Presence-Condition Simplification:

Problem, Solutions, Applications

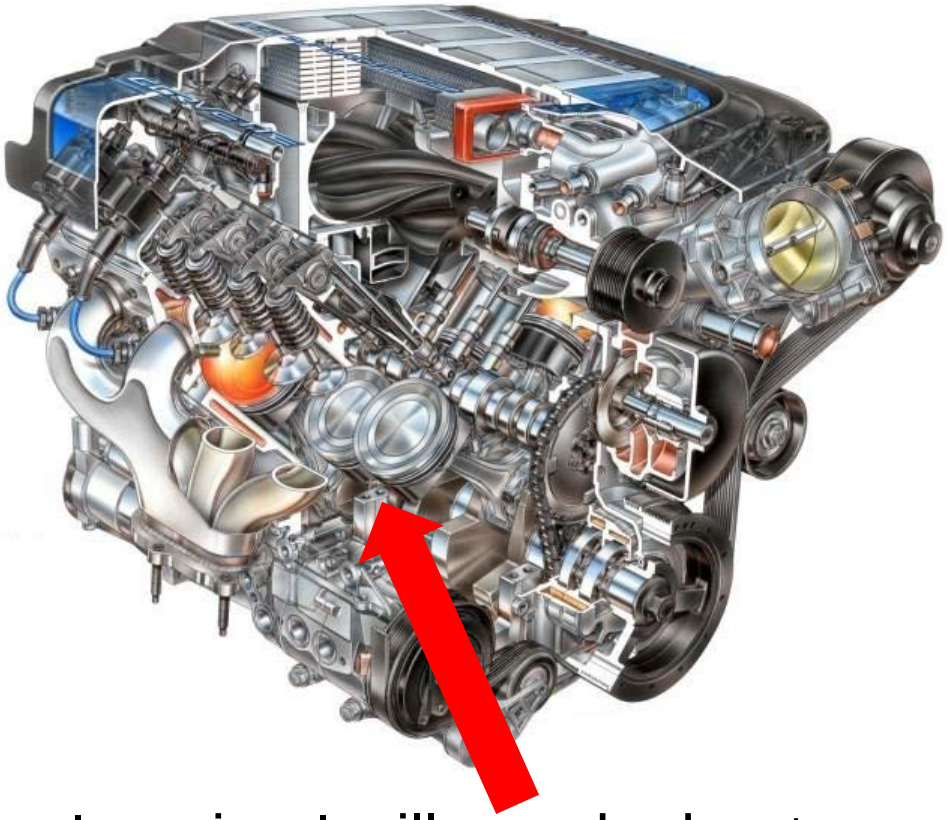
Alexander von Rhein, Alexander Grebhahn, Sven Apel,
Norbert Siegmund, Dirk Beyer, and Thorsten Berger

University of Passau, Germany

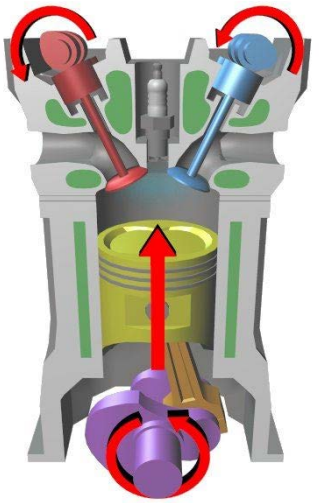
UNIVERSITÄT PASSAU

Software Product-Line Group

Simplification



- Imagine I will speak about the pistons in a diesel engine.



+
Thermodynamics +
Friction +
Turbocharging +
...

- This picture would be much more focused.

