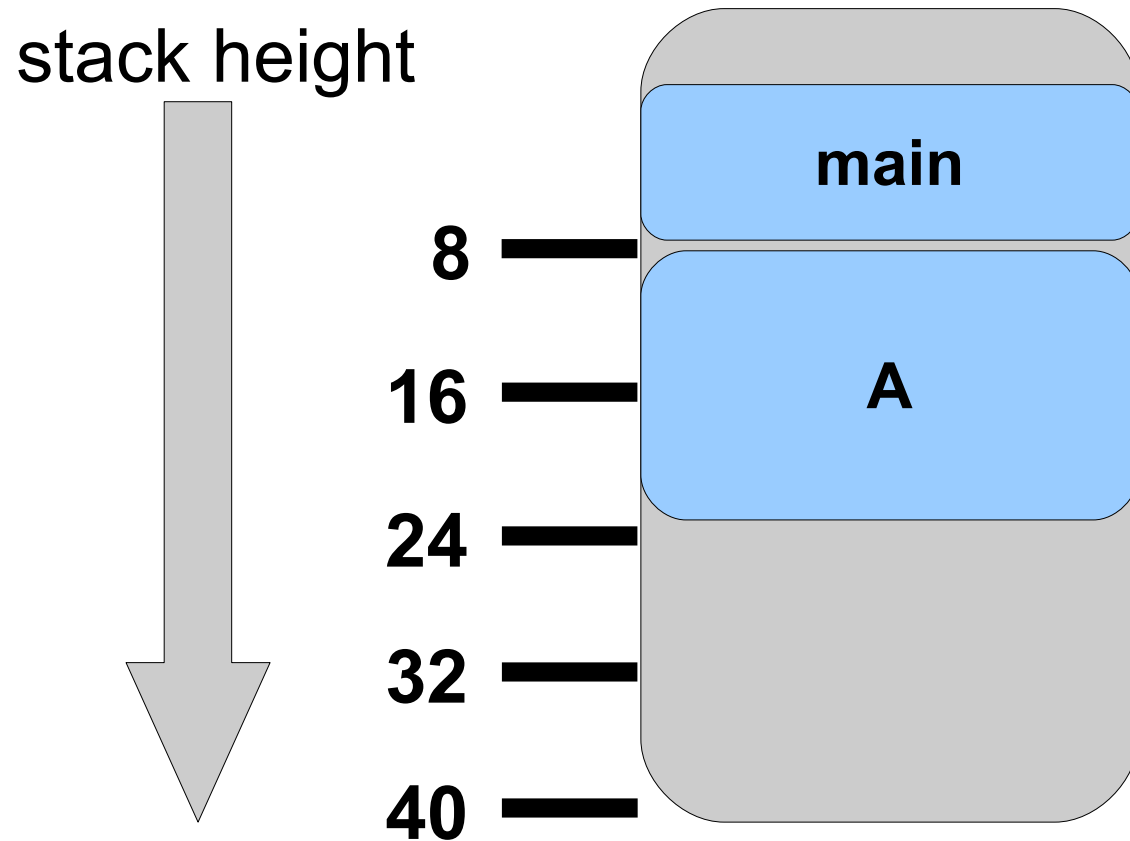
# Phase 1: calling context with ICPP



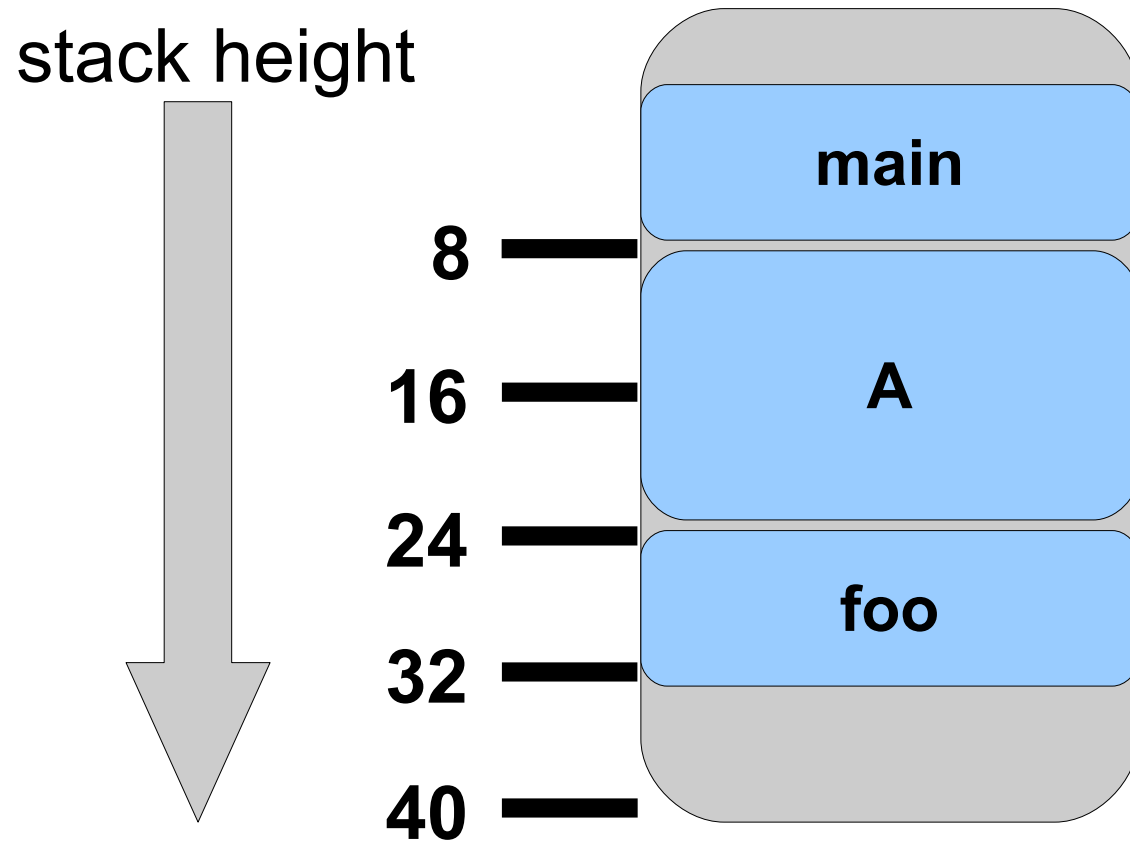
# Phase 1: calling context with ICPP



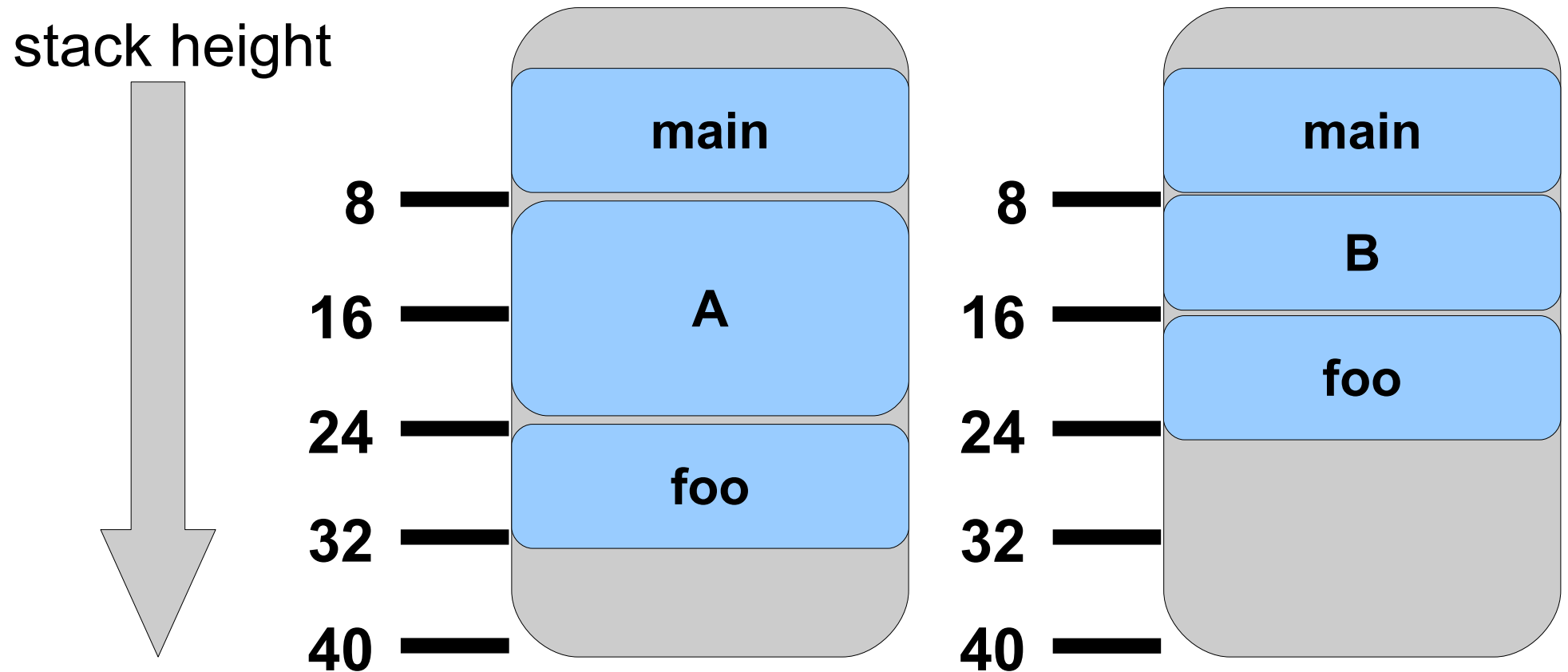
