Reducer-Based Construction of Conditional Verifiers

Dirk Beyer

LMU Munich, Germany

@ CPAchecker/LDV Workshop, Moscow, 2018-09-25

Many Verification Tools Available

HipTNT+
CPA-Seq
VeriAbs
ConSequence
MU-CSeq
CPA-kInd
Blast
UTaipan
ESBMC-kind
UKojak
UAutomizer
Ceagle
ESBMC-incr
Lazy-CSeq-Abs
DepthK
ESBMC-falsi
AProVE
ESBMC
InterpChecker
Yogar-CBMC
CPA-BAM-Slicing
CIVL
2LS
Skink
SMACK
Symbiotic
PredatorHP
UL-CSeq
SymDIVINE
VIAP
CBMC
CPA-BAM-BnB
Lazy-CSeq
Map2Check
Forester

Vision

I have a dream ...

- ... that one day, all tools for formal methods work together to solve hard verification problems and make our world safer and more secure.
- ... that one day, model checkers and theorem provers can be integrated into the software-development process as seamless as unit testing today.
- ... that one day, model checkers, theorem provers, SMT solvers, and testers use common interfaces for interaction and composition.

Outline

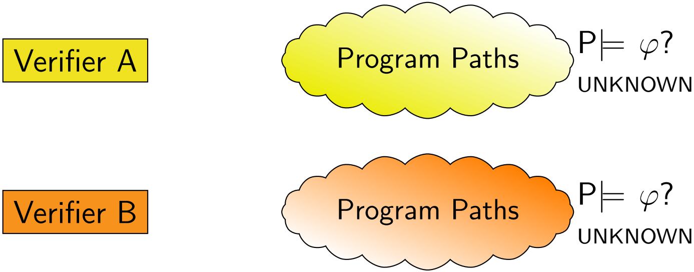
Dream is not utopian — there are a few approaches already ...

- Approach 1: Conditional Model Checking [FSE'12]
- Approach 2: Verification Witnesses [FSE'15, FSE'16]
- Approach 3: Tests from Witnesses [TAP'18]
- ...

Cooperative Verification by Conditional Model Checking and Reducers

Facing Hard Verification Tasks

Given: Program $\mathsf{P} \models \varphi$?



Verifier A

Program Paths

$\mathsf{P} \models \varphi$?

UNKNOWN

Verifier B

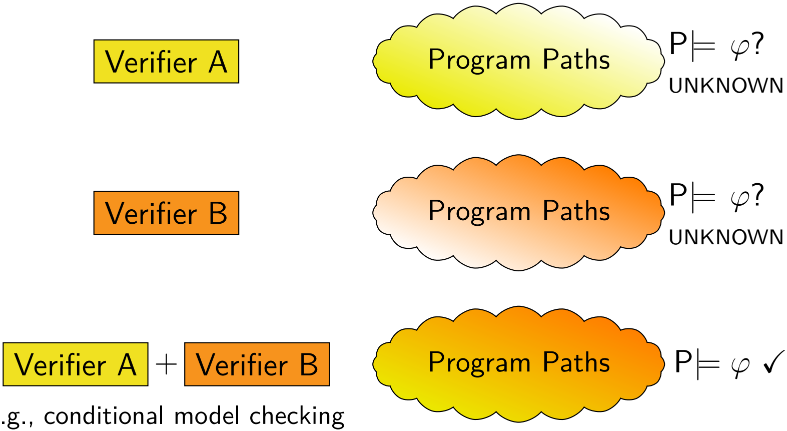
Program Paths

$\mathsf{P} \models \varphi$?

UNKNOWN

Facing Hard Verification Tasks

Given: Program P$\models \varphi$?



Verifier A

Program Paths

P$\models \varphi$?

UNKNOWN

Verifier B

Program Paths

P$\models \varphi$?

