Conditional Testing

Off-the-Shelf Combination of Test-Case Generators

Thomas Lemberger

Joint work with Dirk Beyer

Published at ATVA'19:

https://tinyurl.com/c0ndtest

LMU Munich, Germany

CoVeriTest
AFL-fuzz
Symbiotic
CREST
KLEE
Verifuzz
EXE
DART
FShell
Fair-fuzz
AFL-fast
PRTest
CPA/Tiger-MGP

- Automated test generation is at its peak
- But:
  - Different strengths and weaknesses
  - Every generator working on its own
  - Proprietary interfaces
- Lock-in effect

KLEE
vs
AFL-fuzz

- Automated test generation is at its peak
- But:
  - Different strengths and weaknesses
  - Every generator working on its own
  - Proprietary interfaces
- Lock-in effect

KLEE
+
AFL-fuzz

```
int i = input();

if(i != 1017) {
  while(i > 1017) {
    // branch 1.1
    i--;
  }
  // branch 1.2
} else {
  // branch 2
  // ...
}
```

- Random generation: doesn't find i = 1017
- Symbolic execution with DFS: stuck in while-loop

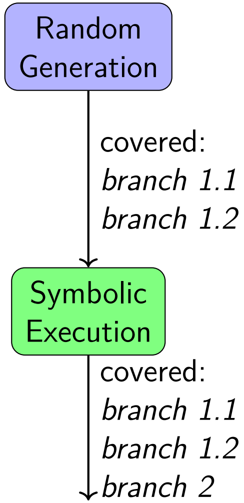
```
int i = input();

if(i != 1017) {
  while(i > 1017) {
    // branch 1.1
    i--;
  }
  // branch 1.2
} else {
  // branch 2
  // ...
}
```

Random Generation

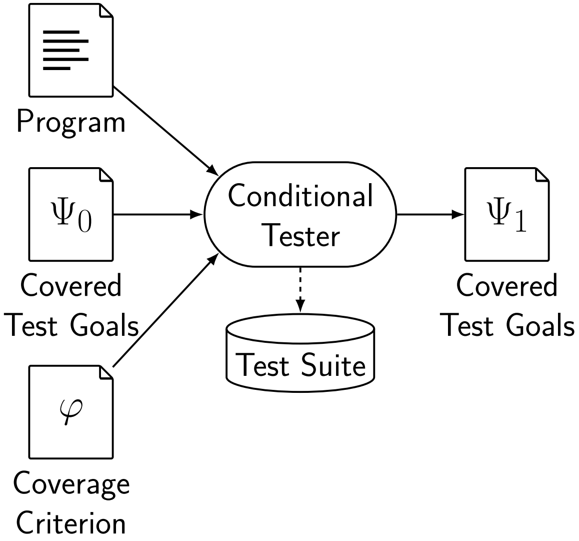
```
int i = input();

if (i != 1017) {




} else {
  // branch 2
  // ...
}
```

Symbolic Execution

```
int i = input();

if(i != 1017) {
  while(i > 1017) {
    // branch 1.1
    i--;
  }
  // branch 1.2
} else {
  // branch 2
  // ...
}
```