# Phase 1: calling context with ICPP



# Phase 1: calling context with ICPP



# Phase 1: calling context with ICPP

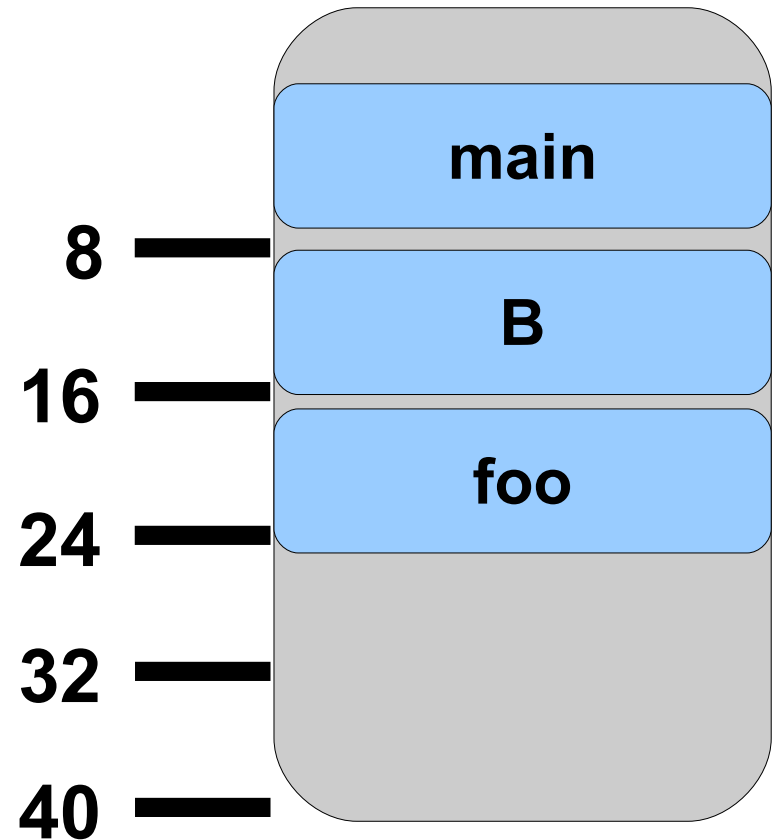
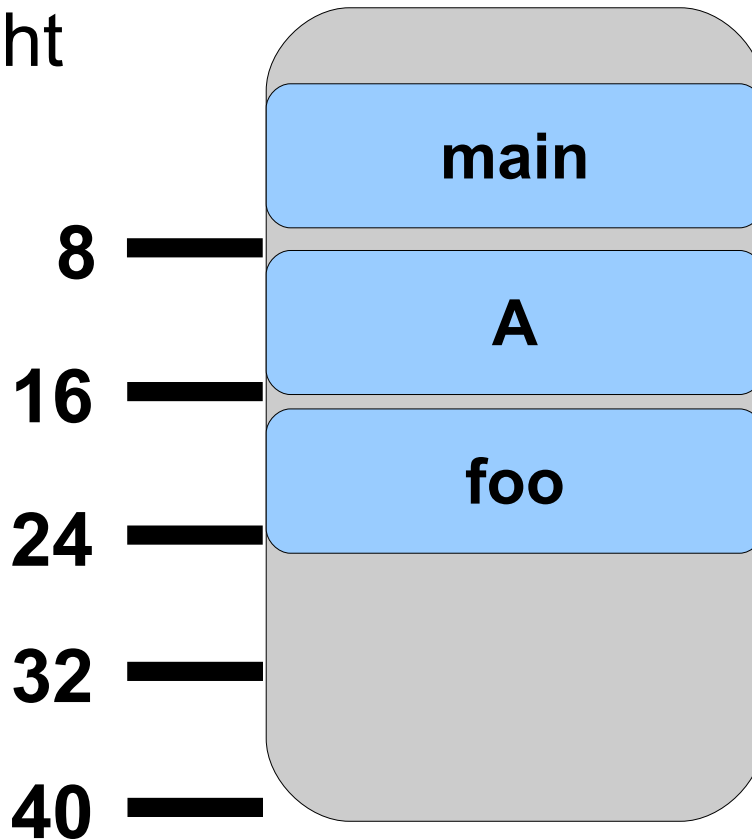
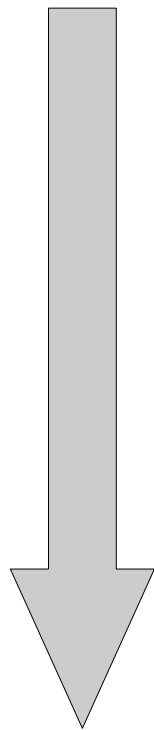


