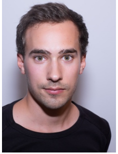
Reducer-Based Construction of Conditional Verifiers

Marie-Christine Jakobs

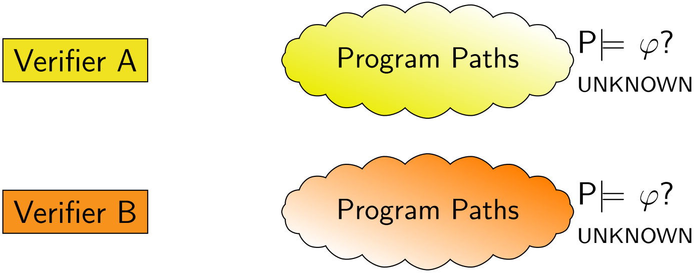
Joint work with Dirk Beyer, Thomas Lemberger, and Heike Wehrheim

LMU Munich, Germany



Facing Hard Verification Tasks

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



Verifier A

Program Paths

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

UNKNOWN

Verifier B

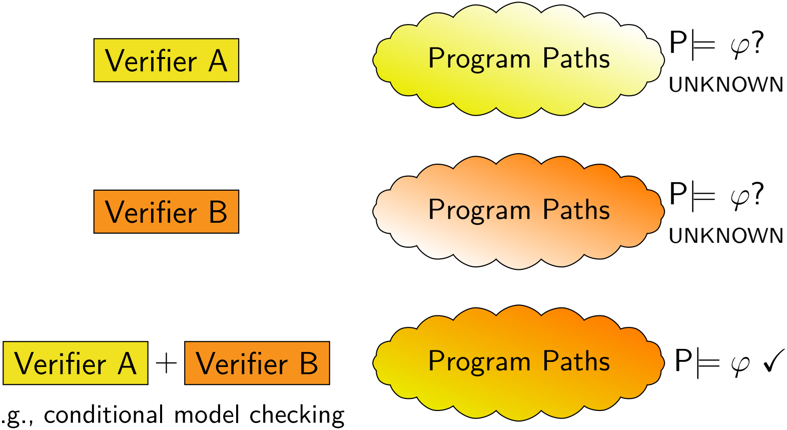
Program Paths

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

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

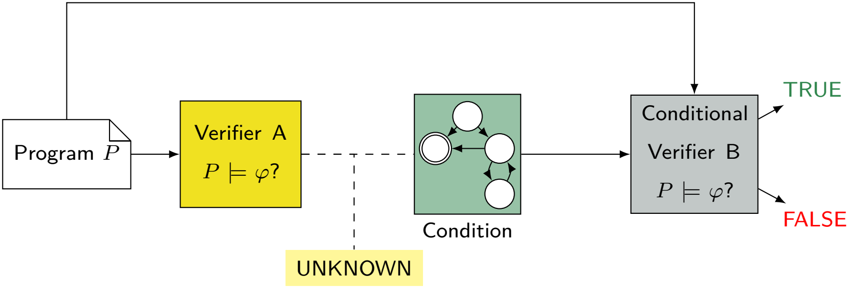
e.g., conditional model checking

Program Paths

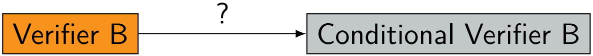
P$\models \varphi$ ✓

Conditional Model Checking

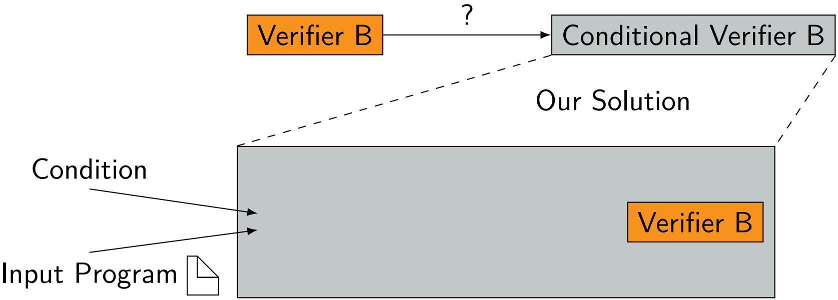
[Beyer/Henzinger/Keremoglu/Wendler FSE'12]