UNKNOWN

Verifier A + Verifier B

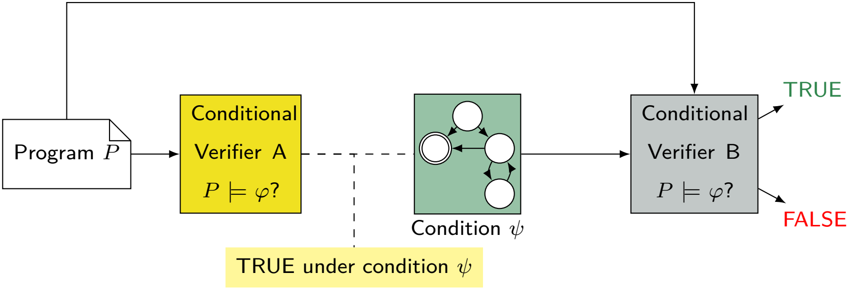
Program Paths

P$\models \varphi$ ✓

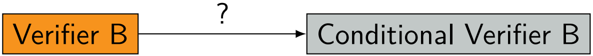
e.g., conditional model checking

Conditional Model Checking

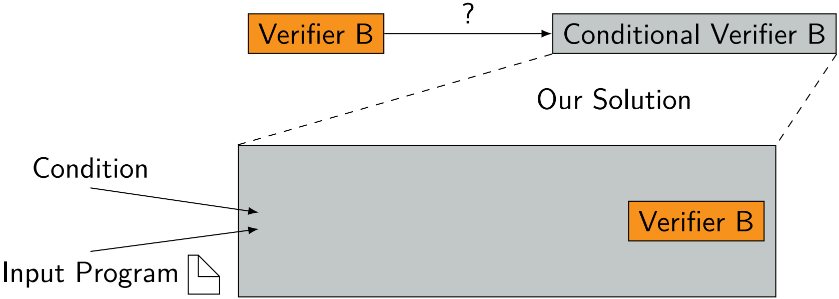
[Beyer/Henzinger/Keremoglu/Wendler FSE'12, DOI Link, Preprint Link]]


Program P
Conditional Verifier A $P \models \varphi$?
Condition ψ
TRUE under condition ψ
Conditional Verifier B $P \models \varphi$?
TRUE
FALSE

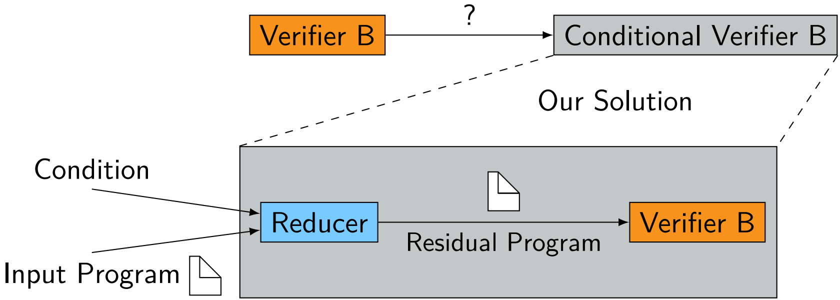
Reducer-Based Conditional Verifier Construction



Reducer-Based Conditional Verifier Construction



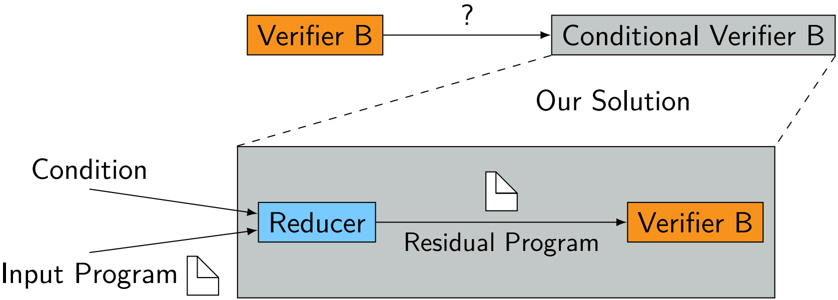
Reducer-Based Conditional Verifier Construction