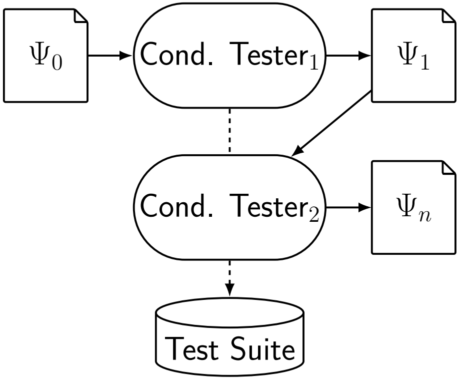


Conditional Tester



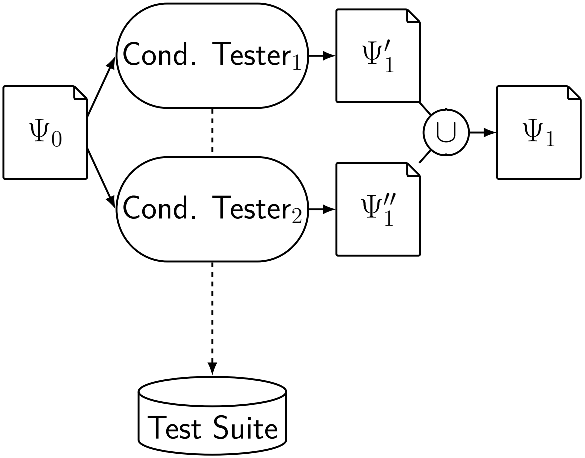
Combinations

- Sequential



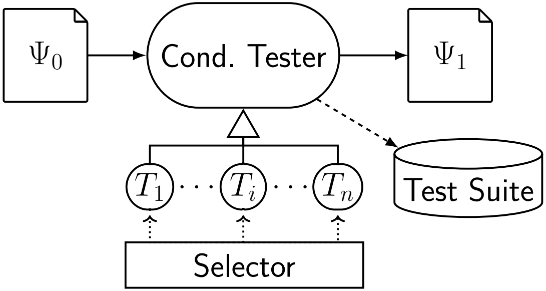
Combinations

- Sequential
- Portfolio



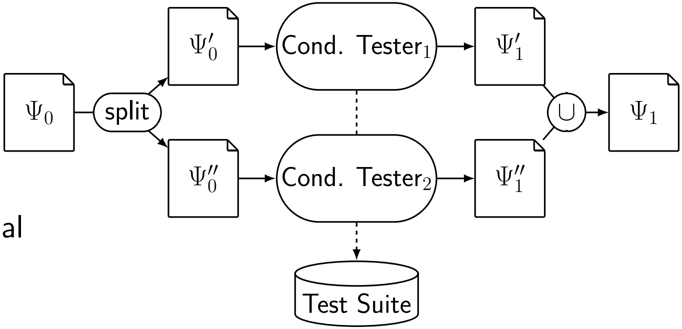
Combinations

- Sequential
- Portfolio
- Strategy selection



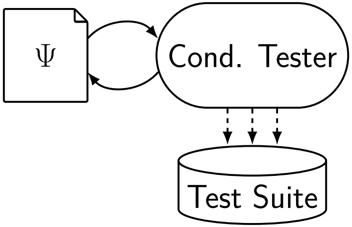
Combinations

- Sequential
- Portfolio
- Strategy selection
- Compositional



Combinations

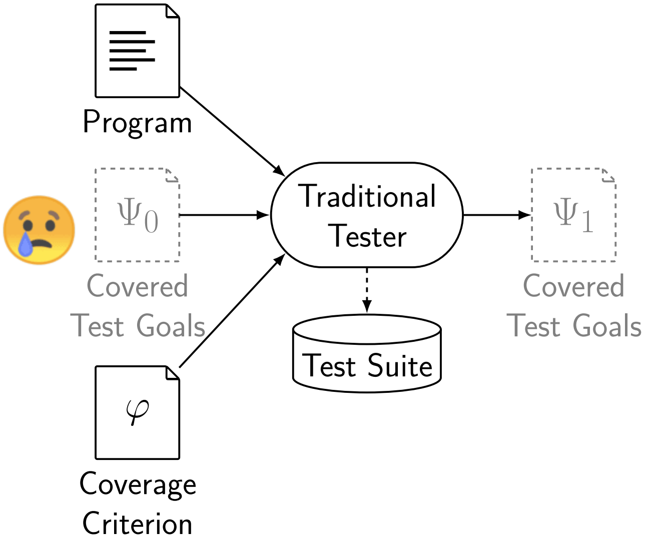
- Sequential
- Portfolio
- Strategy selection
- Compositional
- Cyclic



