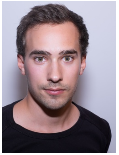
Decomposing Software Verification into Off-the-Shelf Components

An Application to CEGAR

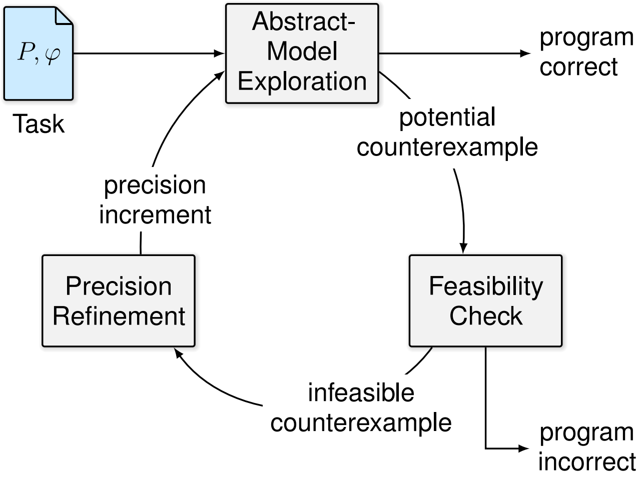
Thomas Lemberger

with Dirk Beyer, Jan Haltermann, and Heike Wehrheim



2022-04-03 COOP Workshop. Original slides by Jan Haltermann.

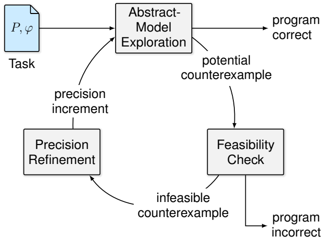
Motivation: Classic CEGAR



Motivation: Classic CEGAR - Issues

Issues:

- Many tools employ flavors of CEGAR (stateful)
- Common underlying schema
- New idea $\Rightarrow$ New implementation



Motivation: Classic CEGAR - Issues

```
int main() {
 unsigned int y = 1;
 while (1) {
   y = y + 2U * nondet();
   if (y != 0) {}
   else
     error();
 }
}
```

Craig interpolation:
$(y \mod 2 = 1)$

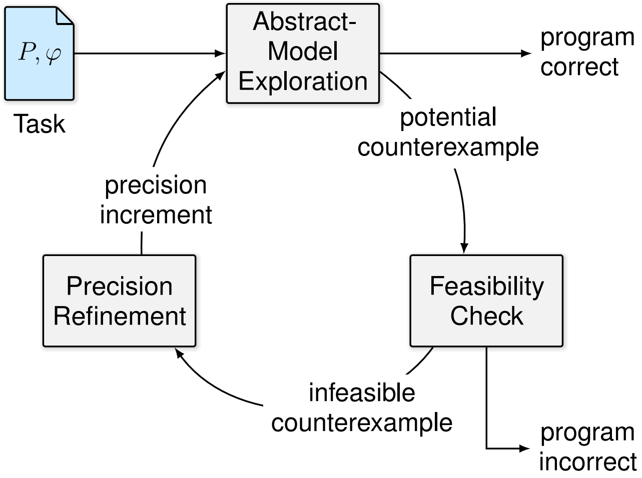
NEWTON refinement:
$1 \leq y{+}2{*}\lfloor((y{*}{-}1{+}1)/2)\rfloor$

```
int main(void) {
 unsigned int x = 0;
 unsigned short N = nondet();
 while (x < N) {
   x += 2;
 }
 if (x % 2 == 0) {}
 else
   error();
}
```