PC/SP is calling context without computation

# Problem: *Ambiguity*

More than one call path maps to same PC/SP

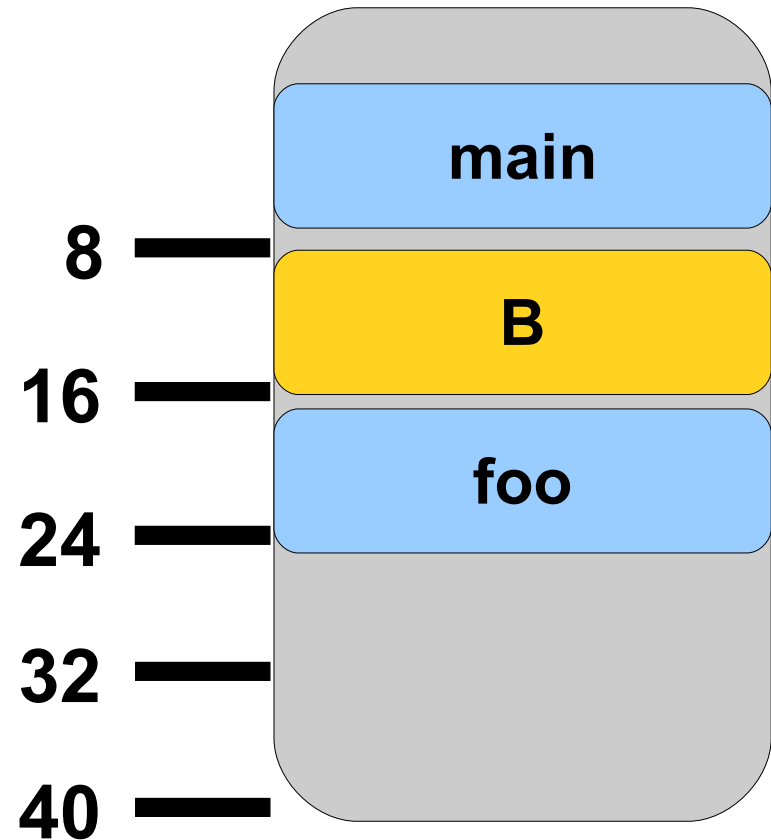
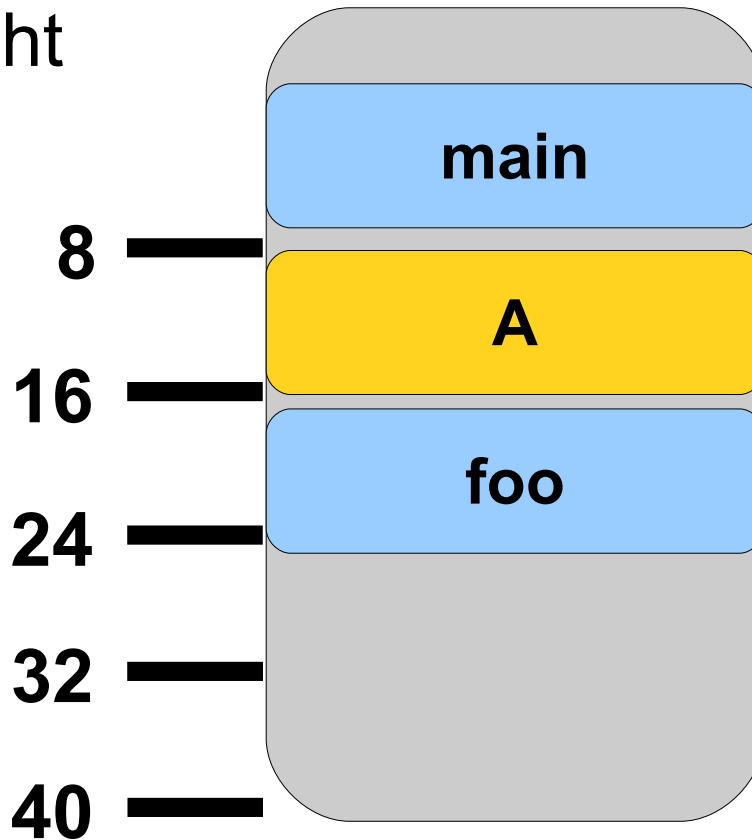
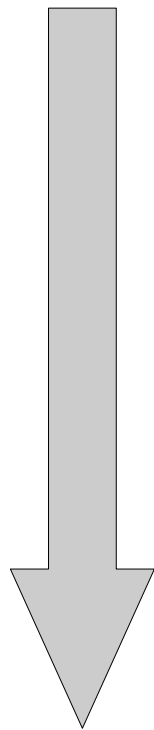
stack height