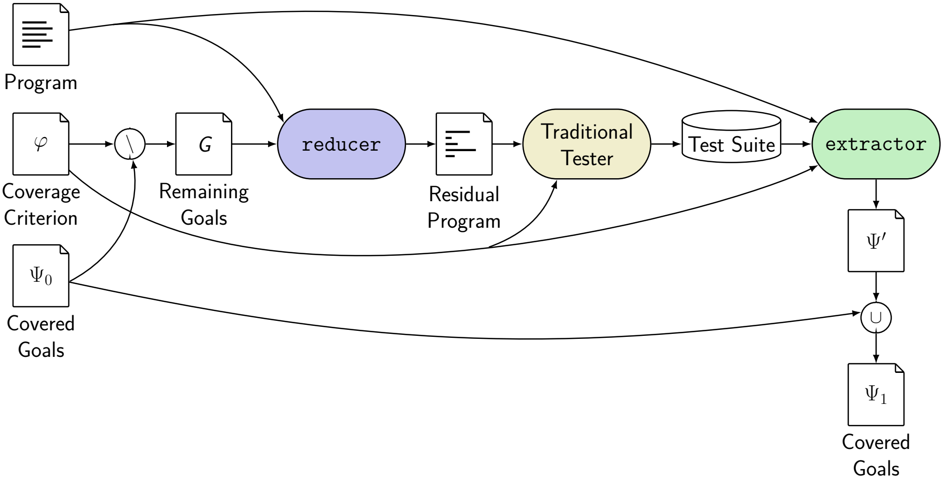
Combinations

- Sequential
- Portfolio
- Strategy selection
- Compositional
- Cyclic
- ...

Traditional Tester



Traditional Tester $\Rightarrow$ Conditional Tester



Reducer

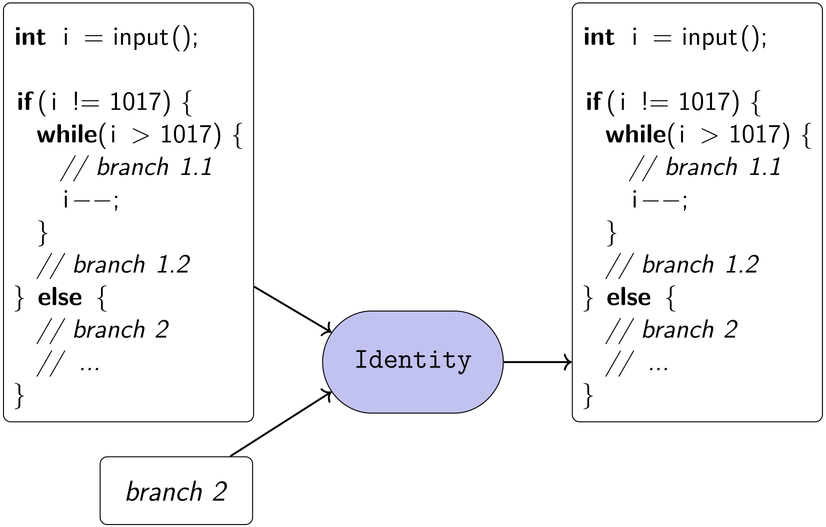
reducer

- Input: Program P, remaining test goals G
- Output: Residual program P'
- P' *reachability-equivalent* to P with regard to G

Reachability Equivalence [2]

Each program input that reaches a test goal of G in P' reaches the same test goal in P

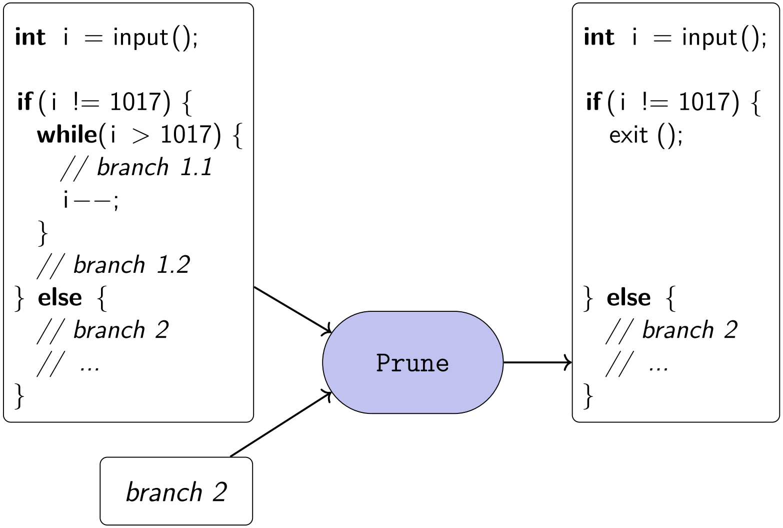
Reducer Example: Identity



```
int i = input();

if(i != 1017) {
  while(i > 1017) {
    // branch 1.1
    i--;
  }
  // branch 1.2
} else {
  // branch 2
  // ...
}
```

```
int i = input();

if(i != 1017) {
  while(i > 1017) {
    // branch 1.1
    i--;
  }
  // branch 1.2
} else {
  // branch 2
  // ...
}
```

