Artifacts Available acm V1.1

Artifacts Evaluated acm Reusable V1.1

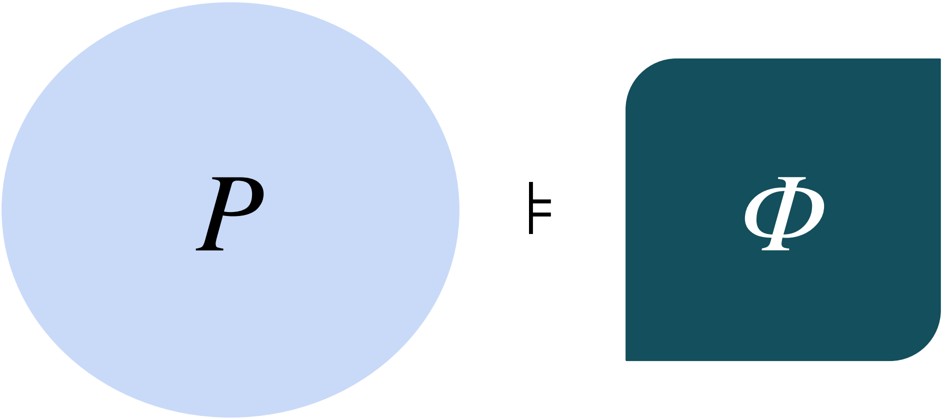
Computing Precise Control Interface Specifications

Eric Hayden Campbell
Hossein Hojjat   Nate Foster

The Unknown

How do we verify programs with unknown code?

A Verification Myth



$P \models \Phi$

Program

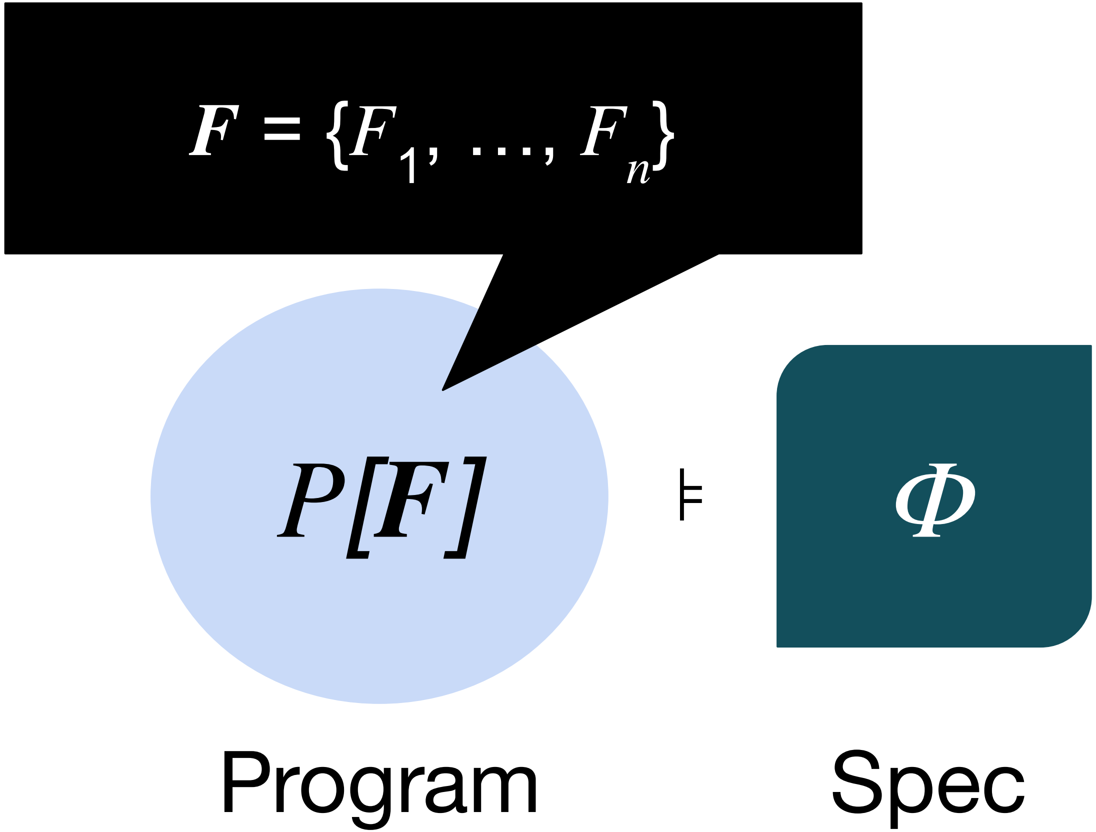
Spec

Source code is incomplete!

Libraries

Modules

System Calls

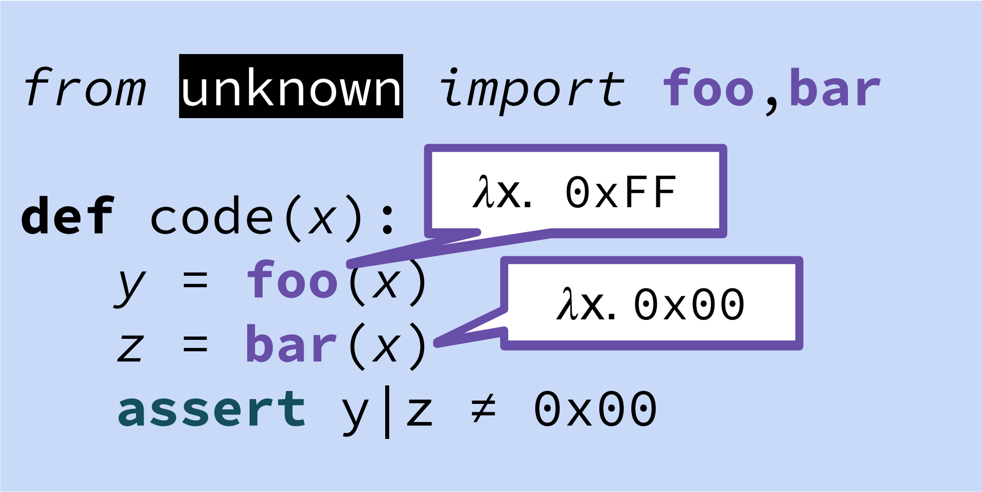


$F = \{F_1, \ldots, F_n\}$

$P[F] \models \Phi$

Program

Spec



```
from unknown import foo,bar

def code(x):
    y = foo(x)
    z = bar(x)
    assert y|z ≠ 0x00
```

λx. 0xFF

λx. 0x00

Which implementations?

```
import unknown as foo,bar

def code(x):
    y = foo(x)
    z = bar(x)
    assert y|z ≠ 0x00
```

Which implementations satisfy the spec?