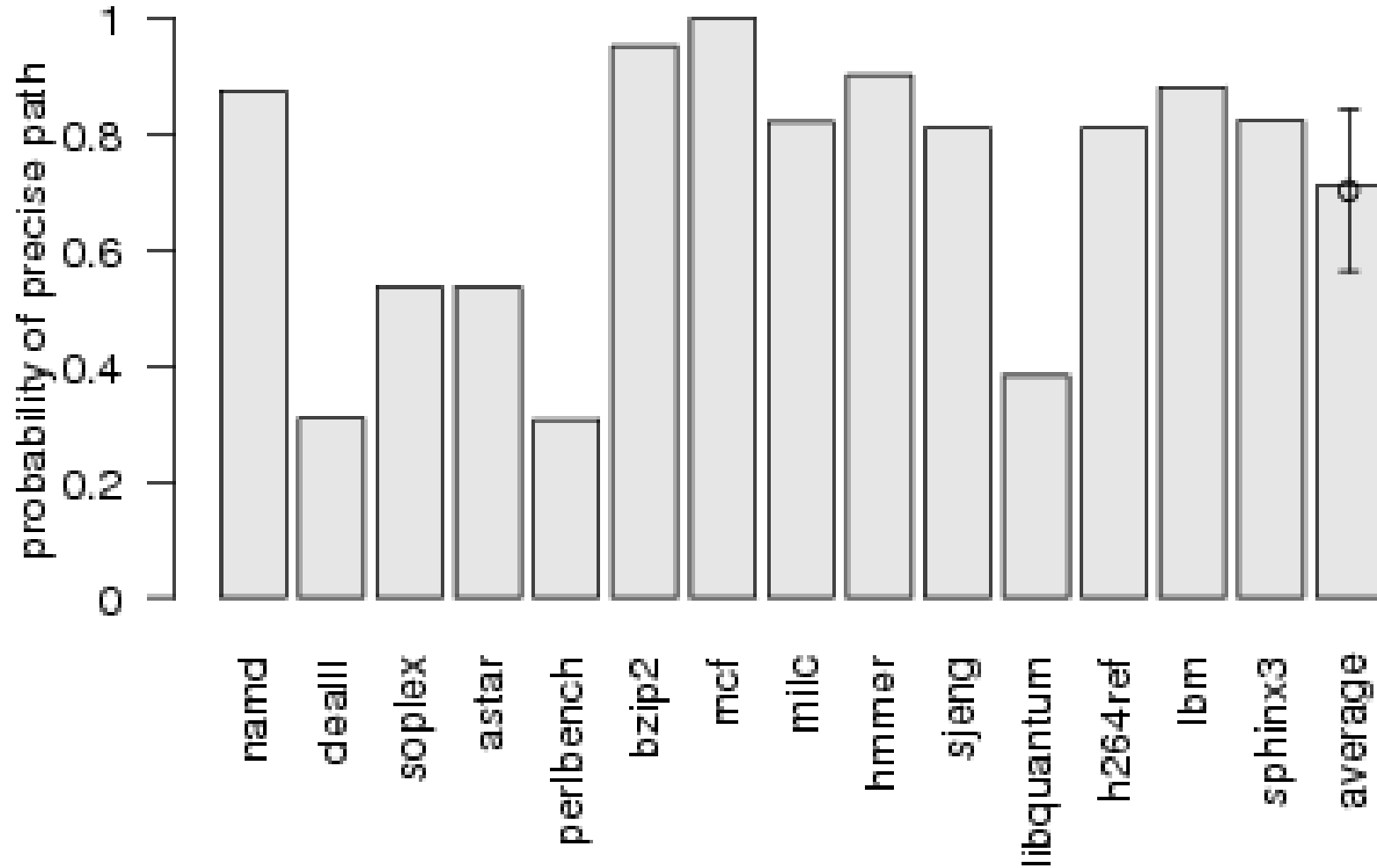
# Problem: *Ambiguity*

More than one call path maps to same PC/SP

stack height