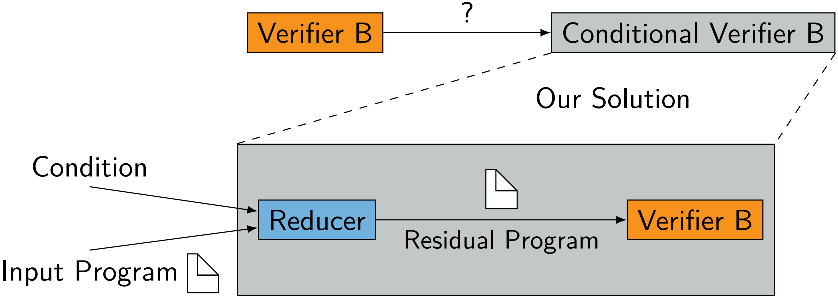
Reducer-Based Conditional Verifier Construction



Reducer-Based Conditional Verifier Construction



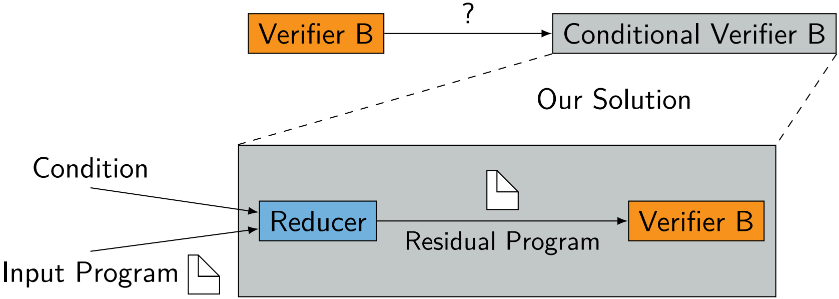
Reducer-Based Conditional Verifier Construction



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)

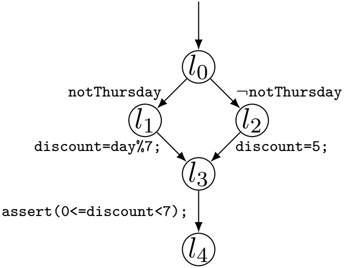
- Builds standard input (C program)
- Representing a subset of paths
- Contains at least all non-verified paths

+ Verifier-unspecific approach
+ Many conditional verifiers possible

Example Program and Condition

Program

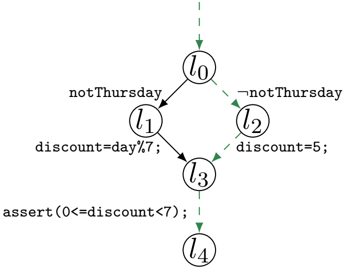
```
0: if(notThursday)
1:  discount=day%7;
   else
2:  discount=5;
3: assert(0<=discount<7);
4:
```