Presence Conditions

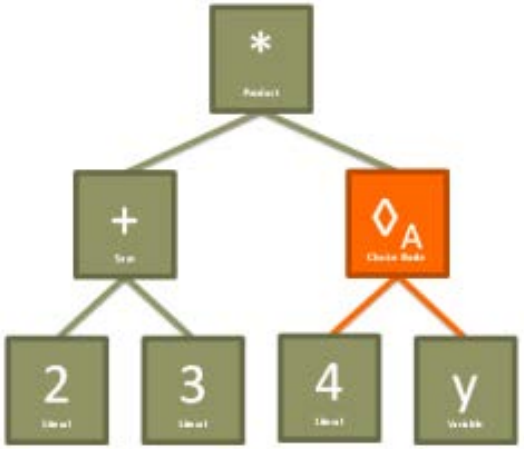
- In analysis reports
  - Verification
  - Type checking
  - Dataflow analysis

```
Specification 11 violated on condition
  encrypt && decrypt && keys &&
  ((sign && verify && base && autoresponder) ||
  (!sign && !verify && base && autoresponder))
```

- In source code
  - #ifdefs

```
1 #if HCF_ASSERT
2 ...
3 #if (HCF_ASSERT) & (HCF_ASSERT_LNK_MSF_RTN |
4   HCF_ASSERT_RT_MSF_RTN)
5   MSF_ASSERT_RTNP IFB_AssertRtn;
6 #endif
7 ...
8 #endif
```

- In internal code representations
  - Variability-aware AST in TypeChef



Presence-Condition Simplification

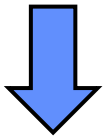
Presence Condition

Specification 11 violated on condition
encrypt && decrypt && keys &&
((sign && verify && base && autoresponder) ||
(!sign && !verify && base && autoresponder))

Context

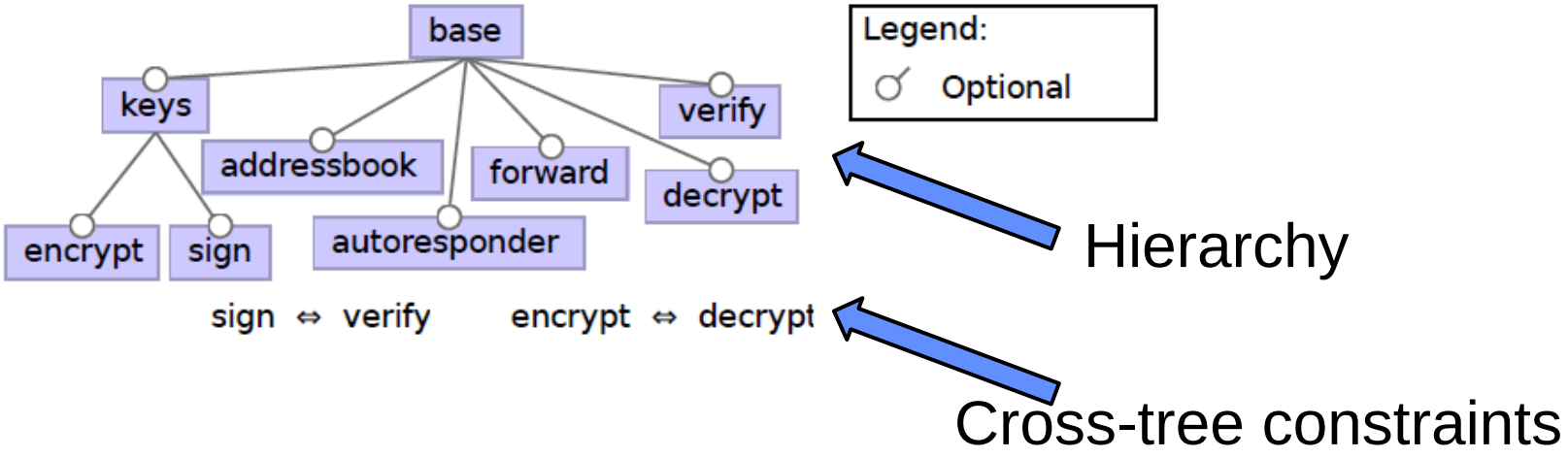
VariabilityModel =
base
Λ(decrypt ⇔ encrypt)Λ(sign ⇔ verify)
Λ(encrypt ⇒ keys)Λ(sign ⇒ keys)

simp (Presence Condition, Context)