Independent Specs
[POPL ‘16, SIGCOMM ‘20]

φ(**foo**) ∧ ψ(**bar**)

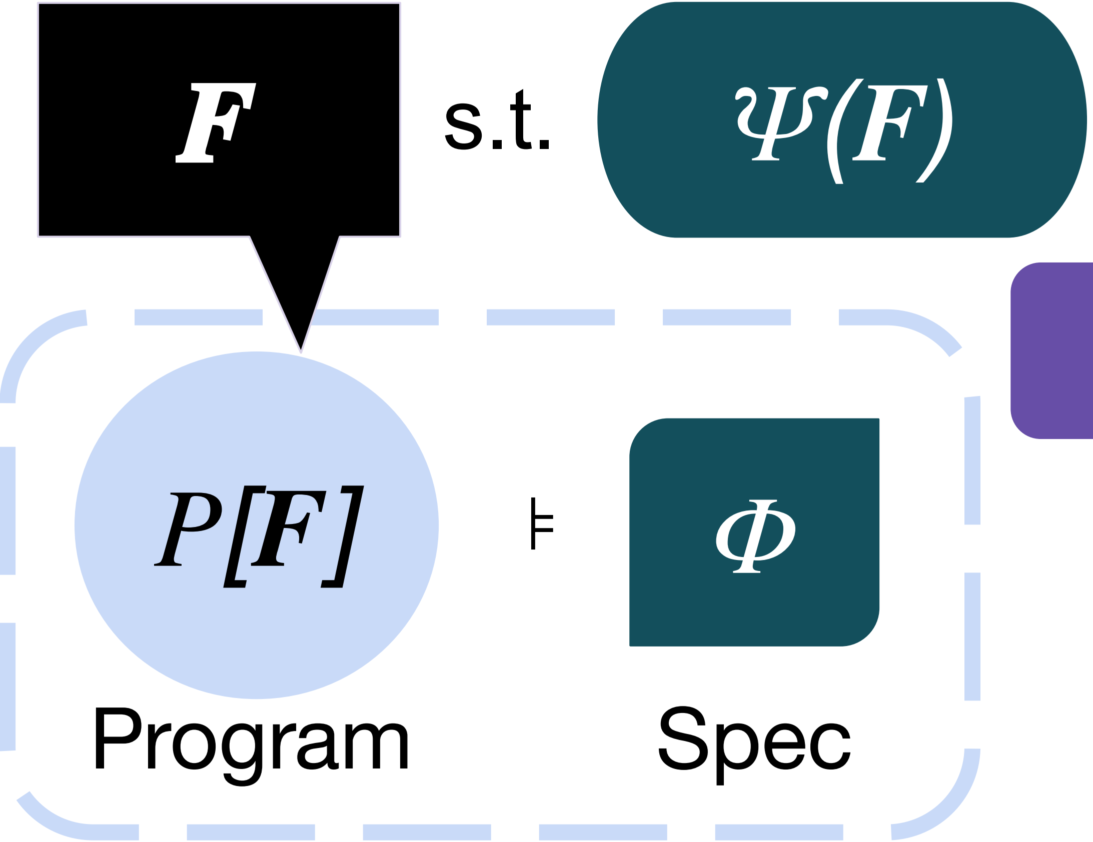
Necessary [SIGCOMM ‘20]

`true`

Not Safe!!!

Eliminates
no “good runs”
[VMCAI ‘13]

Safe	Unsafe!
Overly Restrictive!	Permissive
[POPL ‘16]	[SIGCOMM ‘20]



$\boldsymbol{F}$ s.t. $\Psi(F)$

$P[F] \models \Phi$

Program

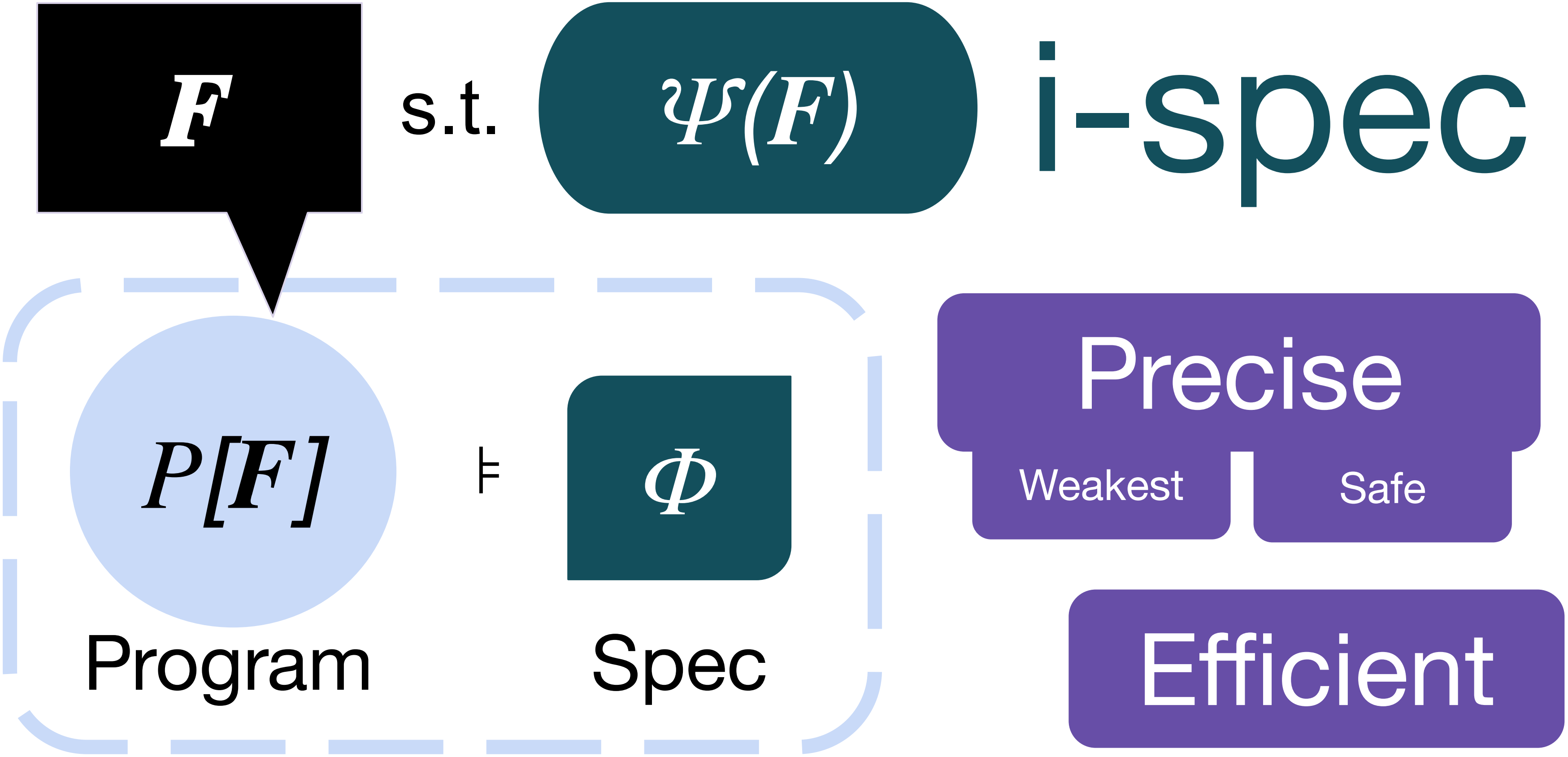
Spec

i-spec

Safe

Weakest

Efficient



F s.t. $\Psi(F)$

i-spec

$P[F] \models \Phi$

Program

Spec

Precise

Weakest

Safe

Efficient

```
import unknown as foo,bar

def code(x):
   y = foo(x)
   z = bar(x)
   assert y|z ≠ 0x00
```