Example Program and Condition

Program

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

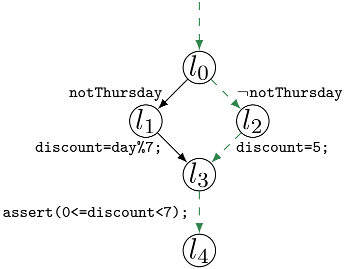


Verifier A only proofs else branch

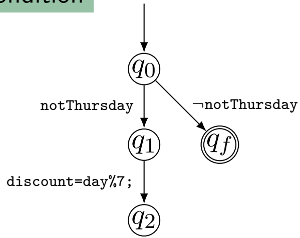
Example Program and Condition

Program

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

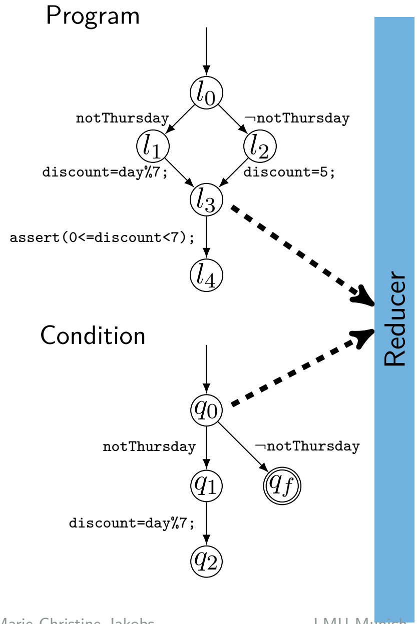


Condition



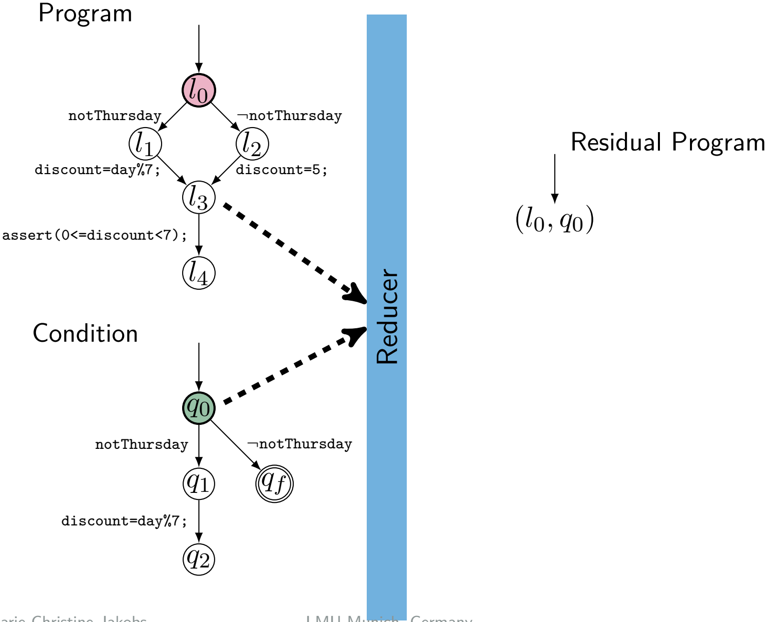