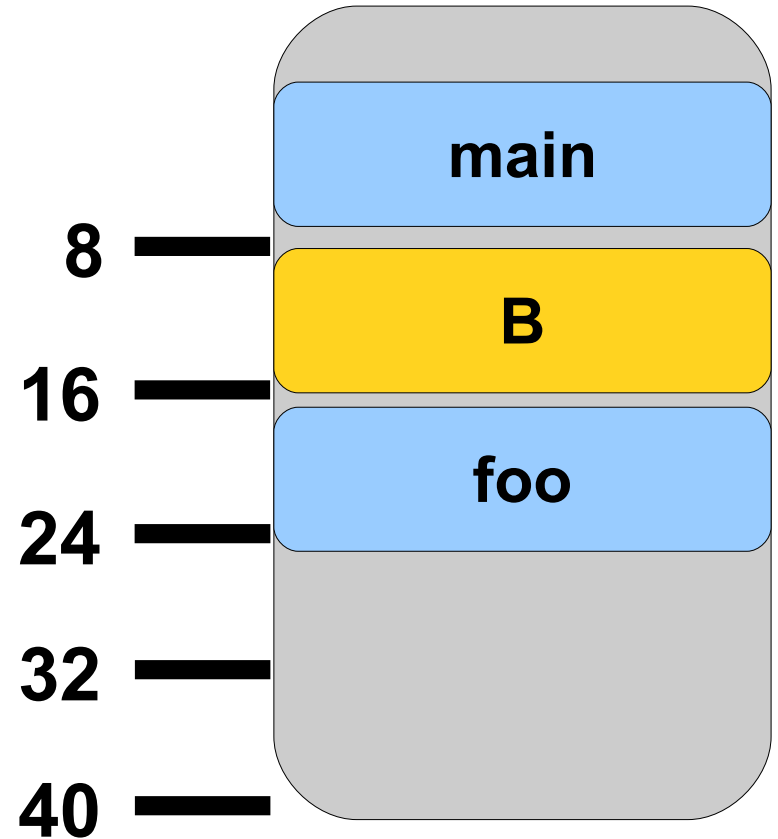
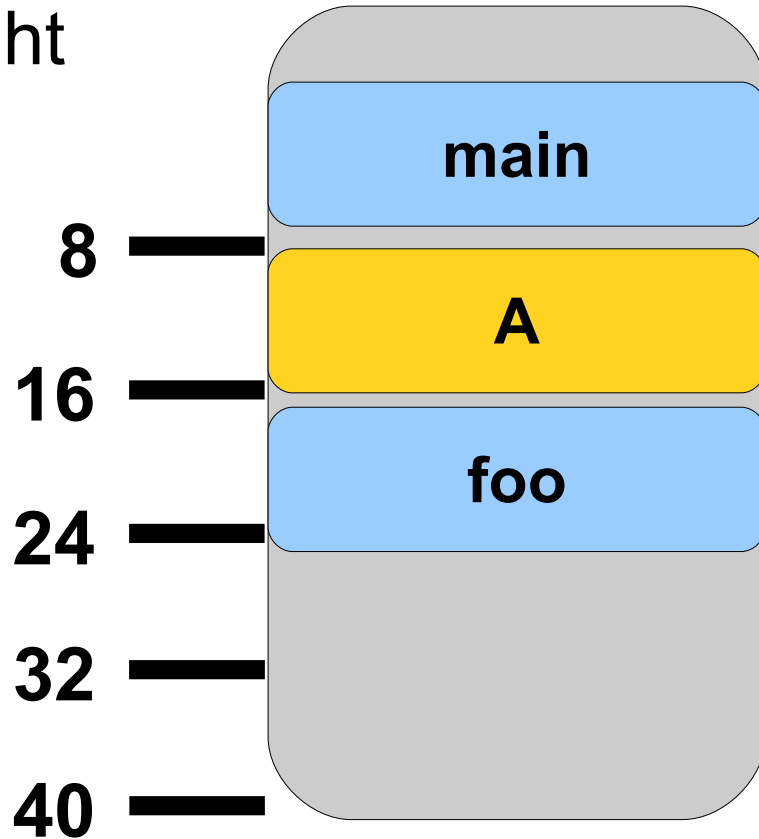
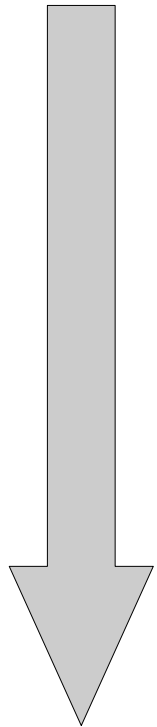
# Is there ambiguity in SPEC?