Goal:
Compute
precise i-specs

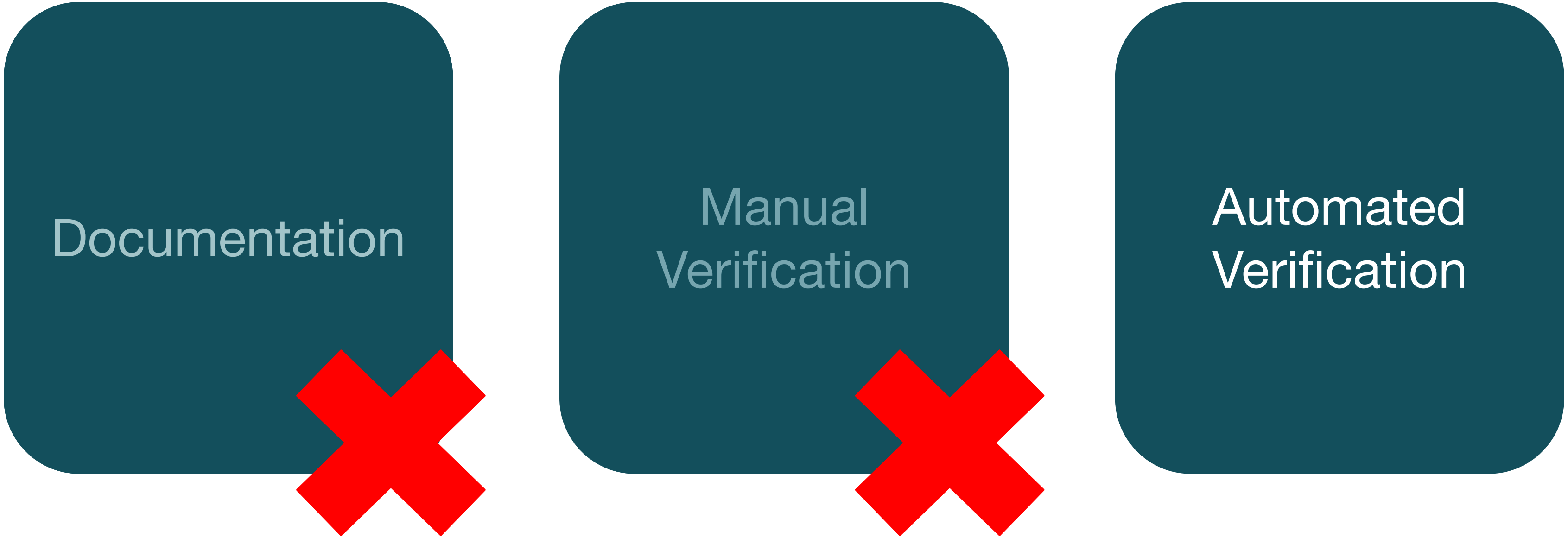
```
foo(x) | bar(x) ≠ 0x00
```

Safe

Weakest

Efficient

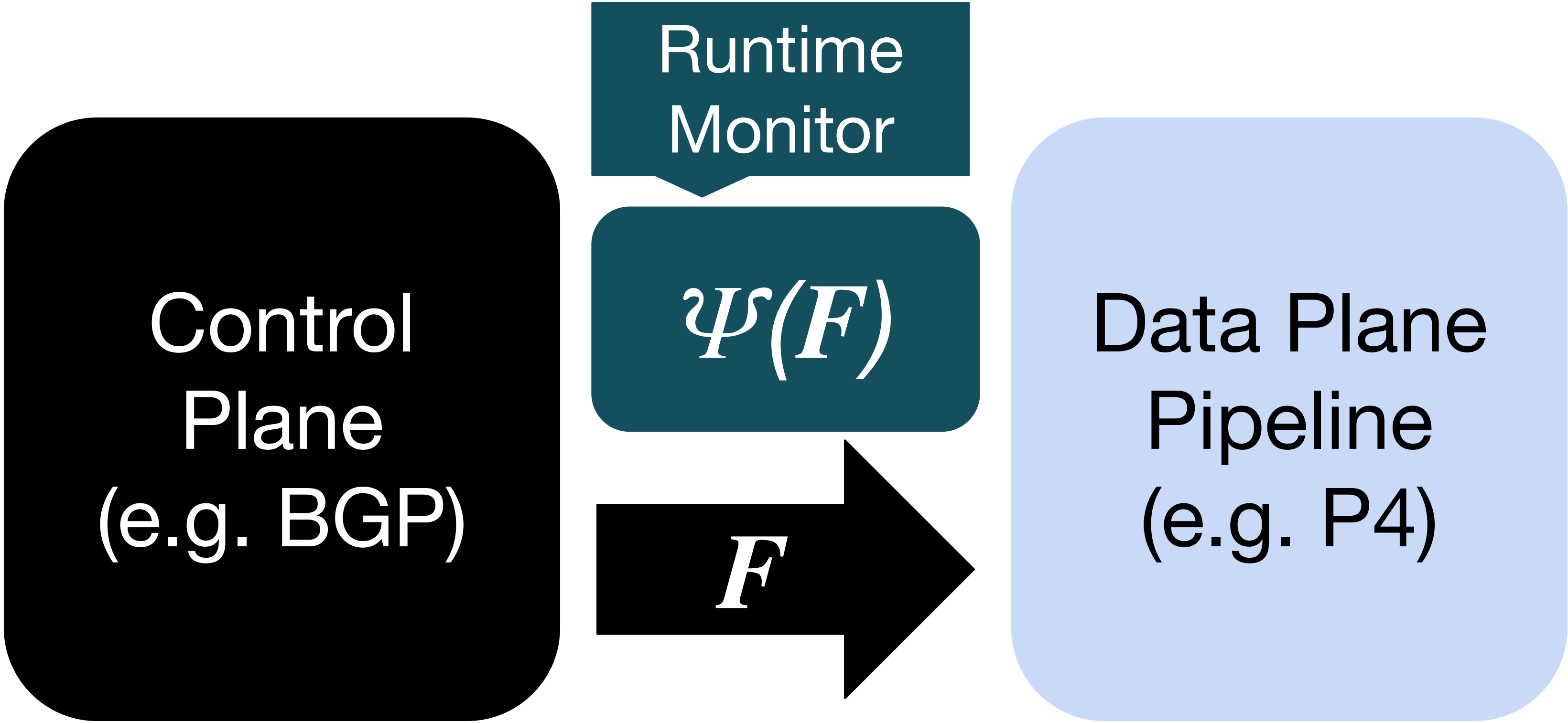
How to use computed i-specs?



Documentation

Manual Verification

Automated Verification



Control Plane (e.g. BGP)

Runtime Monitor

$\Psi(F)$

F

Data Plane Pipeline (e.g. P4)

Computer Network

Capisce

computes control interface specs (ci-specs)

Precise

Safe

Weakest

Efficient

Step 1:

Model Pipeline Programs

GCL(F)

?

```
c ::= assume φ
    | x := e
    | c ; c
    | c [] c
```

$F : 2^w \to 2^l$

e ::= … | F(e)

$\varphi \in$ QFBV

Control Flow in GCL(F)