Verifier A only proofs else branch

Reducer: Residual Program Construction

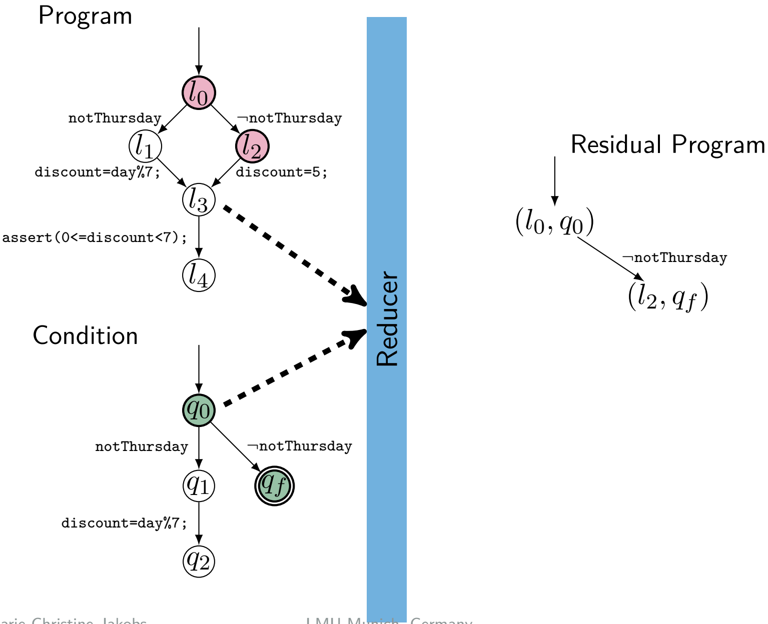


Residual Program

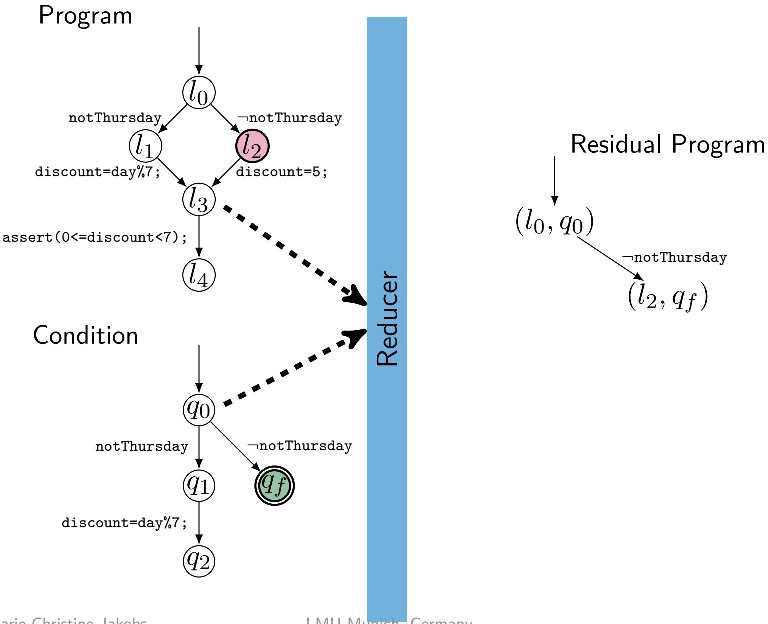
Reducer: Residual Program Construction



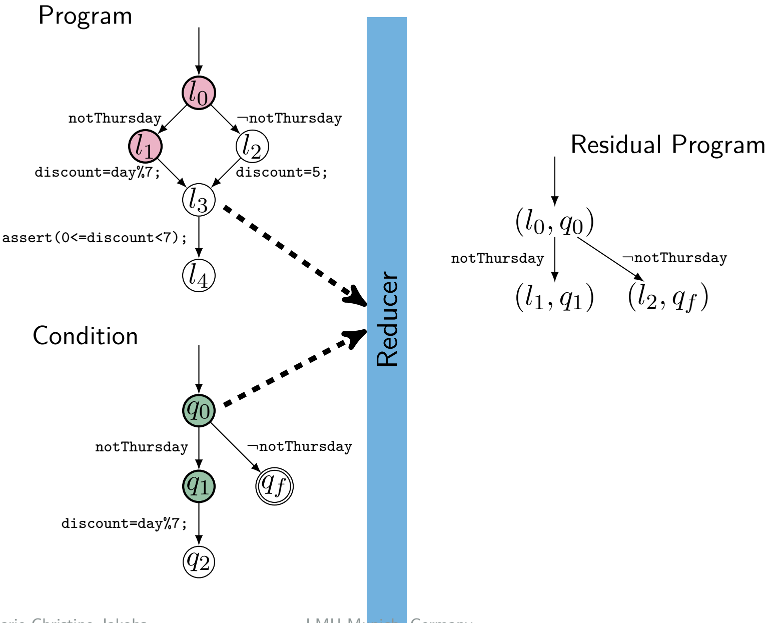
Reducer: Residual Program Construction



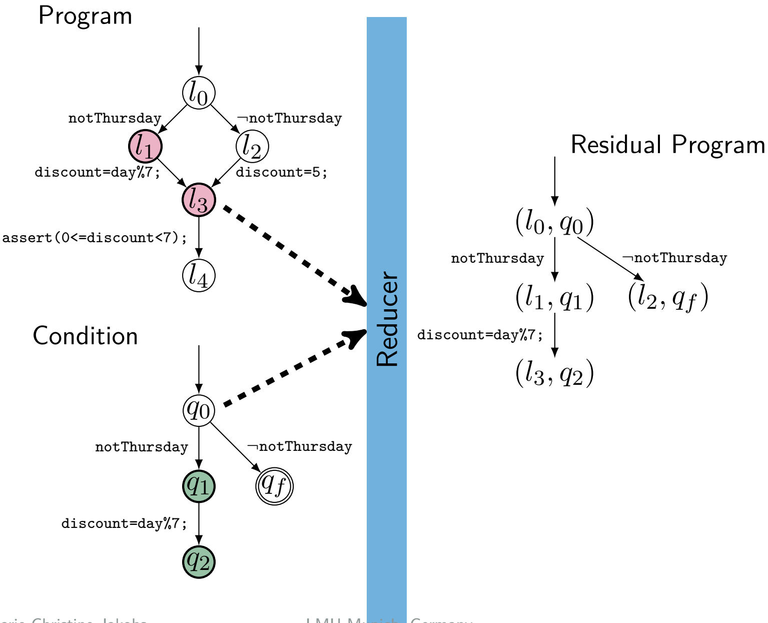
Reducer: Residual Program Construction