Reducer (preprocessor)

- Builds standard input (C program)
- Representing a subset of paths
- Contains at least all non-verified paths

Reducer-Based Conditional Verifier Construction



Reducer (preprocessor)

- Builds standard input (C program)
- Representing a subset of paths
- Contains at least all non-verified paths

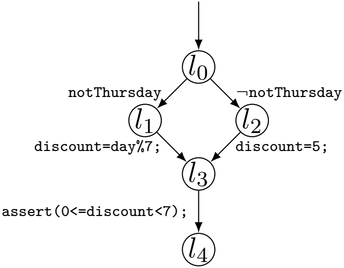
+ Verifier-unspecific approach

+ Many conditional verifiers possible

Example Program and Condition

Program

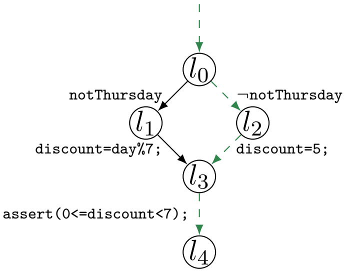
```
0: if(notThursday)
1:  discount=day%7;
   else
2:  discount=5;
3: assert(0<=discount<7);
4:
```



Example Program and Condition

Program

```
0: if(notThursday)
1:  discount=day%7;
   else
2:  discount=5;
3: assert(0<=discount<7);
4:
```

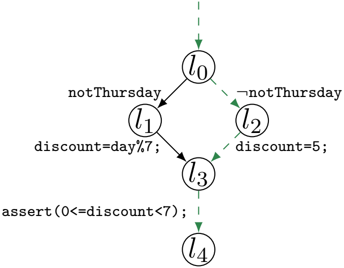


Verifier A only proofs else branch

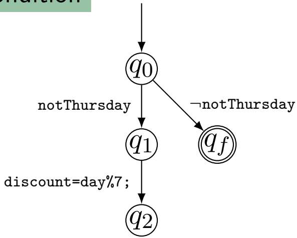
Example Program and Condition

Program

```
0: if(notThursday)
1:  discount=day%7;
   else
2:  discount=5;
3: assert(0<=discount<7);
4:
```

