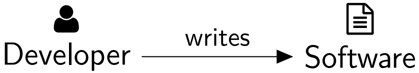
CoVeriTest

Cooperative Verifier-Based Testing

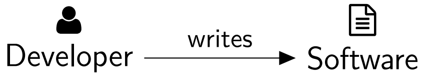
Dirk Beyer and **Marie-Christine Jakobs**

FASE 2019

Testing for Software Quality Assurance

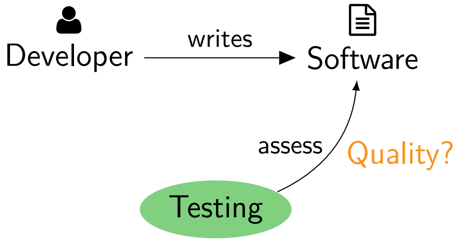


Testing for Software Quality Assurance

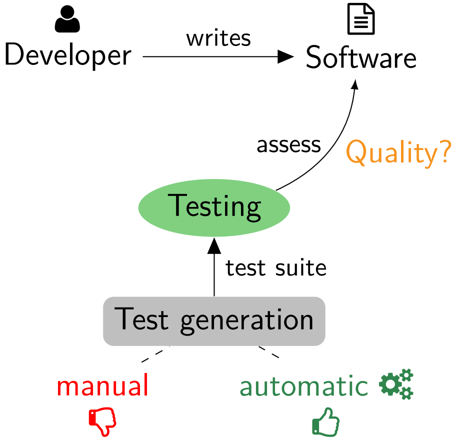


Quality?