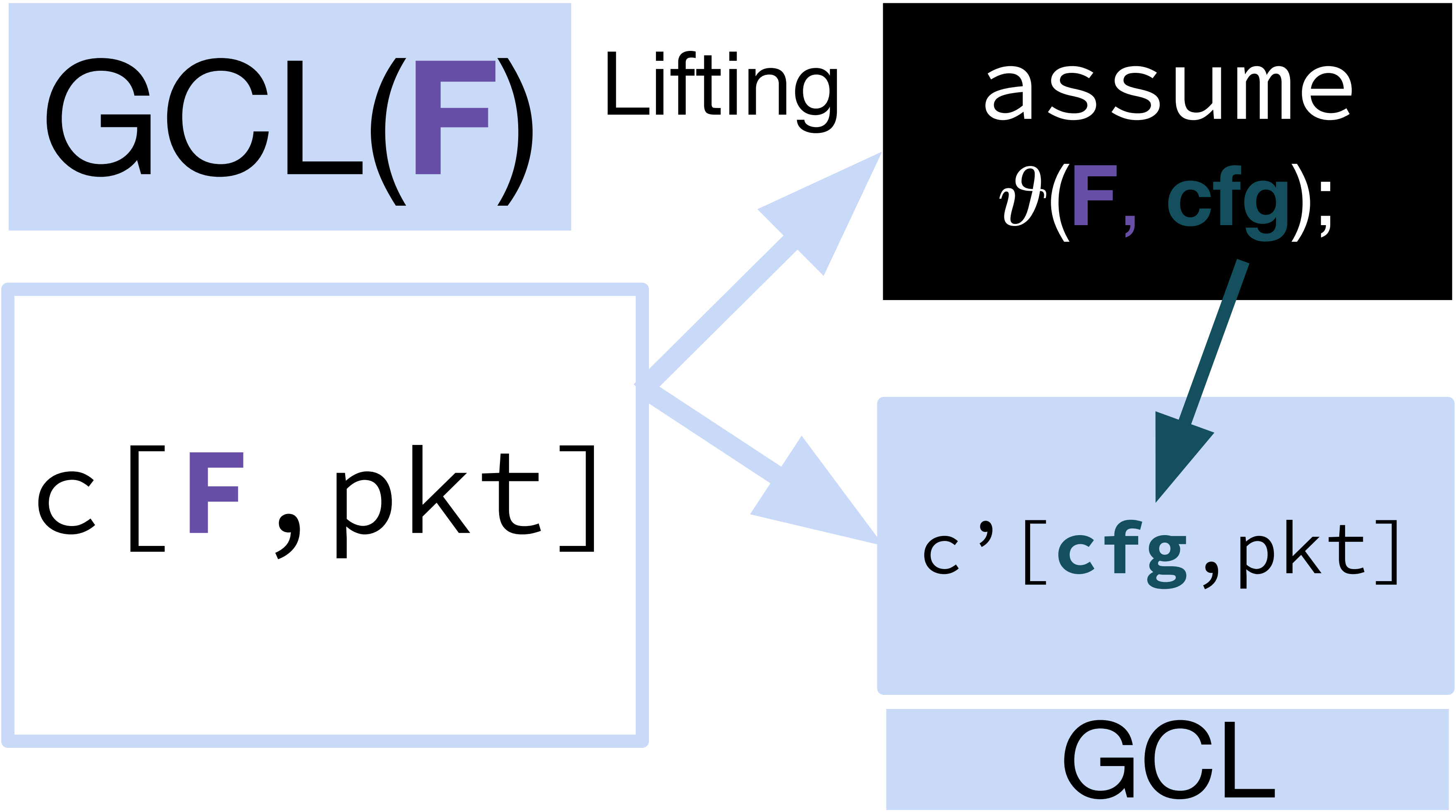
$\texttt{if}\ \varphi\ \texttt{c}_1\ \texttt{c}_2$

$\texttt{assume}\ \ \varphi;\texttt{c}_1$

$\texttt{[]}$

$\texttt{assume}\ \neg\varphi;\texttt{c}_2$

Step 2: Symbolic Compilation



GCL(F)

Lifting

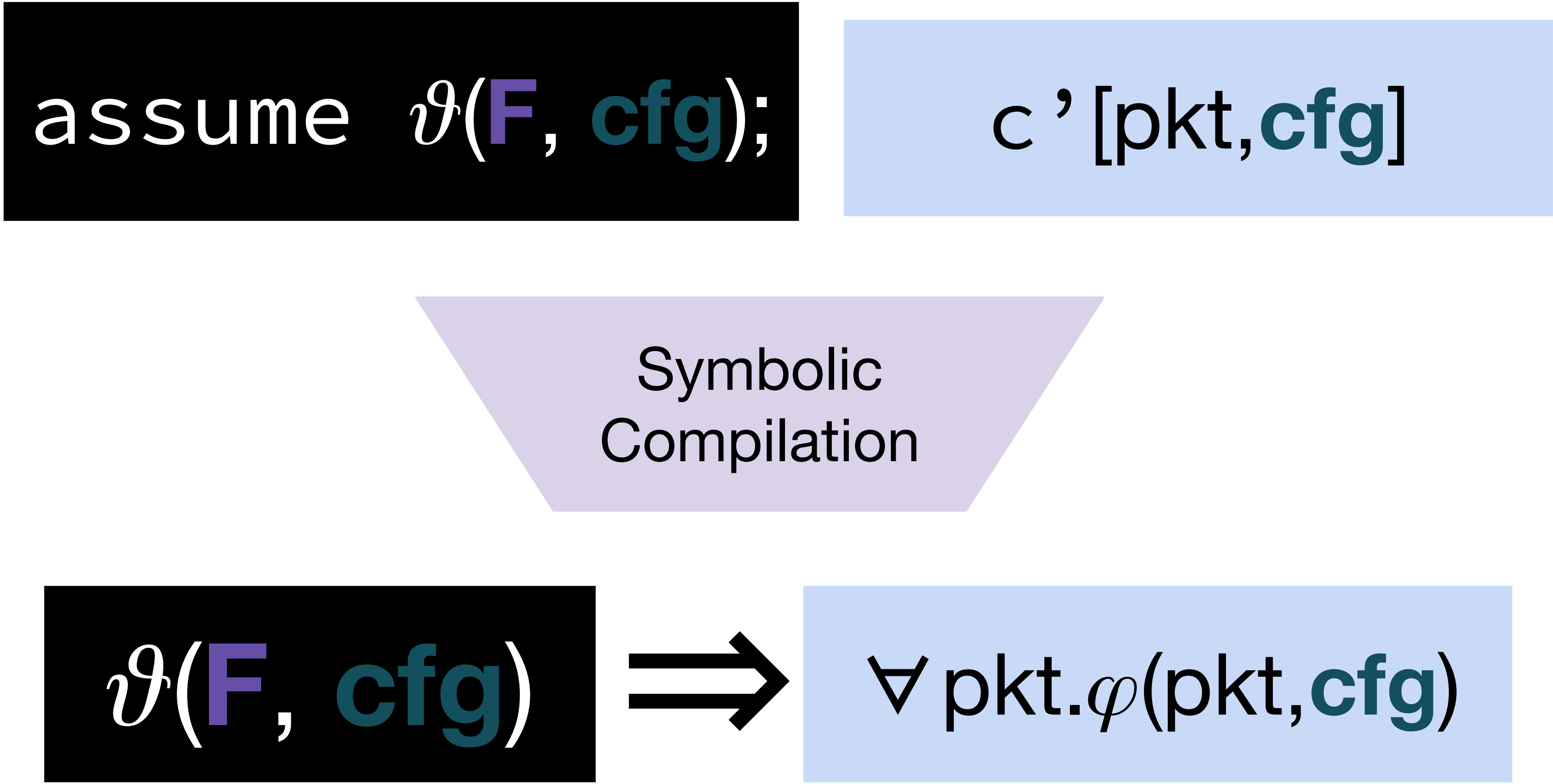
assume

ϑ(F, **cfg**);

`c[F,pkt]`

`c’[cfg,pkt]`

GCL



```
assume ϑ(F, cfg);
```

c’[pkt,cfg]

Symbolic Compilation

$$\vartheta(\mathbf{F}, \mathbf{cfg}) \Rightarrow \forall\, \text{pkt}.\varphi(\text{pkt}, \mathbf{cfg})$$

ci-spec

$$\vartheta(F, \mathrm{cfg}) \Rightarrow \forall\, \mathrm{pkt}.\varphi(\mathrm{pkt}, \mathrm{cfg})$$