Specification 11 violated on condition
VariabilityModel && (encrypt && autoresponder)

Scenario 2: Variability Model Synthesis



- VM synthesis [She, ICSE11] generates models from expressions
- Problem: eliminating redundant facts from CTC
- Solution:
  - Presence condition := cross-tree constraints
  - Context := hierarchy constraints

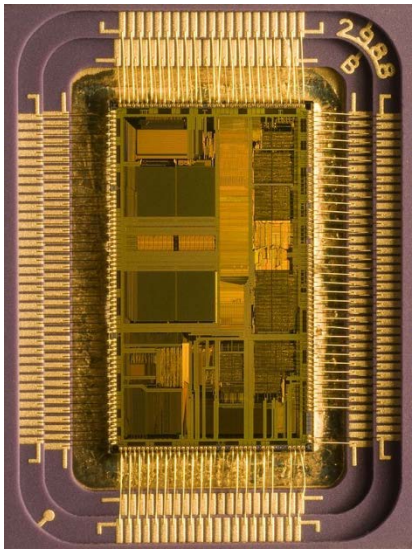
Formal Problem Definition

- Given a presence condition p and its context m:
  (both given as boolean expressions)
- we seek $\mathrm{x} = \mathrm{simp}(p, m)$ such that
  1. $m \rightarrow (\mathrm{x} \equiv p)$
     If m holds, we can replace p by x.

  2. x should be “smaller” than p.
     We define the size of an expression as the number of its operators.

Solutions

- RESTRICT (Coudert & Madre, 1989)
  - Based on binary decision diagrams
  - Heuristic to minimize the node count in the BDD
  - DAG traversal / comparison

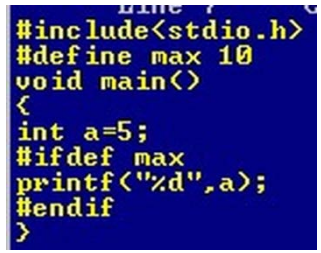


- Two-level logic minimization (1980s)
  - Input: an expression and a ***don't care set*** DC
  - DC states for which variable assignments we don't care about the value of x
  - Our don't care set is $\neg m$
  - QUINE-MCCLUSKEY (1956)
  - ESPRESSO (1986)

Experiments

- E1 “Classification of Variants”
  - Presence conditions from Norbert’s ICSE12 paper
- E2 “Defect-Location Reporting”
  - Presence conditions from Sven’s ICSE13 paper

- E3 “Code Simplification”
  - 21 #ifdef projects
  - Almost no condition-simplification potential