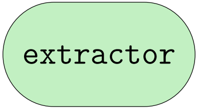
Reducer Example: Pruning



```
int i = input();

if (i != 1017) {
  while(i > 1017) {
    // branch 1.1
    i--;
  }
  // branch 1.2
} else {
  // branch 2
  // ...
}
```

```
int i = input();

if (i != 1017) {
  exit();




} else {
  // branch 2
  // ...
}
```

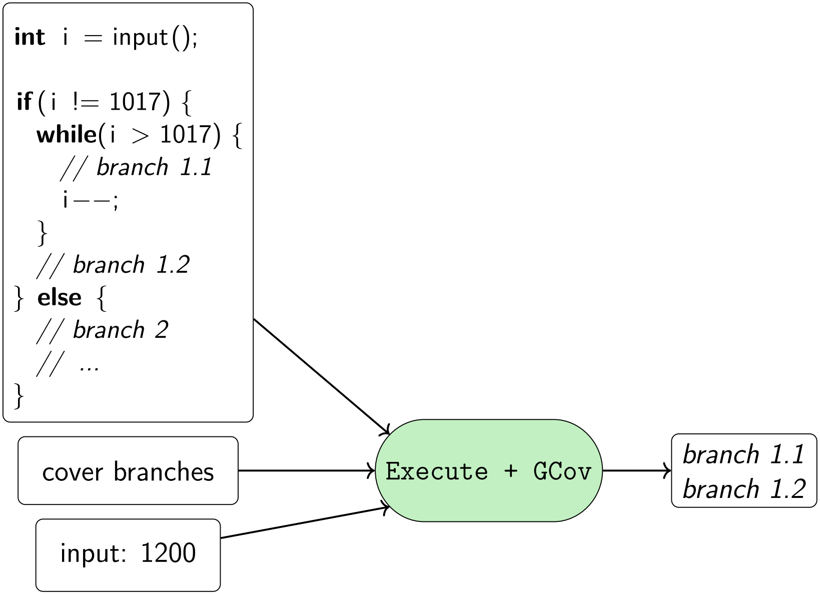
Test-Goal Extractor



- Input: Program P, coverage criterion φ, test suite S
- Output: Test goals Ψ covered by S

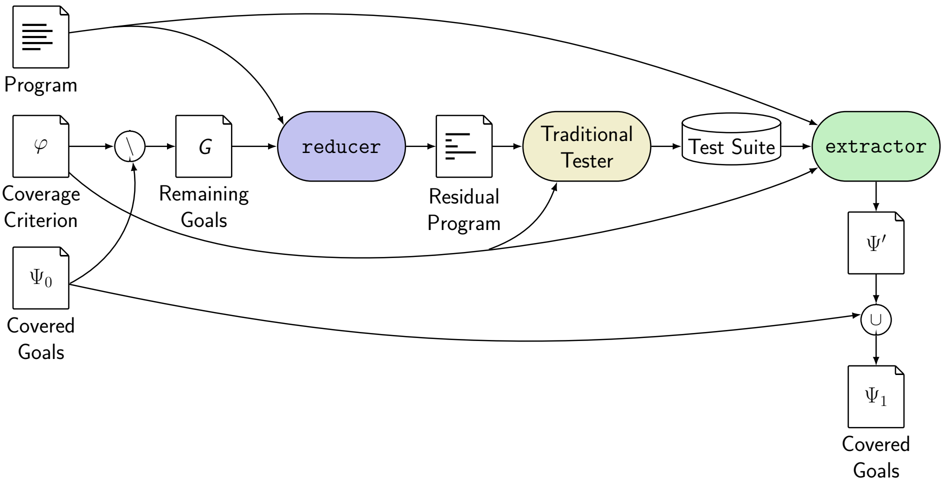
- Example: Test execution $+$ coverage measurement