Precise

pkt is big

~1500 bits

¬Efficient to monitor

Step 3: Quantifier Elimination

$$\begin{tikzcd}
\vartheta(\mathsf{F}, \mathsf{cfg}) \arrow[r, Rightarrow] & \forall\, \mathsf{pkt}.\varphi(\mathsf{pkt}, \mathsf{cfg}) \arrow[d, "\mathsf{QE}"] \\
\vartheta(\mathsf{F}, \mathsf{cfg}) \arrow[r, Rightarrow] & \psi(\mathsf{cfg})
\end{tikzcd}$$

ci-spec

$$\vartheta(F, y) \Rightarrow \psi(y)$$

Theorem. *Precise*
=> *safe*
=> *weakest*

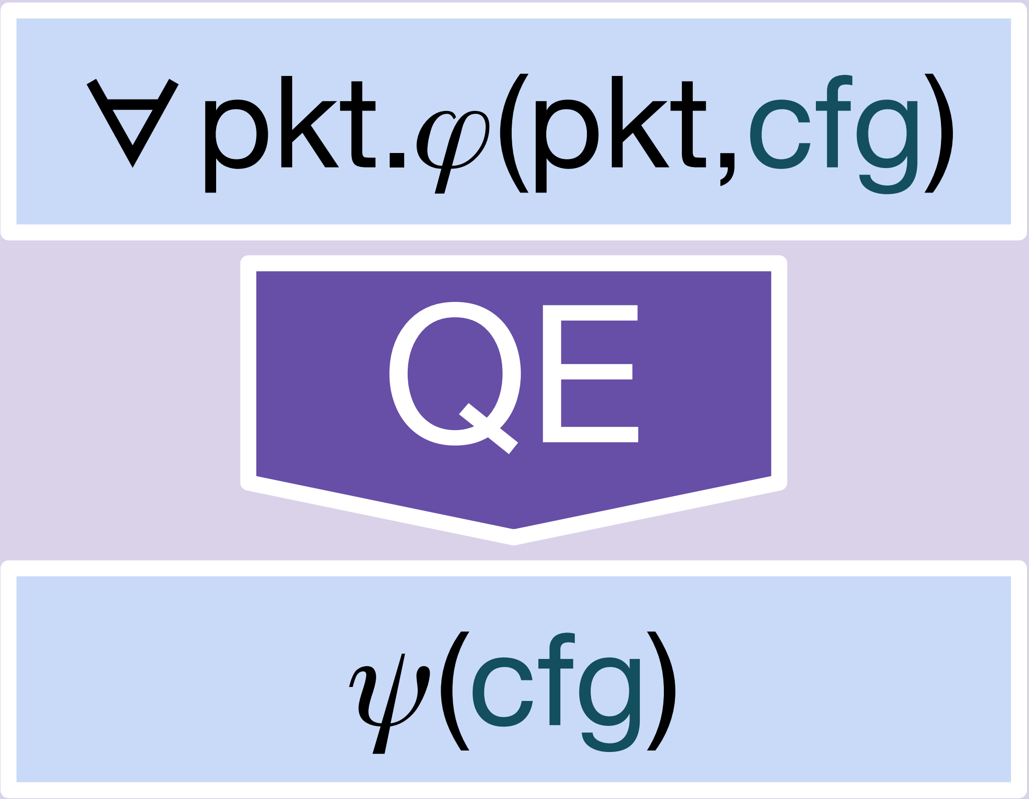
Theorem. *Terminates*

Theorem. *Efficiently monitorable*

Efficiently Control-Monitorable Sentences

… have polynomial expression complexity

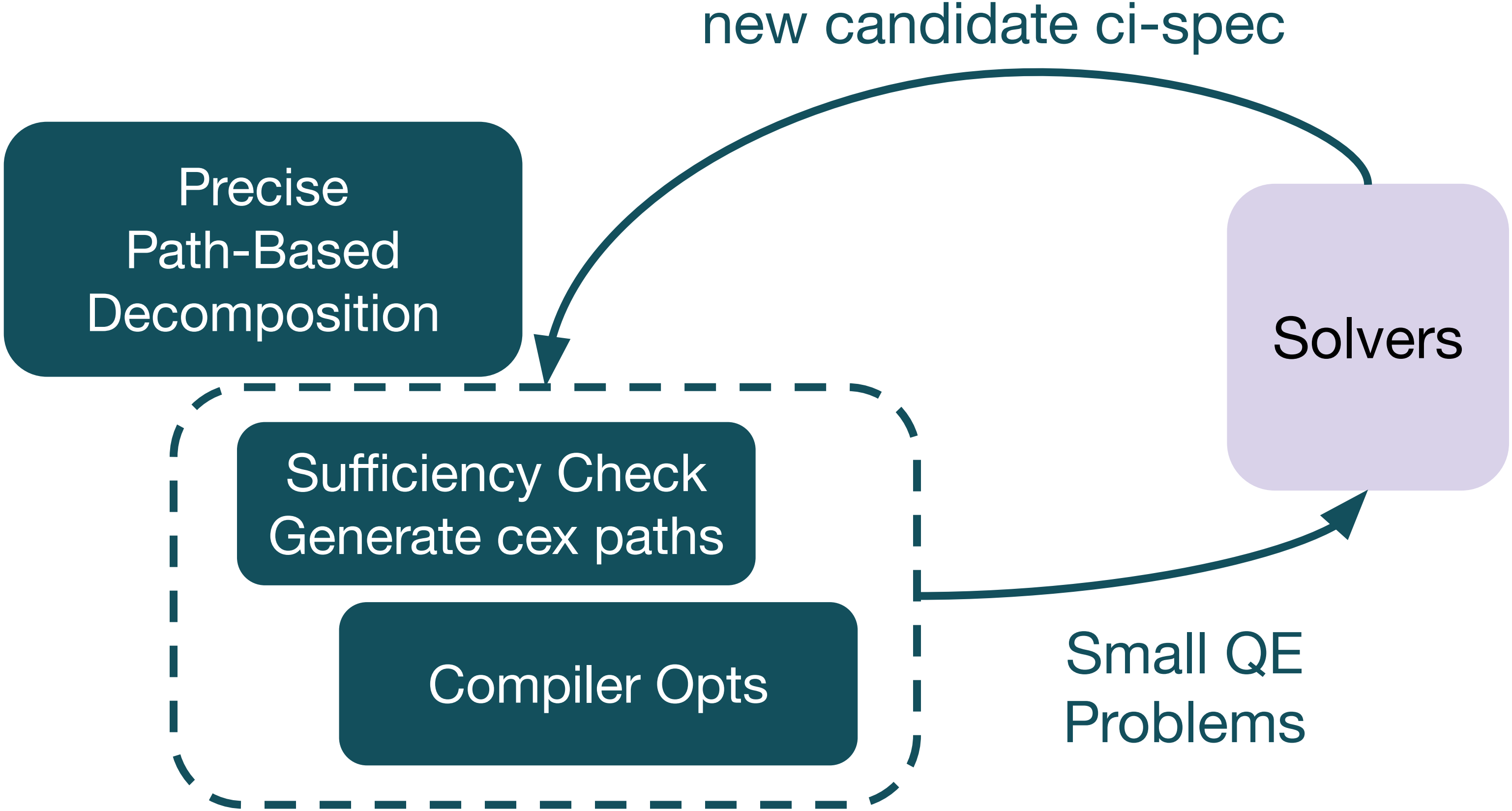
Quantifier Elimination is *Intractable*



$\forall\, \mathrm{pkt}.\varphi(\mathrm{pkt},\mathrm{cfg})$

QE

$\psi(\mathrm{cfg})$



new candidate ci-spec

Precise Path-Based Decomposition

Solvers

Sufficiency Check Generate cex paths

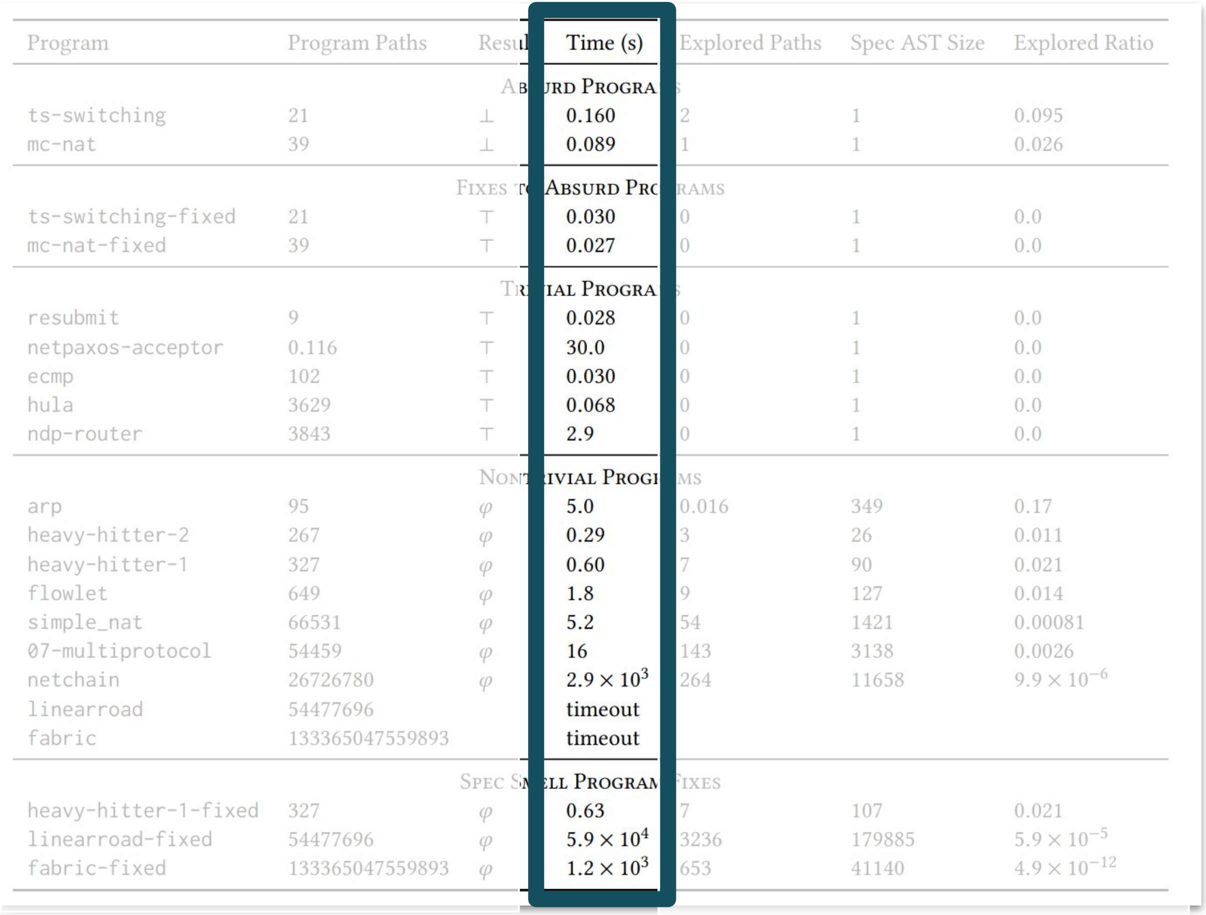
Compiler Opts

Small QE Problems

Evaluation

Survey of Industrial and Academic P4 Programs

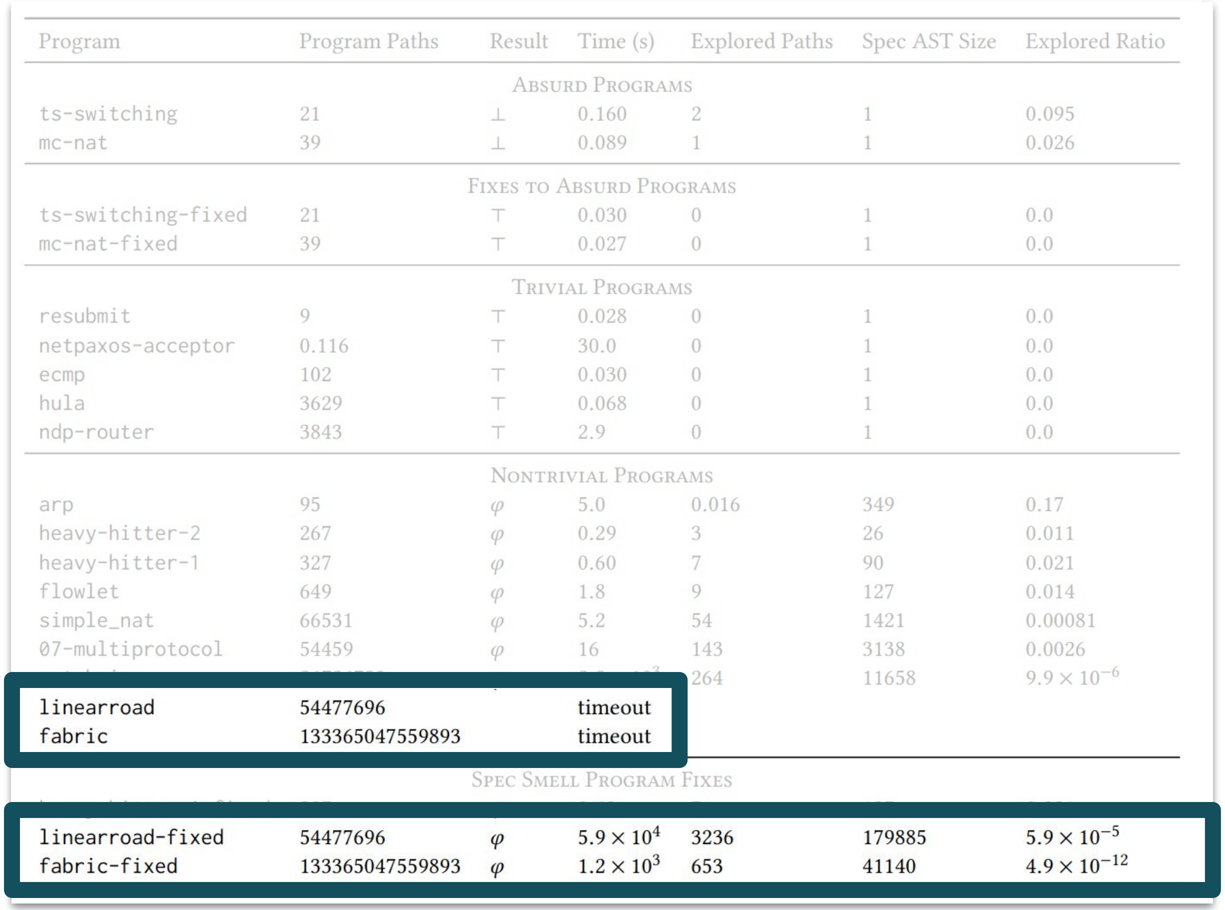