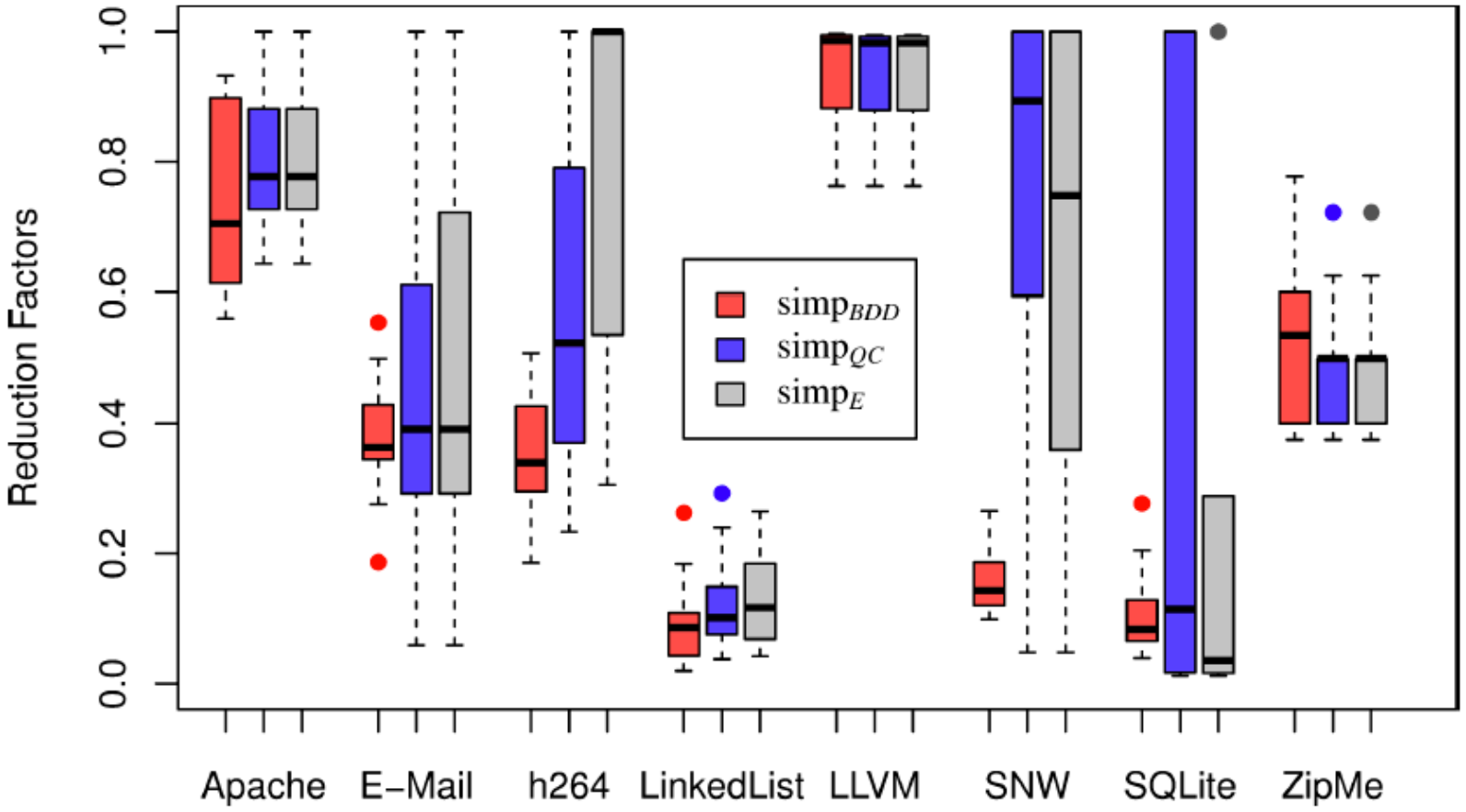


- E4 “AST Simplification”
  - Presence conditions from TYPECHEF AST for Linux kernel
- E5 “Scaling” with VM synthesis
  - Generated variability models from SPLOT
  - Models with 20 to 90 options

E1 “Classification of Variants”