Extractor Example



```
int i = input();

if (i != 1017) {
  while(i > 1017) {
    // branch 1.1
    i--;
  }
  // branch 1.2
} else {
  // branch 2
  // ...
}
```

Traditional ⇒ Conditional Tester



Implementation

- CondTest
  https://gitlab.com/sosy-lab/software/conditional-testing

1. Test-Comp tester ⇒ Conditional Tester
2. SV-COMP formal verifier ⇒ Conditional Tester
3. Sequential combination
4. Cyclic combination

- Plug-and-play through SV-COMP/Test-Comp modules

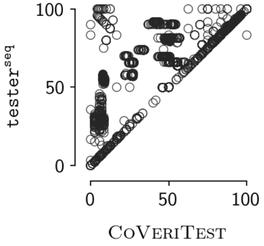
Test-Comp Tester $\Rightarrow$ Conditional Tester

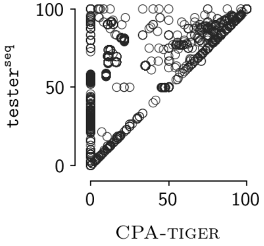
- Reducer: Prune
- Test Generator: Test-Comp Tester
- Extractor: Test execution + coverage measurement

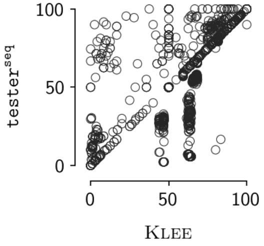
SV-COMP Verifier ⇒ Conditional Tester

- Reducer: Annotate __VERIFIER_error() calls at goals
- Test Generator: SV-COMP verifier (reachability) + witness-to-test [1]
- Extractor: Test execution + coverage measurement
- Wrapped in cyclic tester to get multiple test cases

Evaluation (I)







- Tool (900 s)
- CPA-tiger + CoVeriTest + Klee (300 s each)

Evaluation (II)

- CPA-TIGER + COVERITEST + KLEE (300 s each)
- Sequential without (`id`) and with info exchange (`prune`)

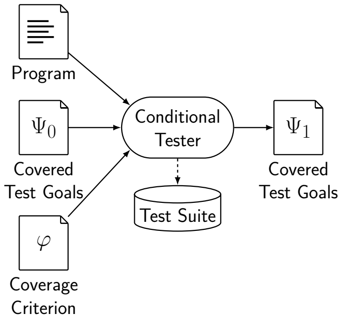
Task	branch coverage		
	`id`	$\rightarrow$	`prune`
mod3.c.v+sep-reducer	75.0	+ 5.00	80.0
Problem07_label35	52.0	+ 2.00	54.0
Problem07_label37	54.2	+ 1.97	56.2
Problem04_label35	79.5	+ 1.79	81.3
Problem06_label02	57.0	+ 1.70	58.7
Problem06_label27	57.5	+ 1.09	58.6
Problem04_label02	80.2	+ 1.06	81.3
Problem06_label18	57.5	+ 1.05	58.6
Problem04_label16	79.1	+ 1.01	80.1
Problem04_label34	80.2	+ 0.99	81.2

Evaluation (III)

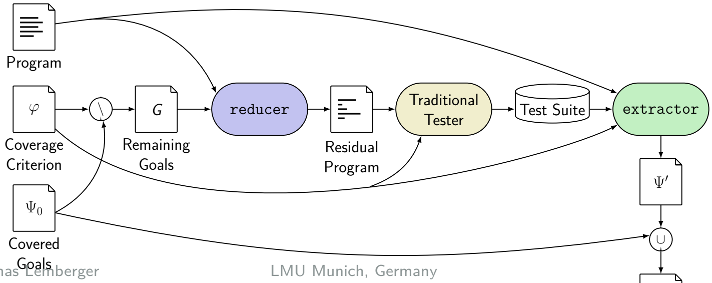
- CPA-tiger + CoVeriTest + Klee (prune)
- CPA-tiger + CoVeriTest + Klee (200 s each) + Esbmc (300 s) (vb)