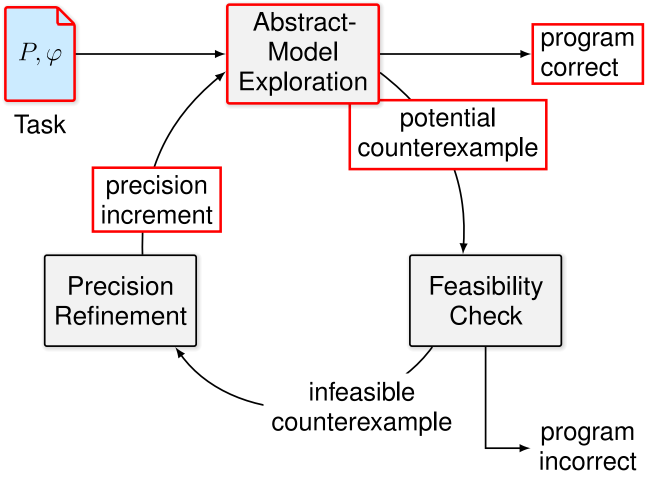
Craig interpolation:
no solution

NEWTON refinement:
$x \leq 2 * (x/2)$

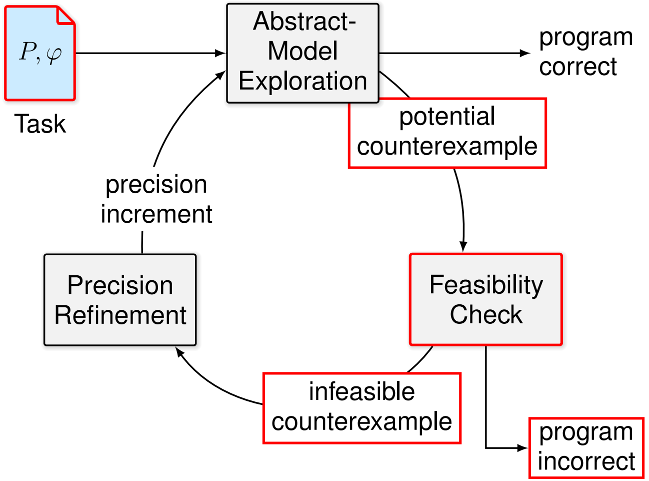
Decomposing CEGAR



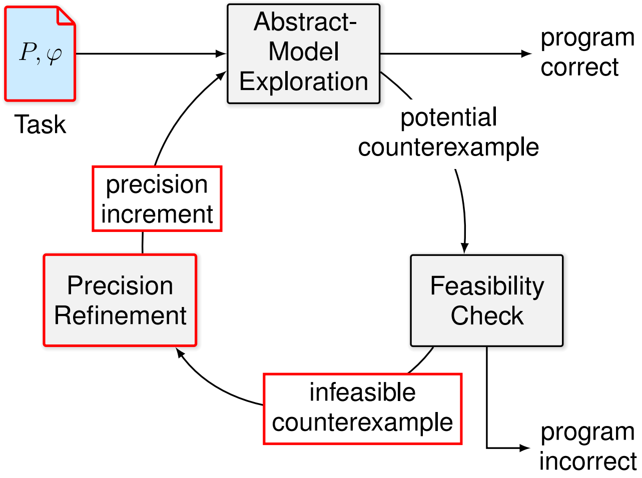
Decomposing CEGAR



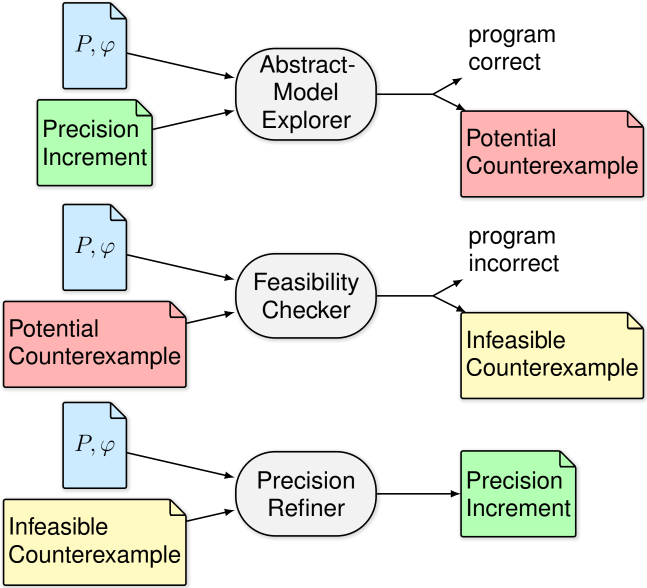
Decomposing CEGAR



Decomposing CEGAR



Component-based CEGAR (C-CEGAR)



Exchange Formats for C-CEGAR

Abstract description of counterexample in GraphML/XML

Exchange Formats for C-CEGAR

Violation Witness	Abstract description of counterexample in GraphML/XML
Path Witness	Rejected violation witness