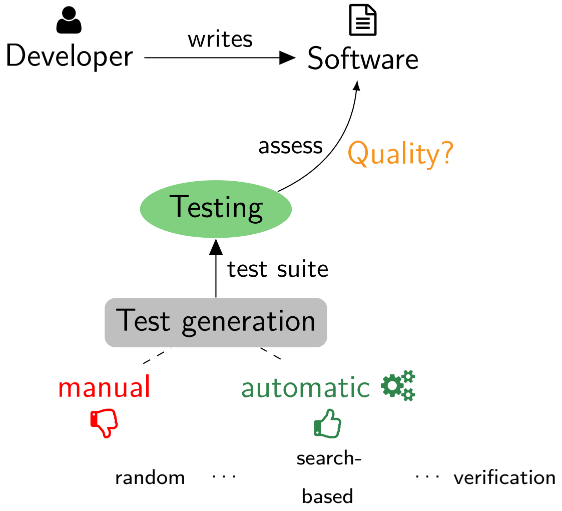
Testing for Software Quality Assurance



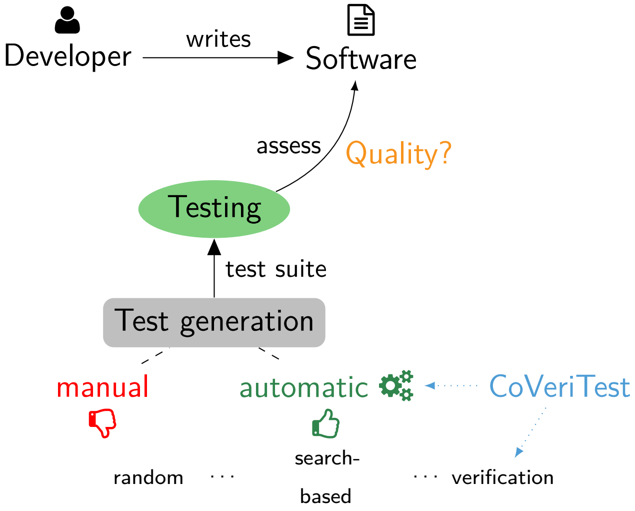
Testing for Software Quality Assurance



Testing for Software Quality Assurance



Testing for Software Quality Assurance

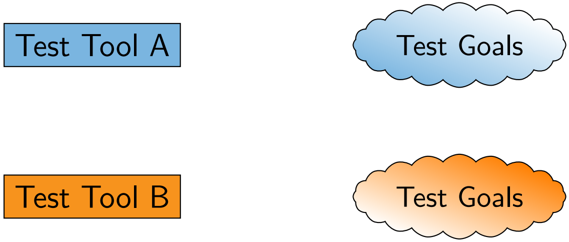


One Problem in Test-Suite Generation

Task:
Generate a test-suite for program P that covers test goals

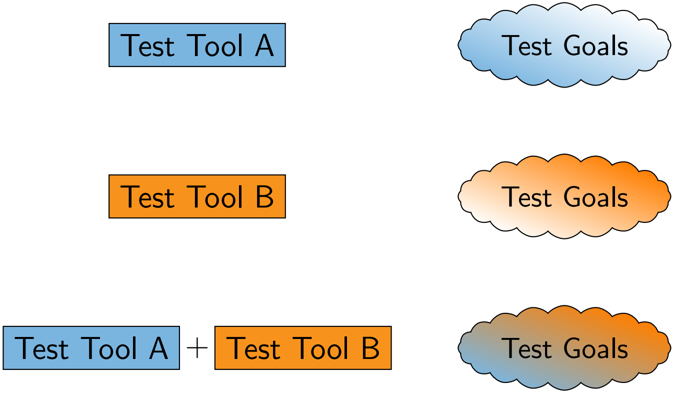
One Problem in Test-Suite Generation

Task:
Generate a test-suite for program P that covers test goals



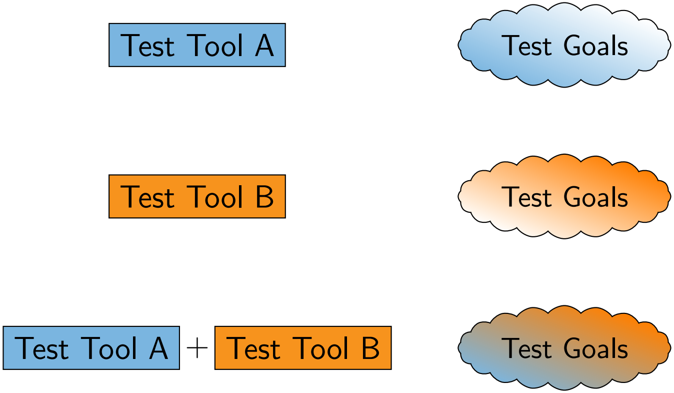
One Problem in Test-Suite Generation

Task:
Generate a test-suite for program P that covers test goals



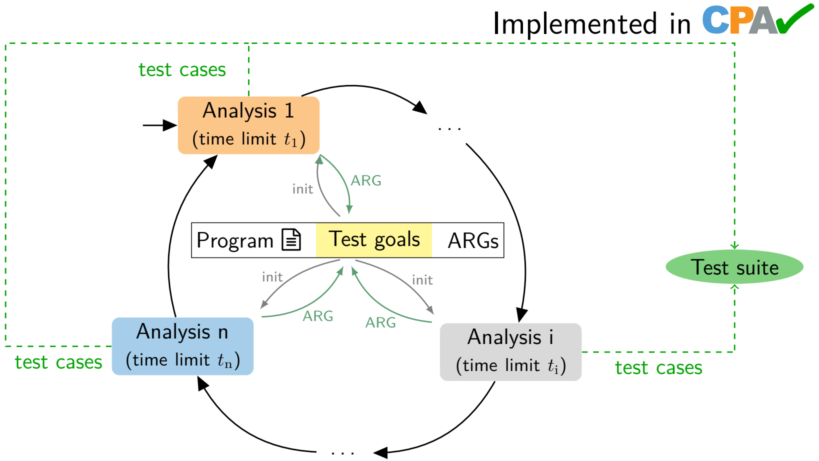
One Problem in Test-Suite Generation

Task:
Generate a test-suite for program P that covers test goals



Need to combine different test tools $\Rightarrow$ Use CoVeriTest

Overview of CoVeriTest Approach



Information exchange realized with ARGs and init procedure

ARG: graph representation of explored, abstract state space

One Analysis Run in CoVeriTest – Initialization

1. Provides program and test goals
2. Realizes information exchange based on previous ARGs

One Analysis Run in CoVeriTest – Initialization

1. Provides program and test goals
2. Realizes information exchange based on previous ARGs
   - Cooperation between analyses
     - Based on conditional model checking [Beyer et al., FSE'12]