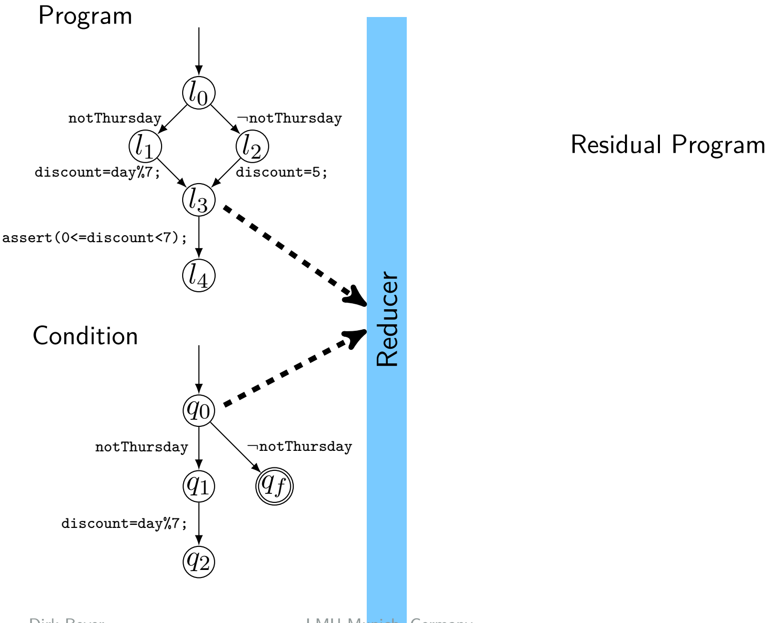


Condition

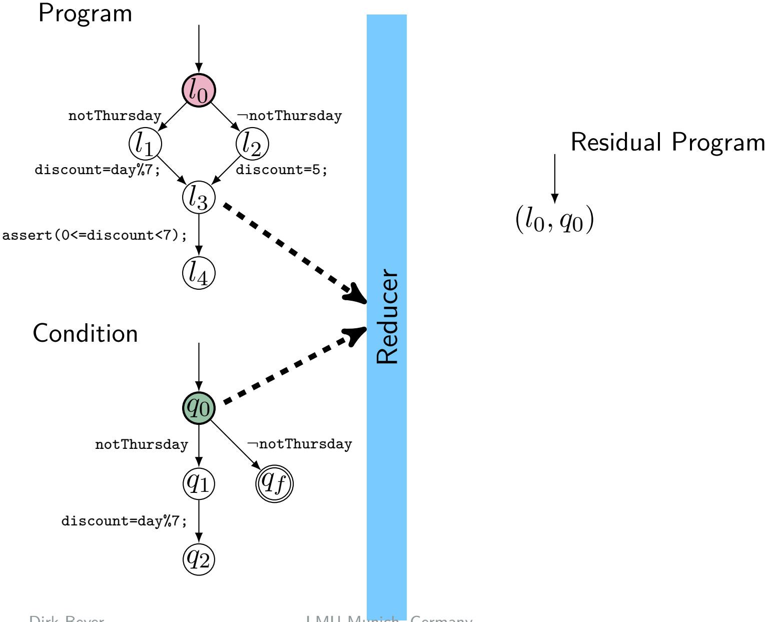


Verifier A only proofs else branch

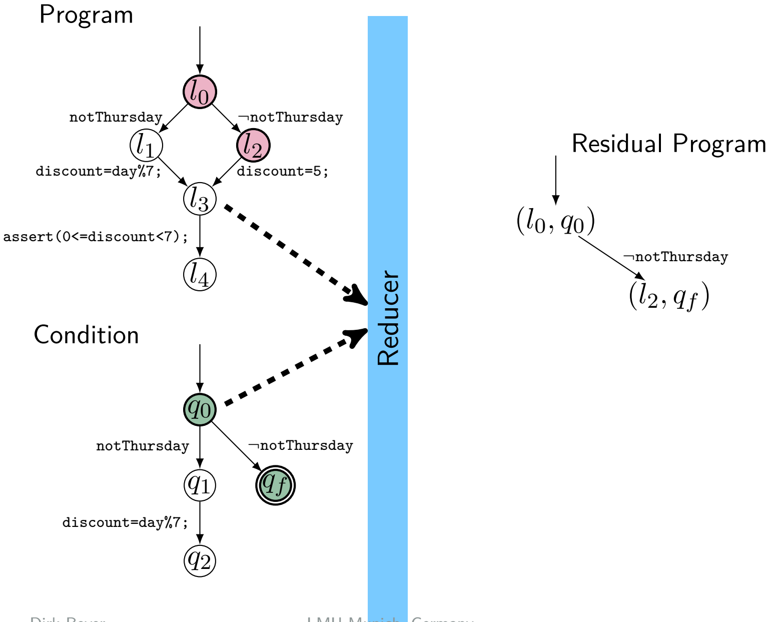
Reducer: Residual Program Construction



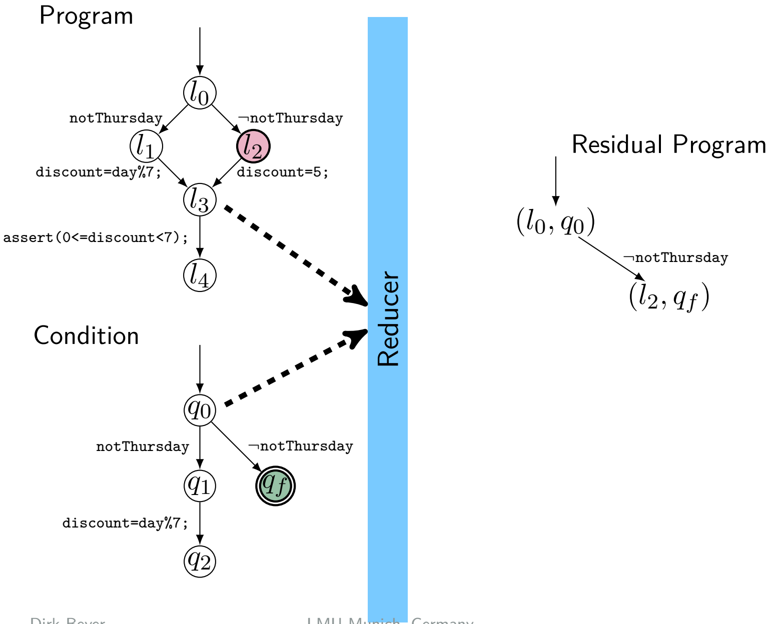
Reducer: Residual Program Construction



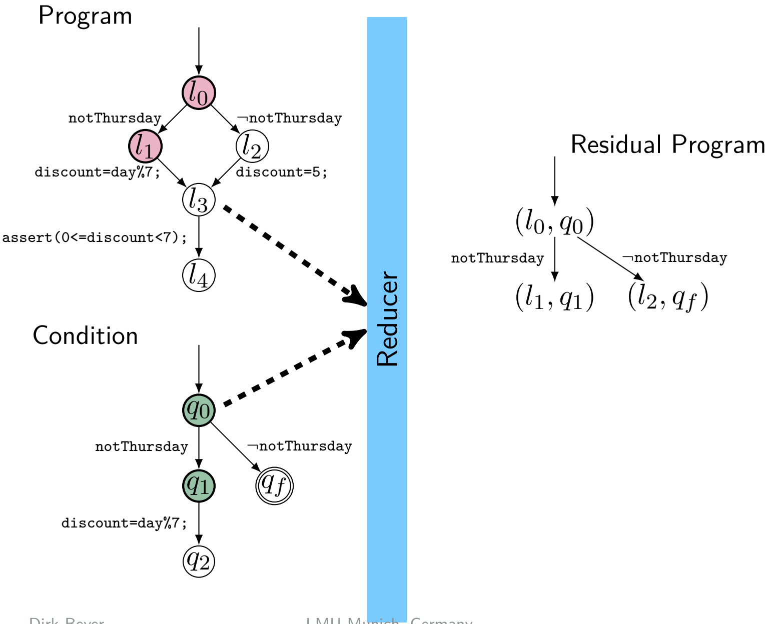