Ensure Invalid Data is not Read



Program	Program Paths	Result	Time (s)	Explored Paths	Spec AST Size	Explored Ratio
ABSURD PROGRAMS						
ts-switching	21	⊥	0.160	2	1	0.095
mc-nat	39	⊥	0.089	1	1	0.026
FIXES TO ABSURD PROGRAMS						
ts-switching-fixed	21	⊤	0.030	0	1	0.0
mc-nat-fixed	39	⊤	0.027	0	1	0.0
TRIVIAL PROGRAMS						
resubmit	9	⊤	0.028	0	1	0.0
netpaxos-acceptor	0.116	⊤	30.0	0	1	0.0
ecmp	102	⊤	0.030	0	1	0.0
hula	3629	⊤	0.068	0	1	0.0
ndp-router	3843	⊤	2.9	0	1	0.0
NONTRIVIAL PROGRAMS						
arp	95	φ	5.0	0.016	349	0.17
heavy-hitter-2	267	φ	0.29	3	26	0.011
heavy-hitter-1	327	φ	0.60	7	90	0.021
flowlet	649	φ	1.8	9	127	0.014
simple_nat	66531	φ	5.2	54	1421	0.00081
07-multiprotocol	54459	φ	16	143	3138	0.0026
netchain	26726780	φ	2.9×10^3	264	11658	9.9×10^{-6}
linearroad	54477696		timeout			
fabric	133365047559893		timeout			
SPEC SMALL PROGRAM FIXES						
heavy-hitter-1-fixed	327	φ	0.63	7	107	0.021
linearroad-fixed	54477696	φ	5.9×10^4	3236	179885	5.9×10^{-5}
fabric-fixed	133365047559893	φ	1.2×10^3	653	41140	4.9×10^{-12}

Program Survey