One Analysis Run in CoVeriTest – Initialization

1. Provides program and test goals
2. Realizes information exchange based on previous ARGs
    - Cooperation between analyses
        - Based on conditional model checking [Beyer et al., FSE'12]
        - Avoid to explore already explored state space
        - For restriction, use ARG of previous analysis

One Analysis Run in CoVeriTest – Initialization

1. Provides program and test goals
2. Realizes information exchange based on previous ARGs
    - Cooperation between analyses
        - Based on conditional model checking [Beyer et al., FSE'12]
        - Avoid to explore already explored state space
        - For restriction, use ARG of previous analysis
    - Reuse own knowledge

      $\Rightarrow$ set up initial ARG $(N, E, root, F, \pi)$

      use ARG returned by last run of this analysis

One Analysis Run in CoVeriTest – Initialization

1. Provides program and test goals
2. Realizes information exchange based on previous ARGs
   - Cooperation between analyses
     - Based on conditional model checking [Beyer et al., FSE'12]
     - Avoid to explore already explored state space
     - For restriction, use ARG of previous analysis
   - Reuse own knowledge

     $\Rightarrow$ set up initial ARG $(N, E, root, F, \pi)$

     use ARG returned by last run of this analysis
     - Start from scratch

       $root = (pc_0, true) \qquad N = F = \{root\} \qquad E = \pi = \emptyset$

One Analysis Run in CoVeriTest – Initialization

1. Provides program and test goals
2. Realizes information exchange based on previous ARGs
   - Cooperation between analyses
     - Based on conditional model checking [Beyer et al., FSE'12]
     - Avoid to explore already explored state space
     - For restriction, use ARG of previous analysis
   - Reuse own knowledge

     $\Rightarrow$ set up initial ARG $(N, E, root, F, \pi)$
     use ARG returned by last run of this analysis
     - Start from scratch
       $root = (pc_0, true)$ $N = F = \{root\}$ $E = \pi = \emptyset$
     - Reuse abstraction level π

One Analysis Run in CoVeriTest – Initialization

1. Provides program and test goals
2. Realizes information exchange based on previous ARGs
   - Cooperation between analyses
     - Based on conditional model checking [Beyer et al., FSE'12]
     - Avoid to explore already explored state space
     - For restriction, use ARG of previous analysis
   - Reuse own knowledge

     $\Rightarrow$ set up initial ARG $(N, E, root, F, \pi)$
     use ARG returned by last run of this analysis
     - Start from scratch
       $root = (pc_0, true) \qquad N = F = \{root\} \qquad E = \pi = \emptyset$
     - Reuse abstraction level π
     - Continue exploration, i.e., reuse ARG

One Analysis Run in CoVeriTest – Execution

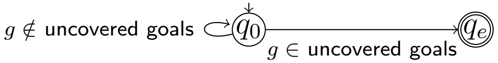
Perform reachability analysis of uncovered goals

$$\begin{tikzcd} q_0 \arrow[loop left, "g \notin \text{uncovered goals}"] \arrow[r, "g \in \text{uncovered goals}"'] & q_e \end{tikzcd}$$

Feasible counterexample $\Rightarrow$ uncovered goal reached

One Analysis Run in CoVeriTest – Execution

Perform reachability analysis of uncovered goals



$$\begin{tikzcd} \arrow[r] & q_0 \arrow[loop left, "g \notin \text{uncovered goals}"] \arrow[r, "g \in \text{uncovered goals}"'] & q_e \end{tikzcd}$$

Feasible counterexample ⇒ uncovered goal reached

- Construct test cases from feasible counterexamples [Beyer et al., ICSE'04]
- Goals for which a test cases is constructed become covered

One Analysis Run in CoVeriTest – Execution

Perform reachability analysis of uncovered goals

$$\begin{tikzcd} q_0 \arrow[loop left, "g \notin \text{uncovered goals}"] \arrow[rrr, "g \in \text{uncovered goals}"'] & & & q_e \end{tikzcd}$$

Feasible counterexample $\Rightarrow$ uncovered goal reached

- Construct test cases from feasible counterexamples [Beyer et al., ICSE'04]
- Goals for which a test cases is constructed become covered

- Stops if goals covered, total or analysis time limit exceeded

Output: test cases + explored state space (ARG)

Research Questions

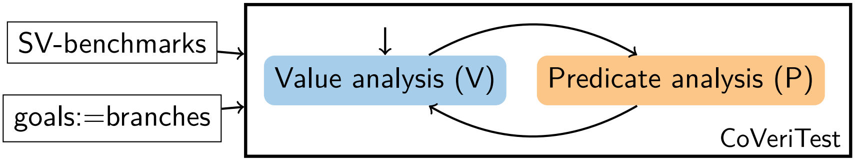
Internal comparison

1. How to configure CoVeriTest?
   time limits, information exchange
2. Does CoVeriTest's interleaving improve over
   - its single analyses,
   - their parallel combination?