Reducer: Residual Program Construction



Reducer: Residual Program Construction



Reducer: Residual Program Construction



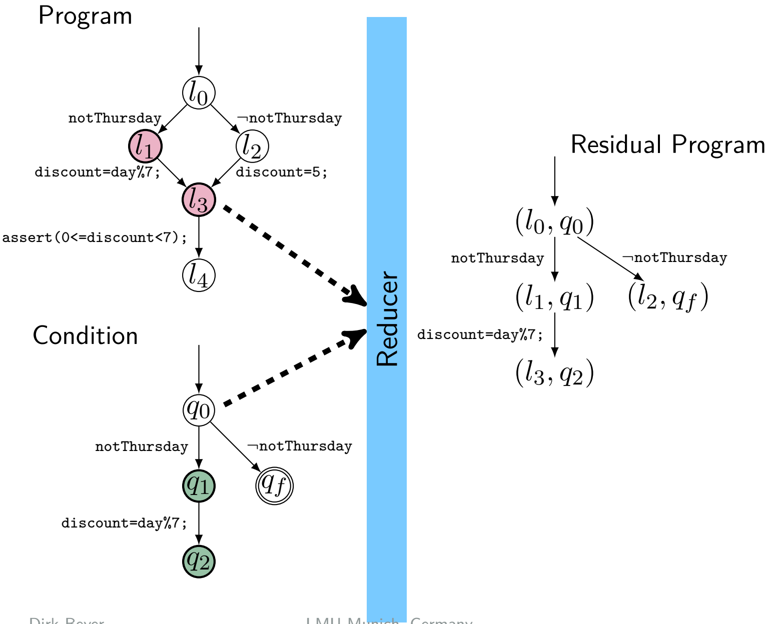
Program

Condition

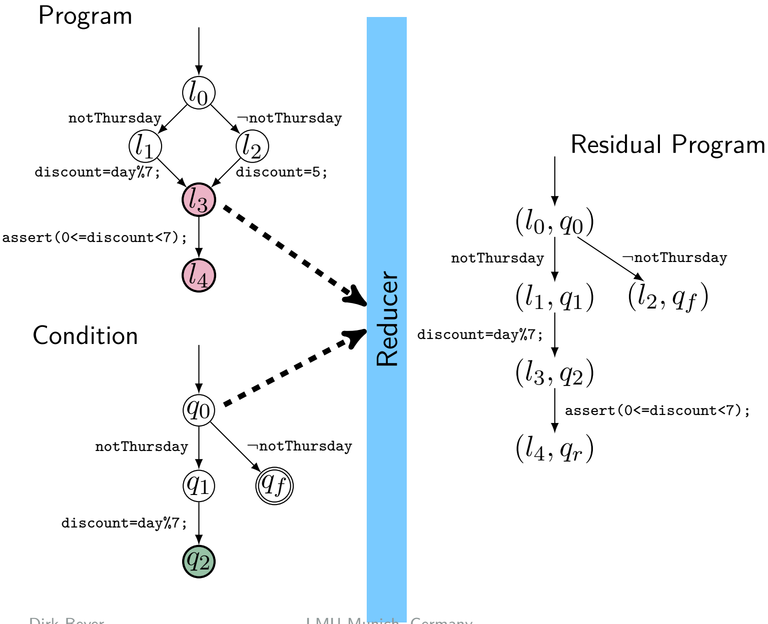
Reducer

Residual Program

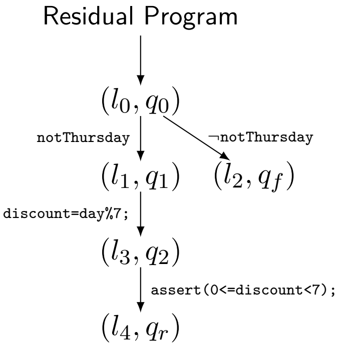