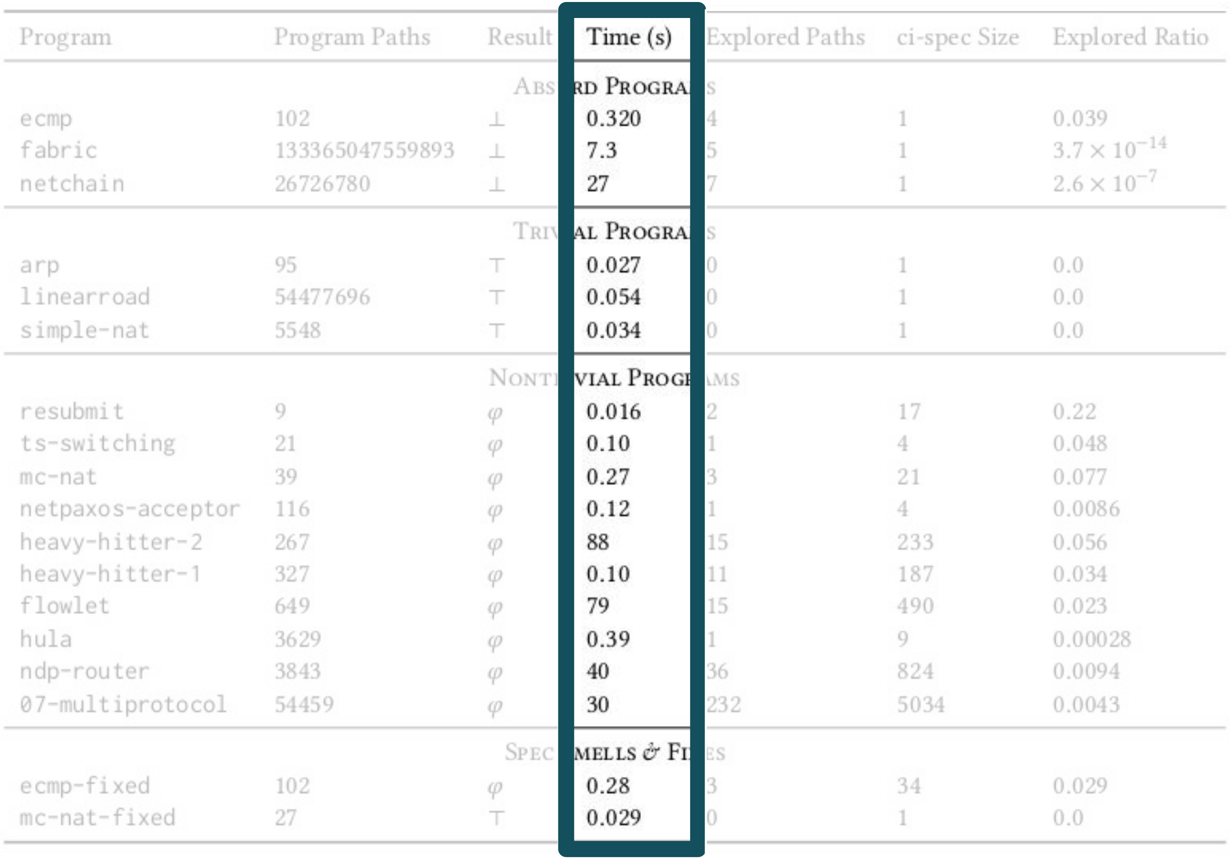
Ensure Invalid Data is not Read



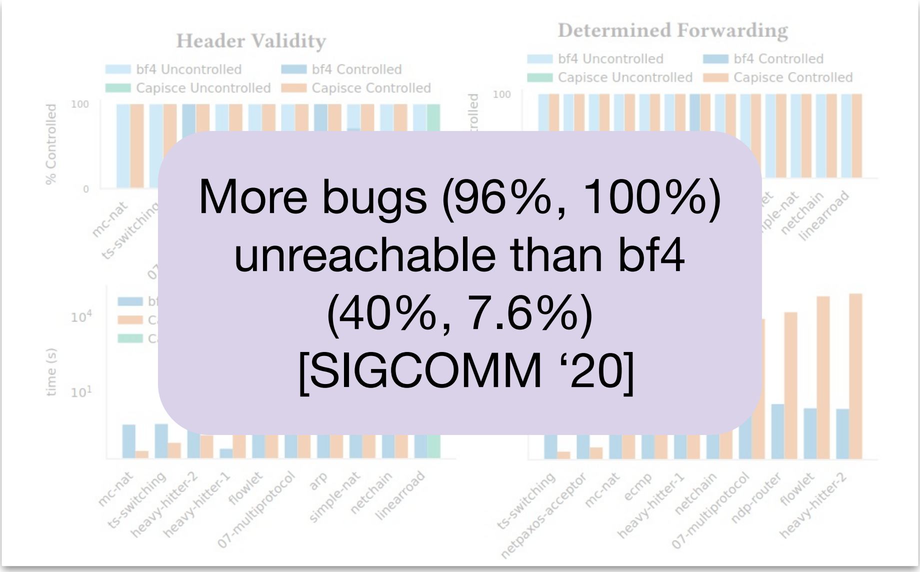
Program	Program Paths	Result	Time (s)	Explored Paths	Spec AST Size	Explored Ratio
			Absurd Programs			
ts-switching	21	⊥	0.160	2	1	0.095
mc-nat	39	⊥	0.089	1	1	0.026
			Fixes to Absurd Programs			
ts-switching-fixed	21	⊤	0.030	0	1	0.0
mc-nat-fixed	39	⊤	0.027	0	1	0.0
			Trivial Programs			
resubmit	9	⊤	0.028	0	1	0.0
netpaxos-acceptor	0.116	⊤	30.0	0	1	0.0
ecmp	102	⊤	0.030	0	1	0.0
hula	3629	⊤	0.068	0	1	0.0
ndp-router	3843	⊤	2.9	0	1	0.0
			Nontrivial Programs			
arp	95	φ	5.0	0.016	349	0.17
heavy-hitter-2	267	φ	0.29	3	26	0.011
heavy-hitter-1	327	φ	0.60	7	90	0.021
flowlet	649	φ	1.8	9	127	0.014
simple_nat	66531	φ	5.2	54	1421	0.00081
07-multiprotocol	54459	φ	16	143	3138	0.0026
[illegible]	[illegible]	[illegible]	[illegible]	264	11658	9.9×10^{-6}
linearroad	54477696		timeout			
fabric	133365047559893		timeout			
			Spec Smell Program Fixes			
linearroad-fixed	54477696	φ	5.9×10^4	3236	179885	5.9×10^{-5}
fabric-fixed	133365047559893	φ	1.2×10^3	653	41140	4.9×10^{-12}