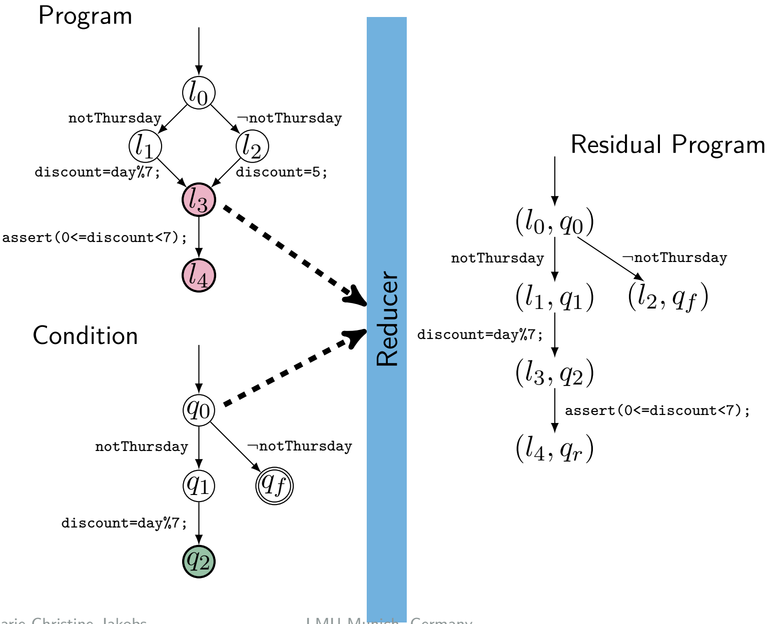
Reducer: Residual Program Construction



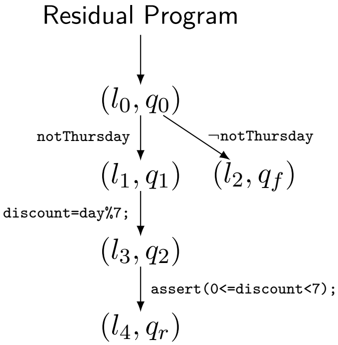
Reducer: Residual Program Construction



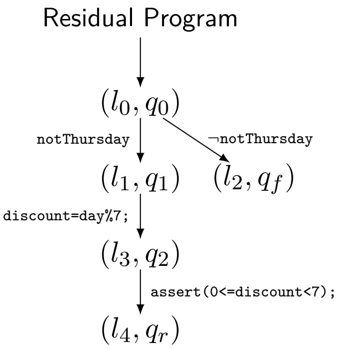
Reducer: Residual Program Construction



Reducer: C Transformation



Reducer: C Transformation

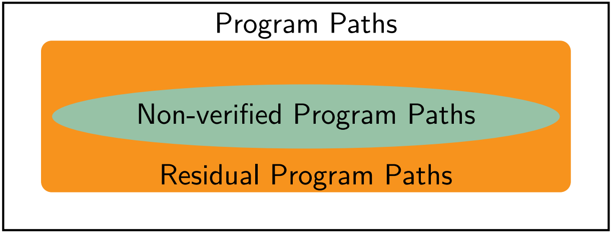


```
if(notThursday)
{
  discount=day%7;
  assert(0<=discount<7);
}
```