Reducer: Residual Program Construction



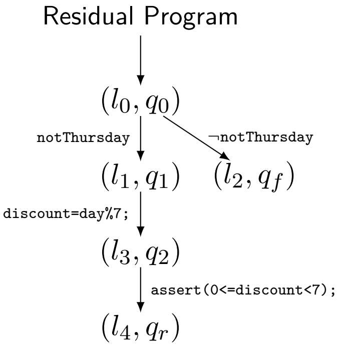
Reducer: Residual Program Construction



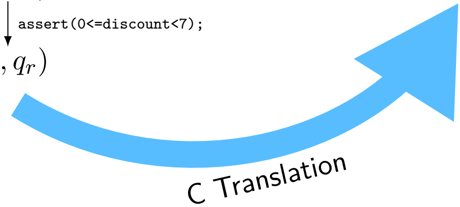
Reducer: C Transformation



Reducer: C Transformation

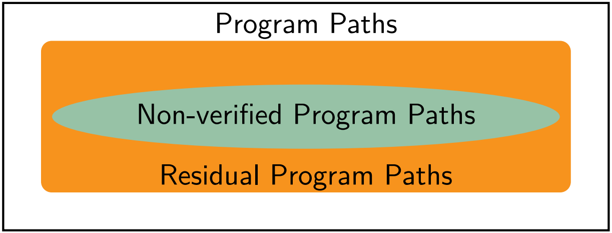


```
if(notThursday)
{
  discount=day%7;
  assert(0<=discount<7);
}
```