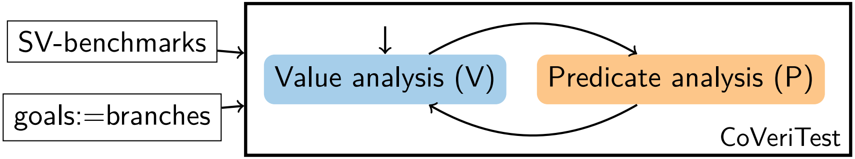
External comparison

3. How does CoVeriTest compete with state-of-the-art?

Evaluation Set Up

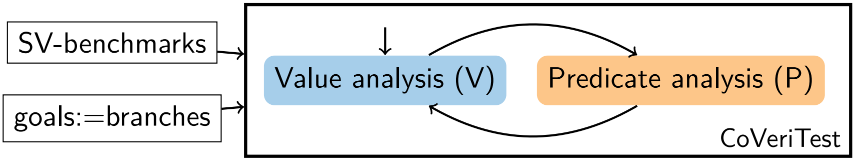


Evaluation Set Up



- Resources
  - 15 GB of memory, 900 s in total
  - Analysis limits (V,P) in (s)
    (10,10) (50,50) (100,100) (250,250) (80,20) (20,80)

Evaluation Set Up



- Resources
  - 15 GB of memory, 900 s in total
  - Analysis limits (V,P) in (s)
    (10,10) (50,50) (100,100) (250,250) (80,20) (20,80)
- Information exchange mode

	Reuse own knowledge		
Coop.	None	Abstraction level	Continue exploration
None	✓	✓	✓
V→P	✓		✓
V←P	✓		✓
V↔P	✓	✓	✗

CoVeriTest Configuration: Which Time Limit?

Per mode m consider

- all 6 configurations C_i with mode m, **but** different limits
- distribution of relative coverage (i.e., relative to best result)

CoVeriTest Configuration: Which Time Limit?

Per mode m consider

- all 6 configurations C_i with mode m, **but** different limits
- distribution of relative coverage (i.e., relative to best result)

Computing relative coverage of a task

	C_1	C_2	C_3	C_4	C_5	C_6	Maximum
# covered	7	13	4	16	9	11	16
relative coverage	$\frac{7}{16}$	$\frac{13}{16}$	$\frac{4}{16}$	$\frac{16}{16}$	$\frac{9}{16}$	$\frac{11}{16}$	

CoVeriTest Configuration: Which Time Limit?

Per mode m consider

- all 6 configurations C_i with mode m, **but** different limits
- distribution of relative coverage (i.e., relative to best result)

$\Rightarrow$ results in two mode clusters

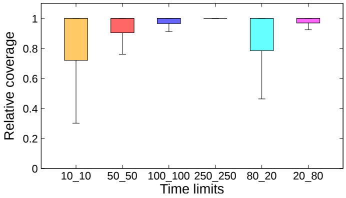
CoVeriTest Configuration: Which Time Limit?

Per mode m consider

- all 6 configurations C_i with mode m, **but** different limits
- distribution of relative coverage (i.e., relative to best result)

$\Rightarrow$ results in two mode clusters

discard own results



Boxes closer to one that are small are better

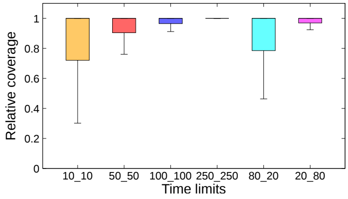
CoVeriTest Configuration: Which Time Limit?