C Translation

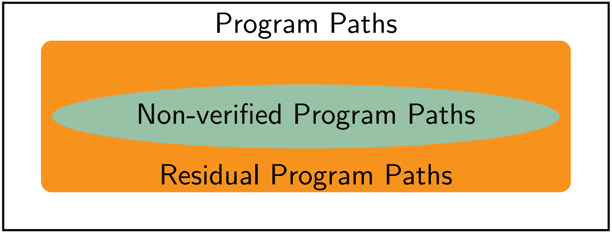
Reducer: Soundness

Residual Condition



Reducer: Soundness

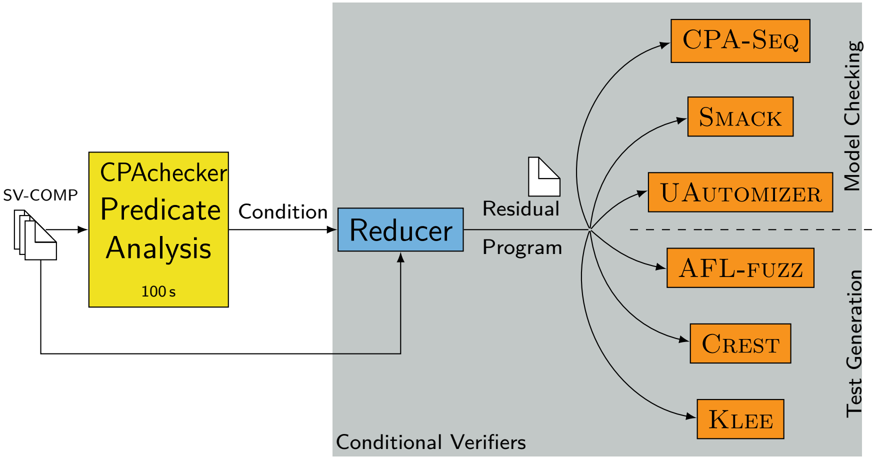
Residual Condition



Theorem

Presented reducer fulfills residual condition.

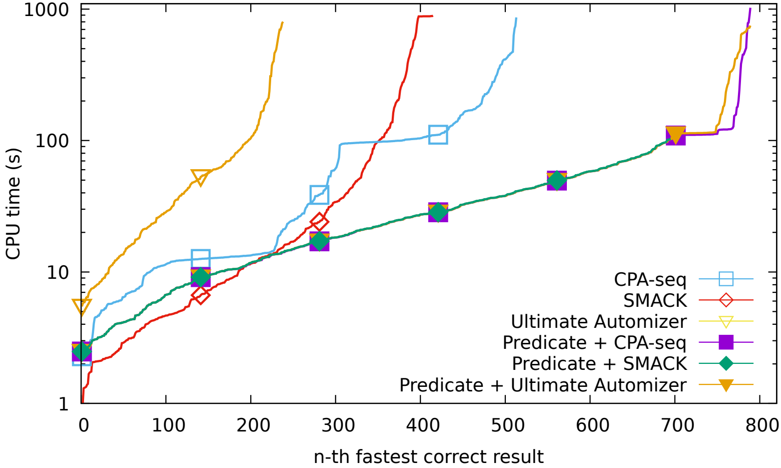
Evaluation Setup



Small Extract of Results

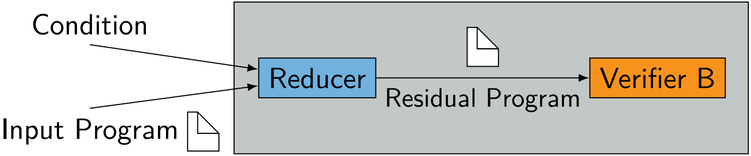
						Predicate		Predicate	
		CPA-Seq		UAutomizer		+Reducer		+Reducer	
						+CPA-Seq		+UAutomizer	
Task	R	S	t(s)	S	t(s)	S	t(s)	S	t(s)
P15l01	T	✗	910	✗	900	✓	120	✓	130
flood4	T	✗	910	✗	910	✓	450	✗	1100
newt3_6	F	✗	950	✗	490	✗	910	✓	260
P07l38	T	✗	950	✗	910	✗	1100	✓	470