Reducer: Soundness

Residual Condition



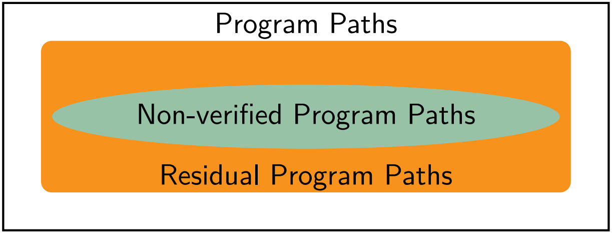
Program Paths

Non-verified Program Paths

Residual Program Paths

Reducer: Soundness

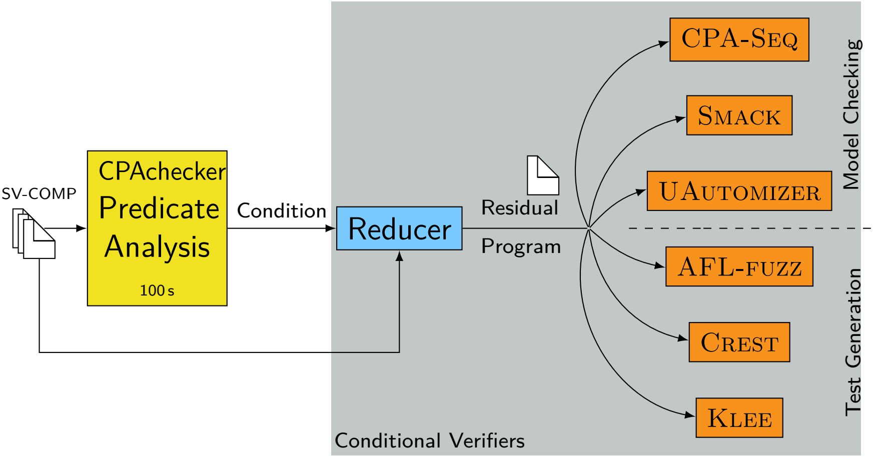
Residual Condition



Theorem

Presented reducer fulfills residual condition.

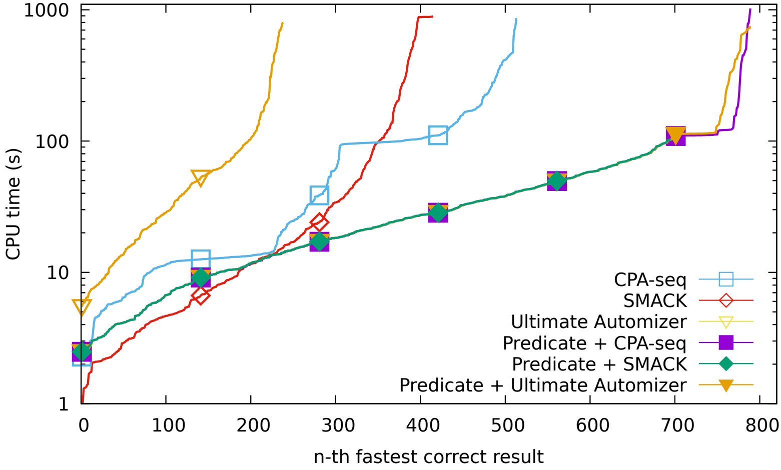
Evaluation Setup



Small Extract of Results

		CPA-Seq		UAutomizer		Predicate +Reducer +CPA-Seq		Predicate +Reducer +UAutomizer	
Task	R	S	t(s)	S	t(s)	S	t(s)	S	t(s)
P15l01	T	✗	910	✗	900	✓	120	✓	130
flood4	T	✗	910	✗	910	✓	450	✗	1100
newt3_6	F	✗	950	✗	490	✗	910	✓	260
P07l38	T	✗	950	✗	910	✗	1100	✓	470