Per mode m consider

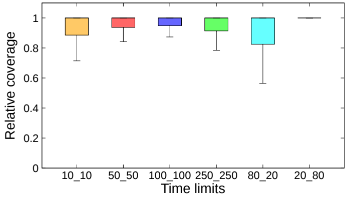
- all 6 configurations C_i with mode m, **but** different limits
- distribution of relative coverage (i.e., relative to best result)

$\Rightarrow$ results in two mode clusters

discard own results



reuse own knowledge



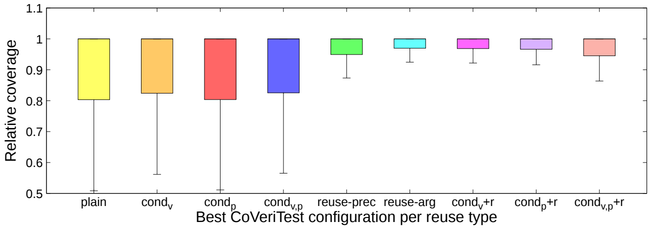
Boxes closer to one that are small are better

CoVeriTest Configuration: Which Mode?

- Use best time limit per mode
- Compare relative coverage of different modes

CoVeriTest Configuration: Which Mode?

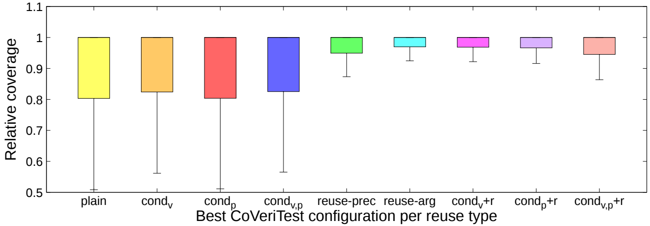
- Use best time limit per mode
- Compare relative coverage of different modes



discard own knowledge (250 s,250 s) ⟷ reuse own knowledge (20 s,80 s)

CoVeriTest Configuration: Which Mode?

- Use best time limit per mode
- Compare relative coverage of different modes



$\longleftrightarrow$ discard own knowledge (250 s,250 s) $\longleftrightarrow$ reuse own knowledge (20 s,80 s)

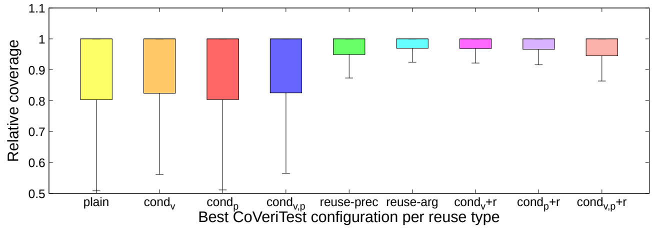
$\Rightarrow$ better reuse own knowledge
(i.e., reuse abstraction level or continue exploration)

CoVeriTest Configuration: Which Mode?

- Use best time limit per mode
- Compare relative coverage of different modes

Best

$\downarrow$



$\longleftrightarrow$ $\longleftrightarrow$

discard own knowledge
(250 s,250 s)

reuse own knowledge
(20 s,80 s)

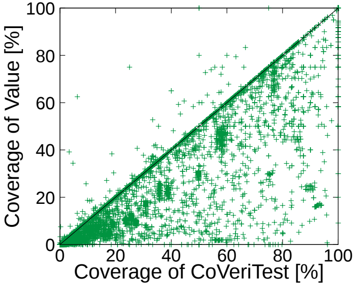
$\Rightarrow$ better reuse own knowledge
(i.e., reuse abstraction level or continue exploration)

Alone vs. Use in CoVeriTest Interleaving

- Compares absolute coverage (i.e., $\frac{\#covered}{\#total}$ goals)
- Uses best CoVeriTest configuration

Alone vs. Use in CoVeriTest Interleaving

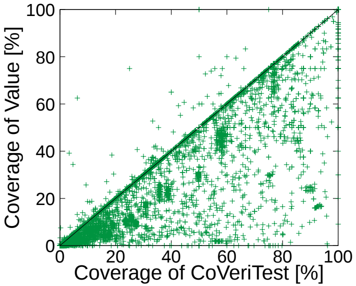
CoVeriTest better for points in lower right half