Program Survey

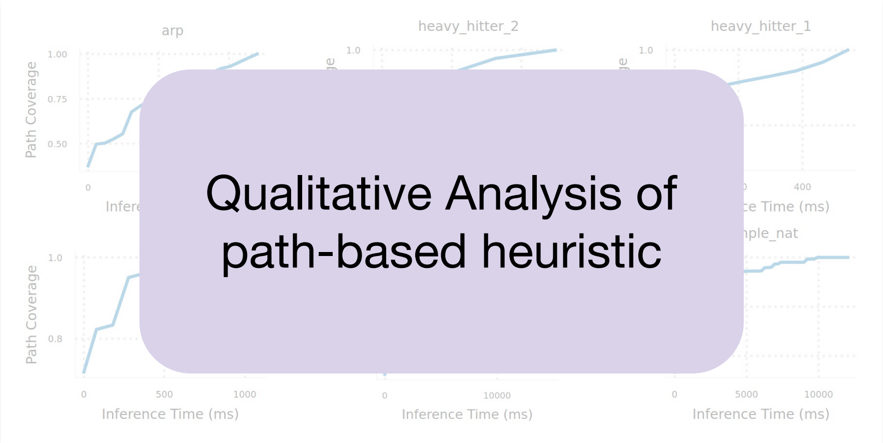
Defined Forwarding



Program	Program Paths	Result	Time (s)	Explored Paths	ci-spec Size	Explored Ratio
			Absurd Programs			
ecmp	102	⊥	0.320	4	1	0.039
fabric	133365047559893	⊥	7.3	5	1	3.7×10^{-14}
netchain	26726780	⊥	27	7	1	2.6×10^{-7}
			Trivial Programs			
arp	95	⊤	0.027	0	1	0.0
linearroad	54477696	⊤	0.054	0	1	0.0
simple-nat	5548	⊤	0.034	0	1	0.0
			Nontrivial Programs			
resubmit	9	φ	0.016	2	17	0.22
ts-switching	21	φ	0.10	1	4	0.048
mc-nat	39	φ	0.27	3	21	0.077
netpaxos-acceptor	116	φ	0.12	1	4	0.0086
heavy-hitter-2	267	φ	88	15	233	0.056
heavy-hitter-1	327	φ	0.10	11	187	0.034
flowlet	649	φ	79	15	490	0.023
hula	3629	φ	0.39	1	9	0.00028
ndp-router	3843	φ	40	36	824	0.0094
07-multiprotocol	54459	φ	30	232	5034	0.0043
			Spec Smells & Fixes			
ecmp-fixed	102	φ	0.28	3	34	0.029
mc-nat-fixed	27	⊤	0.029	0	1	0.0

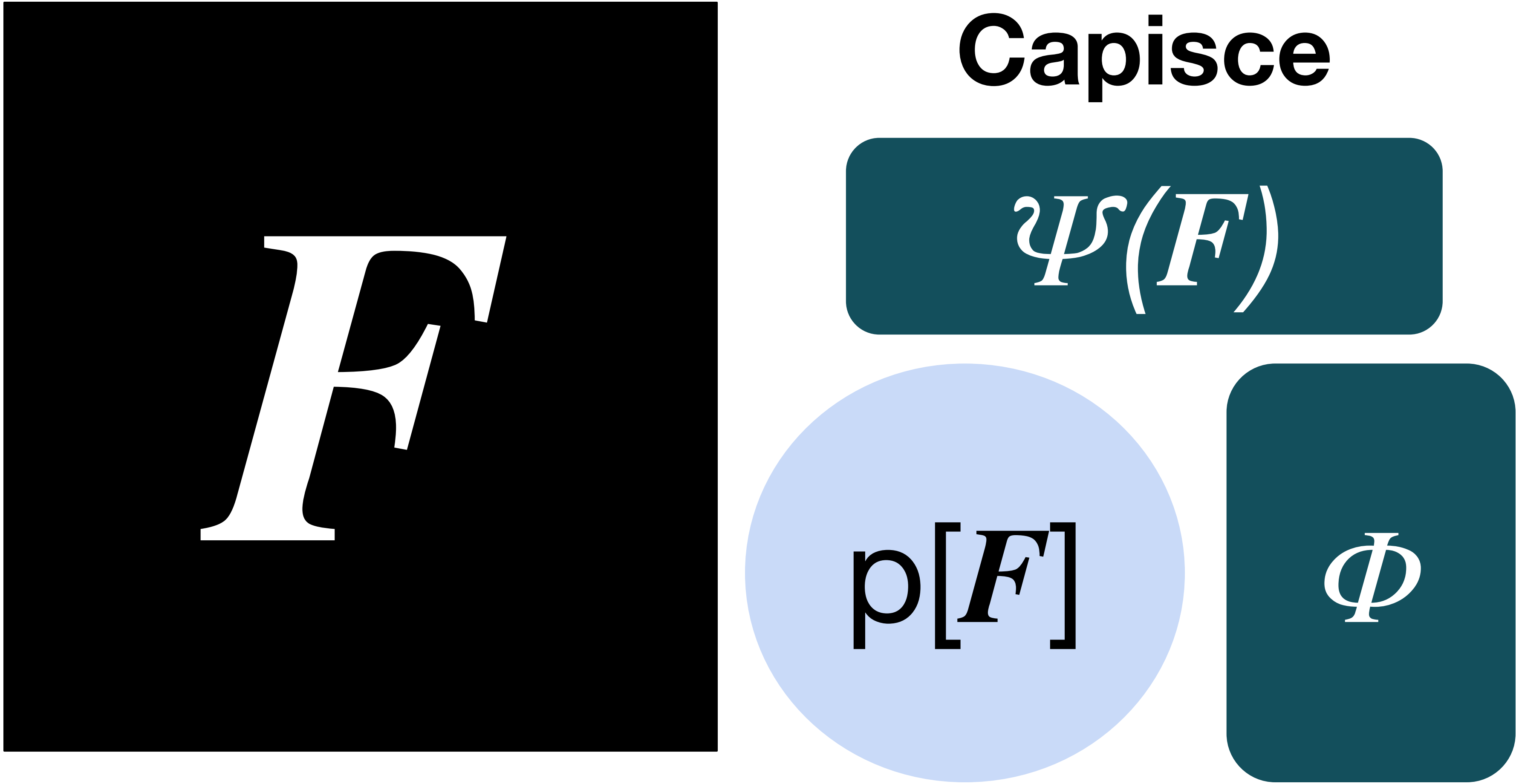


More bugs (96%, 100%) unreachable than bf4 (40%, 7.6%) [SIGCOMM '20]



Qualitative Analysis of path-based heuristic

Capisce



F

$\Psi(F)$

$\mathrm{p}[\boldsymbol{F}]$

Φ