Effectiveness on Hard Tasks

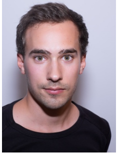


More Information: Reducer-Based Construction of Conditional Verifiers

[Proc. ICSE 2018, pages 1182–1193, ACM. DOI Link, Preprint Link]

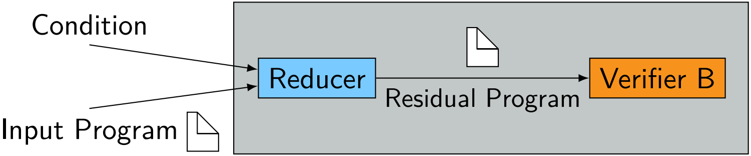
Dirk Beyer, Marie-Christine Jakobs, Thomas Lemberger, and Heike Wehrheim

LMU Munich, Germany and Paderborn University, Germany



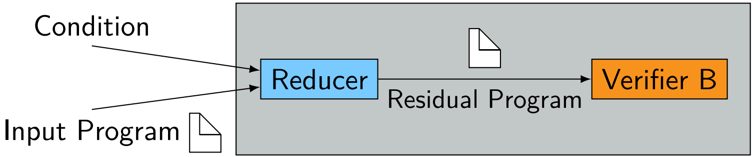
Conclusion — Reducer-Based CMC

- Template-based conditional verifier construction



Conclusion — Reducer-Based CMC

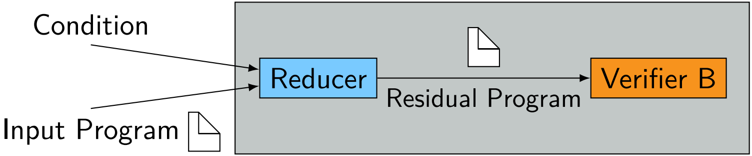
- Template-based conditional verifier construction



- One Reducer
  - Proven sound
  - Used in many conditional verifiers

Conclusion — Reducer-Based CMC

- Template-based conditional verifier construction

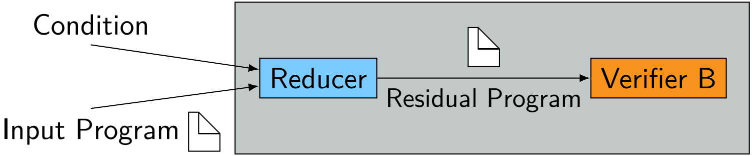


- One Reducer
  - Proven sound
  - Used in many conditional verifiers

- Effective on hard tasks for verifiers and test tools

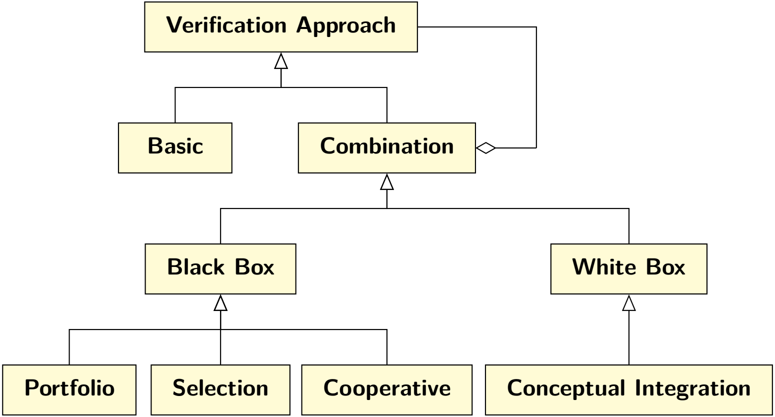
Conclusion — Reducer-Based CMC

- Template-based conditional verifier construction



- One Reducer
  - Proven sound
  - Used in many conditional verifiers
- Effective on hard tasks for verifiers and test tools
- Future Work
  - More reducers
  - Using conditions from other tools

Overview
Approaches for Combinations


Verification Approach
Basic
Combination
Black Box
White Box
Portfolio
Selection
Cooperative
Conceptual Integration