Task	branch coverage		
	prune	$\rightarrow$	vb
Problem08_label30	5.72	+56.2	62.0
Problem08_label32	5.72	+56.1	61.9
Problem08_label06	5.72	+56.1	61.8
Problem08_label35	5.72	+56.0	61.7
Problem08_label00	5.72	+55.9	61.6
Problem08_label11	5.72	+55.8	61.5
Problem08_label19	5.72	+55.7	61.5
Problem08_label29	5.67	+55.7	61.4
Problem08_label22	5.72	+55.7	61.5
Problem08_label56	5.72	+55.7	61.5

Conclusion



Program
Ψ_0
Covered Test Goals
φ
Coverage Criterion
Conditional Tester
Test Suite
Ψ_1
Covered Test Goals



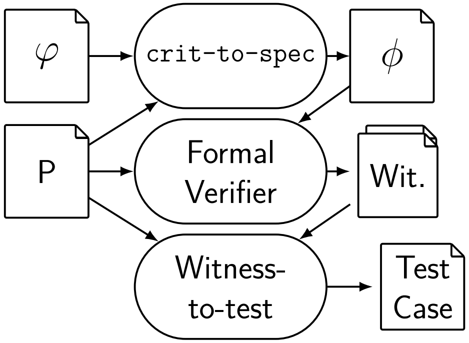
Program
φ
Coverage Criterion
Ψ_0
Covered Goals
$\setminus$
G
Remaining Goals
reducer
Residual Program
Traditional Tester
Test Suite
extractor
Ψ'
$\cup$

References

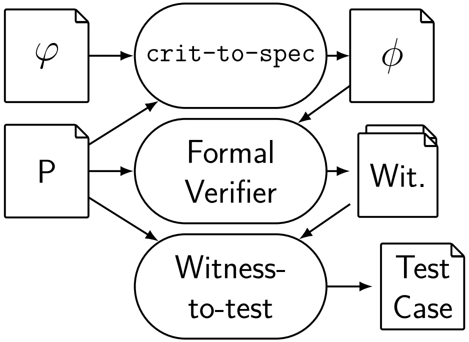
[1] D. Beyer, M. Dangl, T. Lemberger, and M. Tautschnig.
Tests from witnesses: Execution-based validation of verification results.
In *Proc. TAP*, LNCS 10889, pages 3–23. Springer, 2018.

[2] M. Harman, L. Hu, R. M. Hierons, J. Wegener, H. Sthamer, A. Baresel, and M. Roper.
Testability transformation.
IEEE Trans. Software Eng., 30(1):3–16, 2004.

Tests from Formal Verification (I)



Tests from Formal Verification (I)

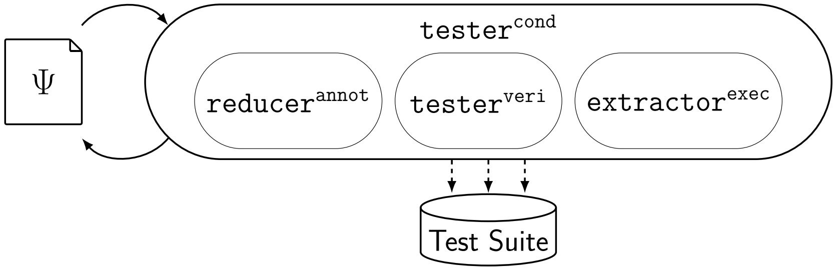


- Only one test per verifier-run

⇒ conditional testing

Tests from Formal Verification (II)

- Reducer: identity + annotate goals with __VERIFIER_error
- Apply cyclic tester


Ψ
tester^cond
reducer^annot
tester^veri
extractor^exec
Test Suite