68% of call paths map to a unique PC/SP

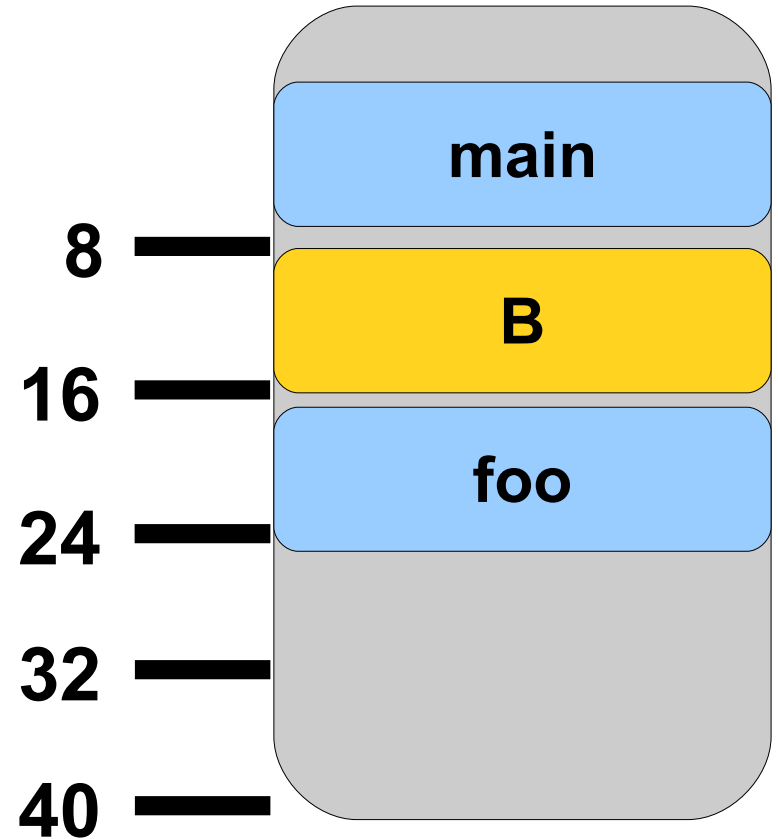
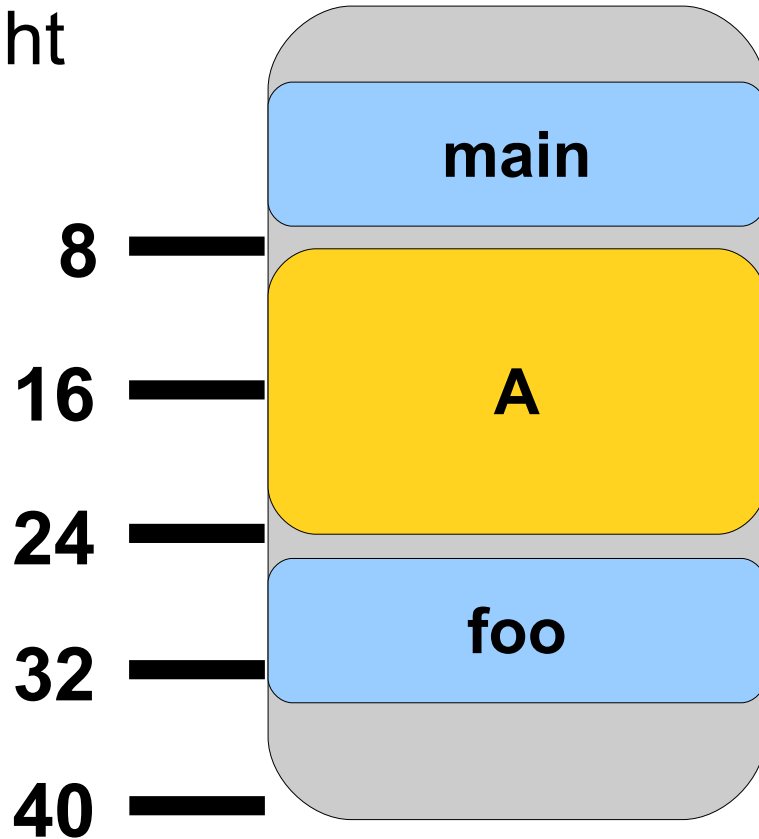
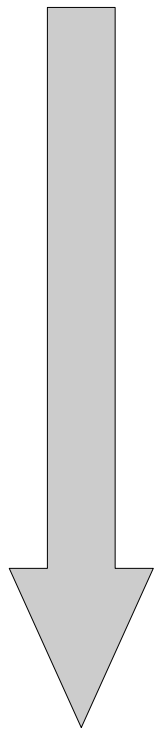
# Activation Record Resizing (ARR)

stack height



# Activation Record Resizing (ARR)

stack height



# Implementing ARR

Before ARR:

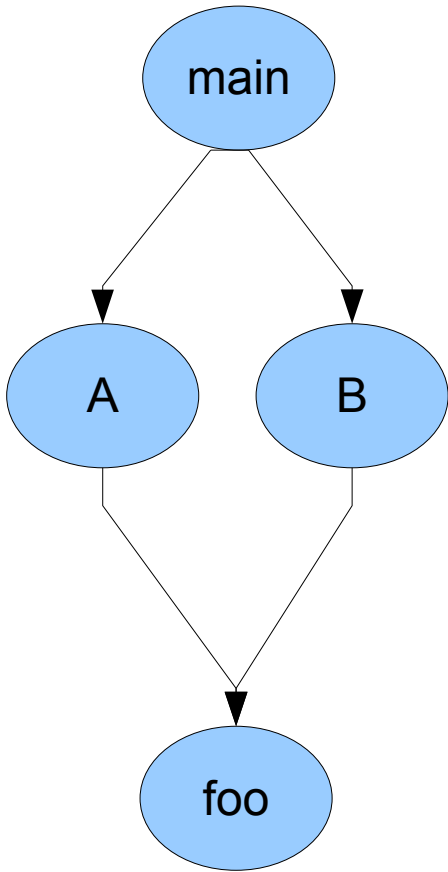
```
<Perl_av_store>:  
402de8:  push %rbp  
402de9:  mov  %rsp, %rbp  
402dec:  sub  $16,  %rsp
```

After ARR:

```
<Perl_av_store>:  
402de8:  push %rbp  
402de9:  mov  %rsp, %rbp  
402dec:  sub  $32,  %rsp
```

Random search to disambiguate binary 16

# Approach



***~~Phase 1: capture calling context~~***

***Phase 2: map calling context to call path***

# Phase 2: mapping calling context to call paths

Calling Context:

PC = foo

SP = 64 bytes

*What is the call path?*

Instrumentation run to build map from PC/SP to call paths

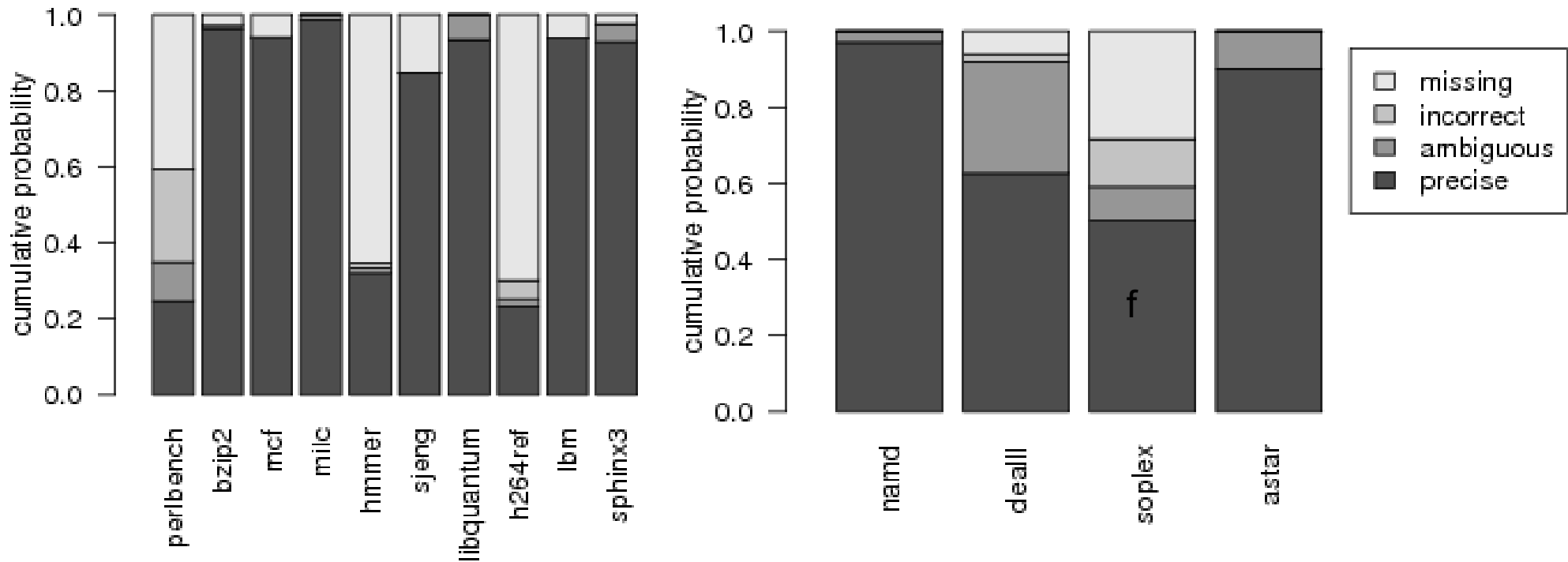
# Experimental Methodology

- Intel ICC compiler on 2.4GHz workstation
- SPEC C/C++ CPU 2006
  - Some have too many paths:
    - xalan, gcc, gobmk
  - One use setjmp/longjmp for co-routines
    - omnetpp

# Experimental Methodology(II)

- Instrumentation on train input
- Evaluate on ref input
  - Does a call path map to a unique PC/SP?
  - Path could be incorrect
  - Path could be missing