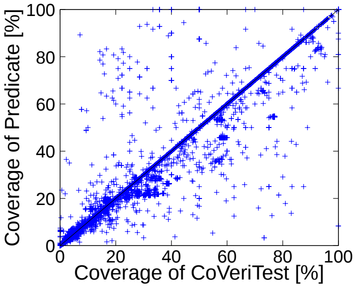


- Compares absolute coverage (i.e., $\frac{\#covered}{\#total}$ goals)
- Uses best CoVeriTest configuration

Alone vs. Use in CoVeriTest Interleaving

CoVeriTest better for points in lower right half

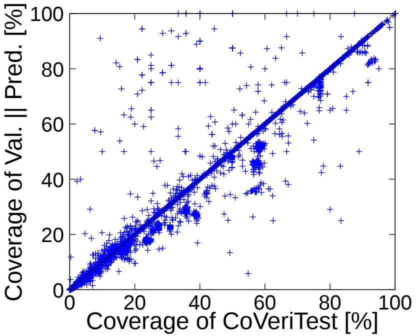




- Compares absolute coverage (i.e., $\frac{\#covered}{\#total}$ goals)
- Uses best CoVeriTest configuration

Parallel vs. Interleaving with CoVeriTest

CoVeriTest better for points in lower right half



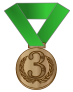
- Compares absolute coverage (i.e., $\frac{\#covered}{\#total}$ goals)
- Uses best CoVeriTest configuration

Comparison to State-of-the-Art

Participated in 1. Intl. Competition on Software Testing: 🥉

Comparison to State-of-the-Art

Participated in 1. Intl. Competition on Software Testing:



Complements other participants, e.g.,

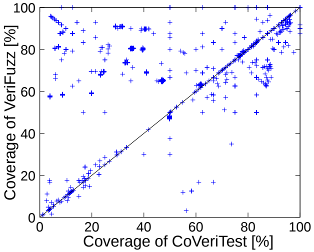
- Compare coverage measured by `gcov`

Comparison to State-of-the-Art

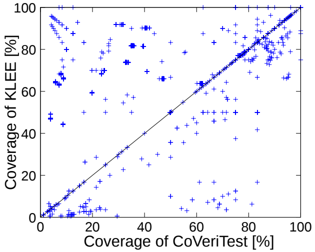
Participated in 1. Intl. Competition on Software Testing:

Complements other participants, e.g.,

first winner (VeriFuzz)



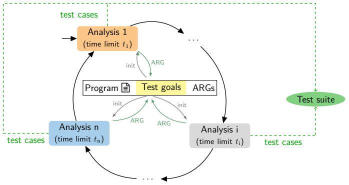
second winner (KLEE)



- Compare coverage measured by `gcov`
- CoVeriTest better in lower right half

Conclusion

CoVeriTest approach for cooperative, test generation

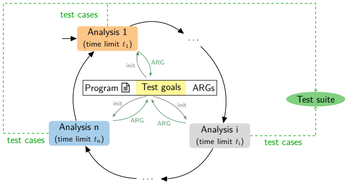


Conclusion

CoVeriTest approach for cooperative, test generation

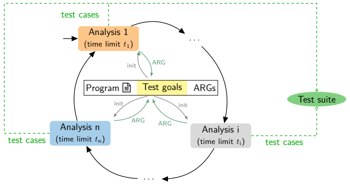
Evaluation results

1. Configuration
   - Continue own exploration
   - Prefer more mature analysis



Conclusion

CoVeriTest approach for cooperative, test generation



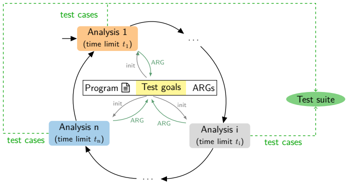
Evaluation results

1. Configuration
   - Continue own exploration
   - Prefer more mature analysis

2. Comparison
   - CoVeriTest improves over component, parallelization
   - CoVeriTest complements state-of-the-art tools

Conclusion

CoVeriTest approach for cooperative, test generation



Evaluation results

1. Configuration
   - Continue own exploration
   - Prefer more mature analysis

2. Comparison
   - CoVeriTest improves over component, parallelization
   - CoVeriTest complements state-of-the-art tools

`https://www.sosy-lab.org/research/coop-testgen/`

DOI 10.5281/zenodo.2566735