Exchange Formats for C-CEGAR

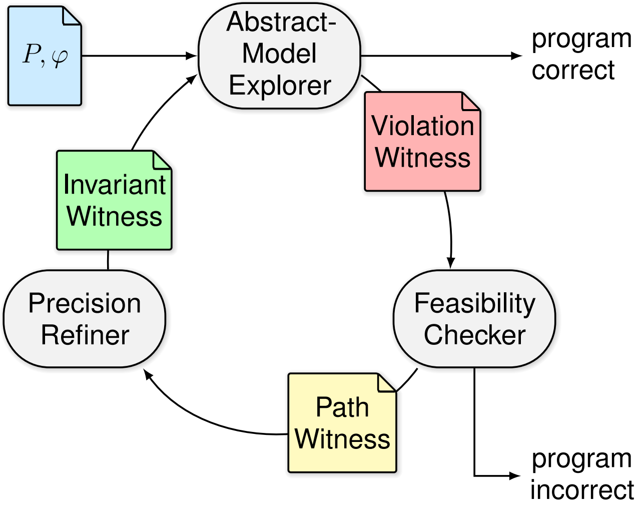
Violation Witness	Abstract description of counterexample in GraphML/XML
Path Witness	Rejected violation witness
Invariant Witness	Candidate invariants to help correctness proof in GraphML/XML

Exchange Formats for C-CEGAR

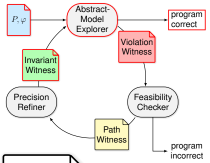
Violation Witness	Abstract description of counterexample in GraphML/XML
Path Witness	Rejected violation witness
Invariant Witness	Candidate invariants to help correctness proof in GraphML/XML

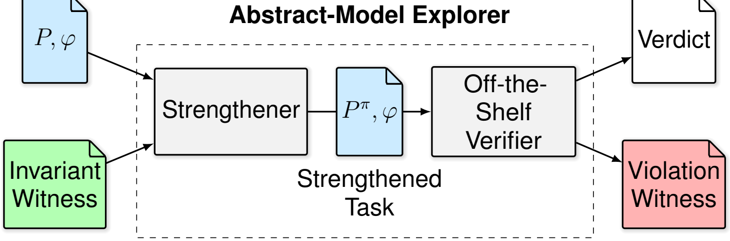
Formats required by SV-COMP $\Rightarrow$ good tool support

Component-based CEGAR (C-CEGAR)



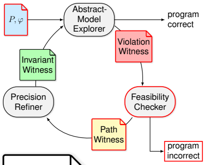
Use of Off-the-Shelf Components: Model Explorer

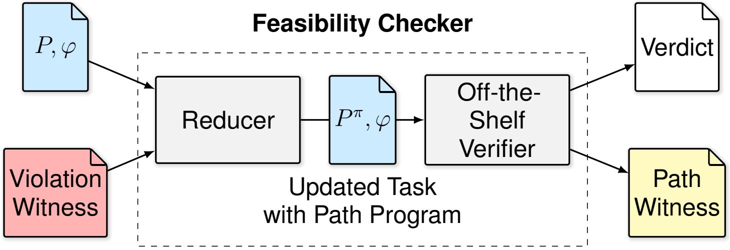




Idea of METAVAL (Beyer, Spiessl, CAV 2020)

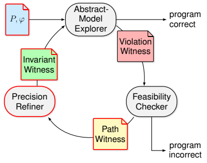
Use of Off-the-Shelf Components: Feasibility Checker

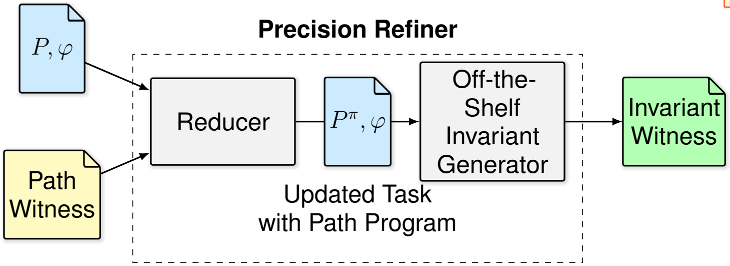




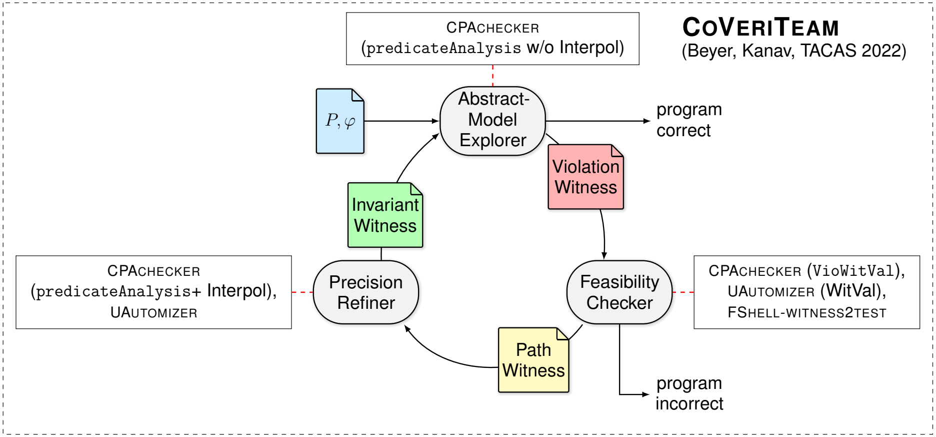
Idea of METAVAL (Beyer, Spiessl, CAV 2020)