Effectiveness on Hard Tasks



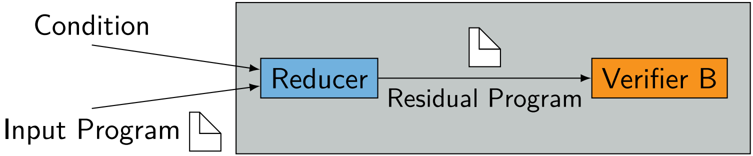
Conclusion

- Template-based conditional verifier construction



Conclusion

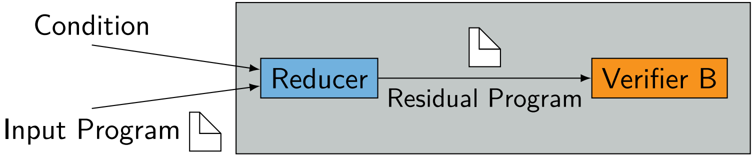
- Template-based conditional verifier construction



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

Conclusion

- Template-based conditional verifier construction

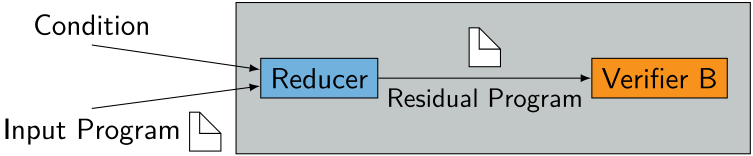


- One Reducer
  - Proven sound
  - Used in many conditional verifiers

- Effective on hard tasks for verifiers and test tools

Conclusion

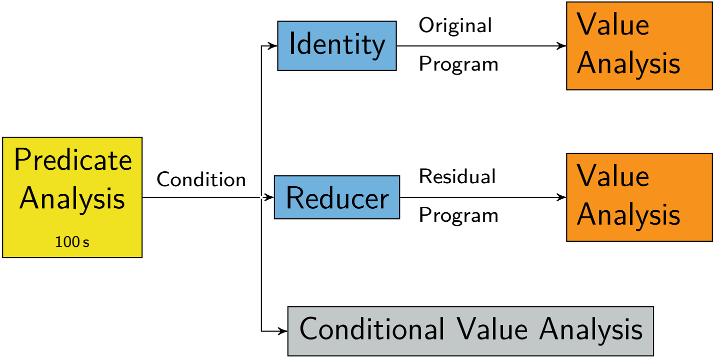
- Template-based conditional verifier construction



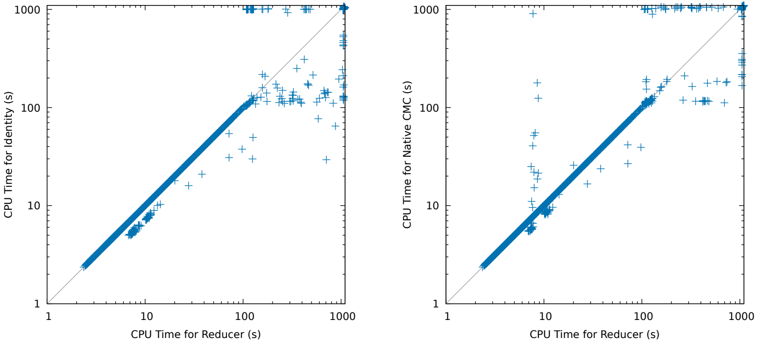
- One Reducer
  - Proven sound
  - Used in many conditional verifiers
- Effective on hard tasks for verifiers and test tools
- Future Work

  More reducers Using conditions from other tools

Comparison Setup



Comparison Results



(a) Identity vs.reducer

(b) Native vs. reducer-based

References I

D. Beyer, T. A. Henzinger, M. E. Keremoglu, and P. Wendler. Conditional Model Checking: A Technique to Pass Information Between Verifiers. In *Proc. FSE*.ACM, 2012.