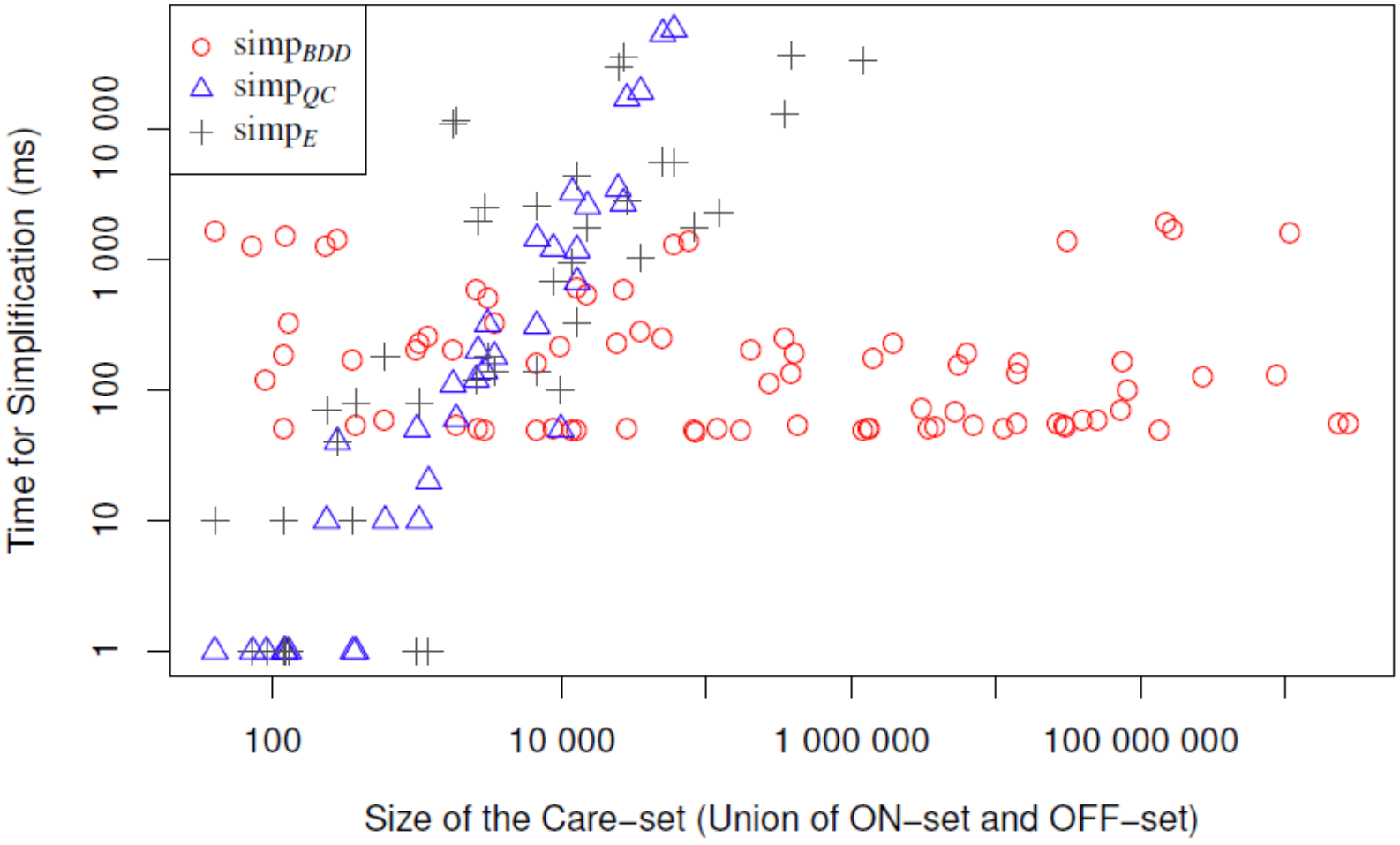
$$\text{ReductionFactor} = \frac{\text{size}(\text{simp}(p,m))}{\text{size}(\text{p})}$$

lower is better



E5 “Scaling”

Care-set: DNF-clauses needed to express $p \wedge m$ and $\neg p \wedge m$
Can increase with larger variability models



Conclusion

- Presence-condition simplification
  - has many application scenarios
    (reporting, code conditions, …)
  - is effective
    (can make conditions much smaller)
  - is efficient
    (fast as long as conditions/contexts can be expressed in BDDs)

- Future work
  - Other application scenarios
  - Specialized algorithms?

Christian's Application? "Order of Feature Interactions"

Presence Condition

Bug-triggering condition (e.g. Linux)

```
Specification 11 violated on condition
  encrypt && decrypt && keys &&
  ((sign && verify && base && autoresponder) ||
  (!sign && !verify && base && autoresponder))
```

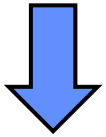
Context

Variability model ∧ other global conditions

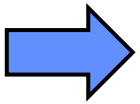
VariabilityModel =
base
∧(decrypt ⇔ encrypt)∧(sign ⇔ verify)
∧(encrypt ⇒ keys)∧(sign ⇒ keys)

simp (Presence Condition, Context)

Heuristics!



```
Specification 11 violated on condition
  VariabilityModel && (encrypt && autoresponder)
```



count variables in simplified condition
2 variables left -> 2-way interaction