# Does a call path map to a unique PC/SP?



C: 75% precise

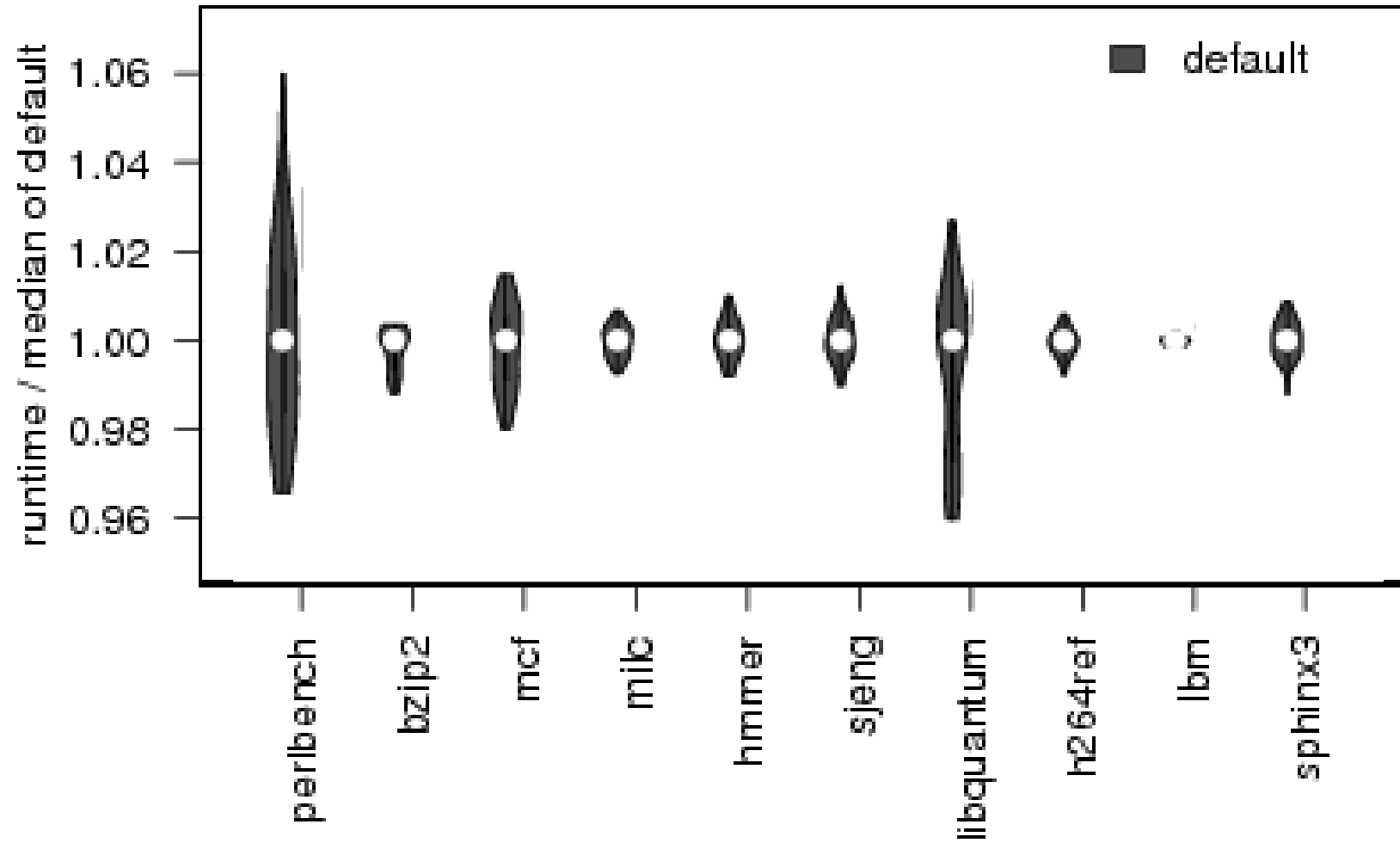
C++: 73% precise<sup>21</sup>

# What about overhead?

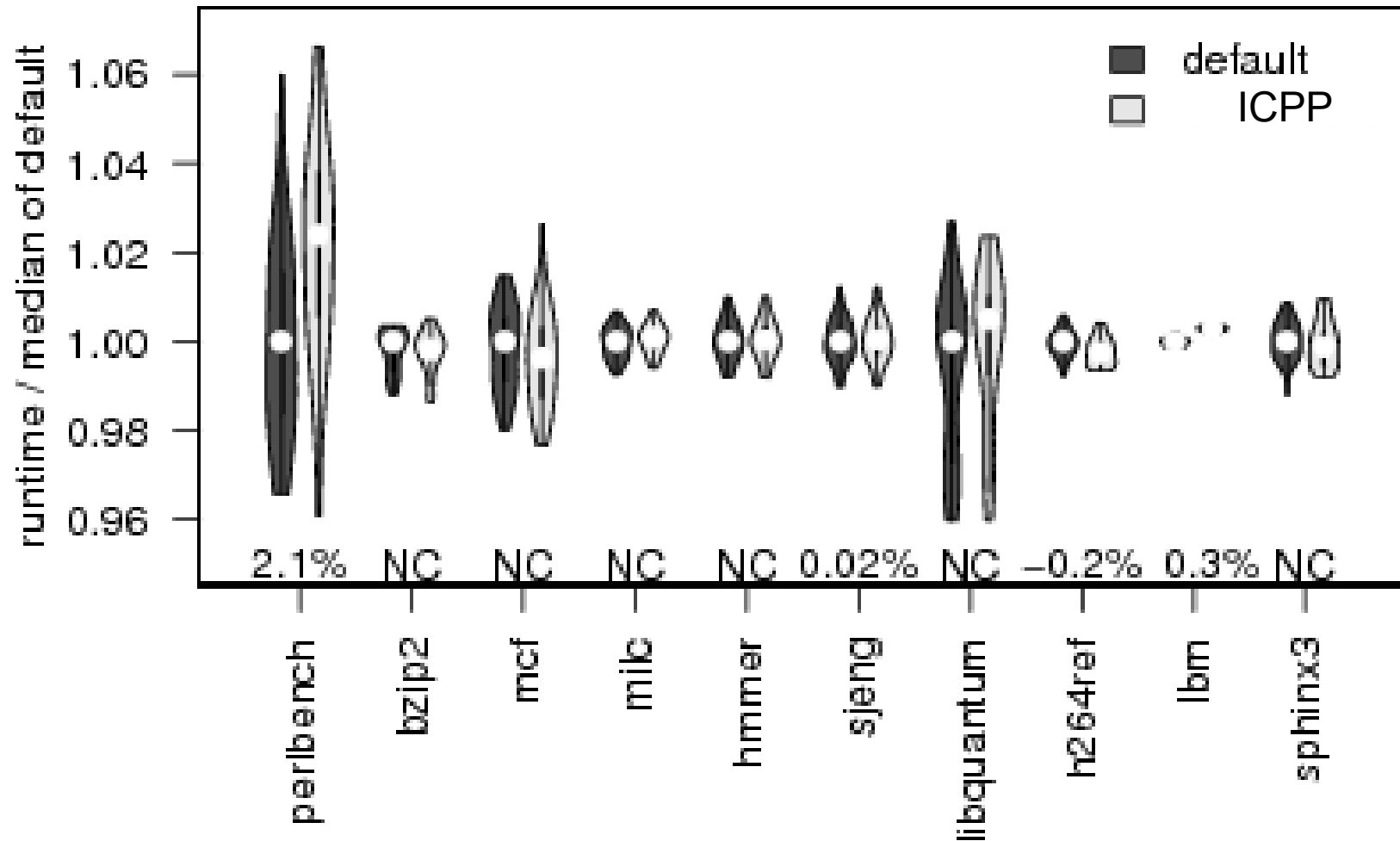
# Where does overhead come from?

- Excess stack utilization
- Hardware sampling PC/SP pairs

# Overhead of ICPP



# Overhead of ICPP



Negligible overhead (geomean = 0.2%) <sup>25</sup>

# Related Work

- Inoue and Nakatani '09
- Exhaustive Instrumentation:
  - [Ball and Larus '94, Graham et al '82, Bond and McKinley '07]
- Selective Instrumentation
  - [Bernat and Miller '07, Zhuang et al '06]
- Stack Walking
  - [Froyd '05]

Amount of Context

High

Low

Low

Cost

High

*ICPP*

Software  
Instrumentation

Hardware  
Counters