Use of Off-the-Shelf Components: Precision Refiner





Implementation



Evaluation

Research Questions:

- RQ1: Overhead of a component-based approach
- RQ2: Cost using existing formats
- RQ3: Use of off-the-shelf components

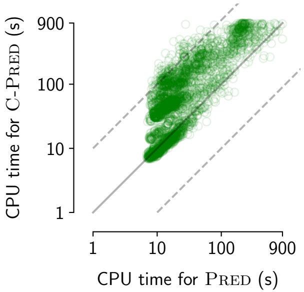
Dataset: SV-BENCHMARKS (8 347 tasks), SV-COMP setting

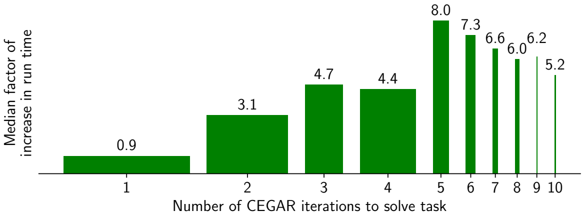
RQ1: Overhead of component based design

- Pred: CPACHECKER's predicate abstraction
- C-Pred: C-CEGAR with CPACHECKER as each component
- Proprietary precision exchange format

	correct			incorrect	
	overall	**proof**	**alarm**	**proof**	**alarm**
Pred	3769	2 556	1 213	3	9
C-Pred	3524	2 450	1 074	0	3

RQ1: Overhead of component based design





Run time of
Pred and C-Pred, per task

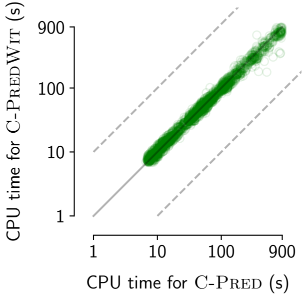
Median factor of run-time increase by C-Pred, compared to Pred.
Overall median increase is 3.2

RQ2: Cost of Standardized Formats

- C-PredWit: C-Pred, but use violation witnesses **and** invariant witnesses

		correct	
	overall	**proof**	**alarm**
C-Pred	3524	2 450	1 074
C-PredWit	2854	2 110	744

- Effectiveness reduces by 20%
- Reasons:
  - Not all discovered predicates are exported as invariants
  - No loop unrollings in witness



Run time of C-Pred and C-PredWit

RQ3: C-CEGAR using different components

RQ 3.1: C-PredWit + different feasibility checker (precision refiner: CPAchecker)

		correct			
	overall	**proof**	unique	**alarm**	unique
CPAchecker	1 573	978	61	595	317
FShell-witness2test	612	515	0	97	56
UAutomizer	1 225	918	1	307	20

RQ3: C-CEGAR using different components

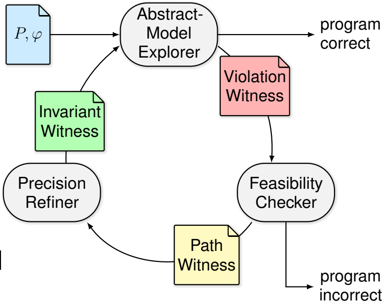
RQ 3.2: C-PredWit + different precision refiner (feasibility checker: CPAchecker)

		correct			
	overall	**proof**	unique	**alarm**	unique
CPAchecker	1 573	978	301	595	303
UAutomizer	1 016	716	41	300	6

Summary

- C-CEGAR breaks
  tightly coupled CEGAR into smaller,
  stand-alone components
- C-CEGAR clearly defines interfaces
- Flexible compositions in CoVeriTeam
- Evaluation shows:
  - Similar effectiveness as tightly coupled CEGAR
  - Different techniques improve effectiveness

⇒ Decompositions possible **and worth it**



Read our ICSE 2022 preprint now!