[Jahrtausendturm Magdeburg]

Tseitin or not Tseitin?

The Impact of CNF Transformations on Feature-Model Analyses

SE 2023 (ASE 2022) — February 22–24 — Paderborn

Elias Kuiter[1], Sebastian Krieter[2], Chico Sundermann[2], Thomas Thüm[2], Gunter Saake[1]

University of Magdeburg[1], Ulm[2]

Implementing Configurable Software Systems

A Configurable Graph

```
class Node {
  #ifdef LABELED
    std::string label;
  #endif
  #ifdef COLORED
    std::string color;
  #endif
};

class Edge {
  #ifdef DIRECTED
    Node from, to;
  #elif UNDIRECTED && HYPER
    std::set<Node> nodes;
  #endif
};
```

Product Line Implementation
(here: C++ with C preprocessor)

Implementing Configurable Software Systems

A Configurable Graph

```
class Node {
  #ifdef LABELED
    std::string label;
  #endif
  #ifdef COLORED
    std::string color;
  #endif
};

class Edge {
  #ifdef DIRECTED
    Node from, to;
  #elif UNDIRECTED && HYPER
    std::set<Node> nodes;
  #endif
};
```

Product Line Implementation
(here: C++ with C preprocessor)

#define
LABELED
DIRECTED

Configuration

A Labeled Directed Graph

```
class Node {
  std::string label;
};

class Edge {
  Node from, to;
};
```

Product Implementation

Implementing Configurable Software Systems

A Configurable Graph

```
class Node {
  #ifdef LABELED
    std::string label;
  #endif
  #ifdef COLORED
    std::string color;
  #endif
};

class Edge {
  #ifdef DIRECTED
    Node from, to;
  #elif UNDIRECTED && HYPER
    std::set<Node> nodes;
  #endif
};
```

Product Line Implementation
(here: C++ with C preprocessor)

#define LABELED DIRECTED

#define COLORED UNDIRECTED HYPER

Configuration

A Labeled Directed Graph

```
class Node {
  std::string label;
};

class Edge {
  Node from, to;
};
```

A Colored Undirected Hypergraph

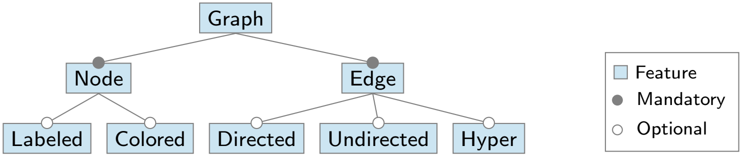
```
class Node {
  std::string color;
};

class Edge {
  std::set<Node> nodes;
};
```

Product Implementation

Modeling Features and their Dependencies

Feature Models

- tree models **features**
- cross-tree **constraints** model dependencies
- solver-based **analyses** can be used to understand the configuration space better



$\neg(\mathit{Directed} \land \mathit{Undirected})$ **feature conflict**

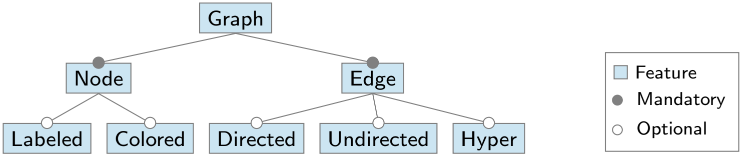
$\mathit{Hyper} \rightarrow \mathit{Undirected}$ **missing implementation**

$\mathit{Directed} \not\leftrightarrow (\mathit{Undirected} \land \mathit{Hyper})$ **well-defined #ifdef**

Modeling Features and their Dependencies

Feature Models

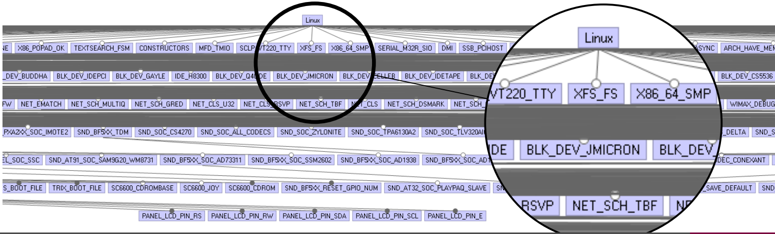
- tree models **features**
- cross-tree **constraints** model dependencies
- solver-based **analyses** can be used to understand the configuration space better



$\neg(\mathit{Directed} \wedge \mathit{Undirected})$	**feature conflict**
$\mathit{Hyper} \rightarrow \mathit{Undirected}$	**missing implementation**
$\mathit{Directed} \not\leftrightarrow (\mathit{Undirected} \wedge \mathit{Hyper})$	**well-defined #ifdef**

The Linux Kernel

- $>$ 12000 features [2016]
- $>10^{5000}$ products [2016]
- 114 dead features [2013]
- 151 reverse dependency bugs [2019]



Analyzing Feature Models with SAT and #SAT Solvers

Feature-Model Analysis

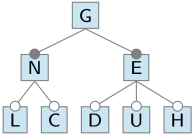
$$
\begin{tikzcd}
 & & \text{FM} \arrow[r, "\Phi"] & \text{Formula} \arrow[d, "\Theta"] \\
\text{Result} & \text{Query} \arrow[l, "\text{SAT}"', "\text{\#SAT}"] & & \text{CNF} \arrow[ll]
\end{tikzcd}
$$

Analyzing Feature Models with SAT and #SAT Solvers

Feature-Model Analysis

$$\begin{tikzcd} & \text{FM} \arrow[r, "\Phi"] & \text{Formula} \arrow[d, "\Theta"] \\ \text{Result} & \text{Query} \arrow[l, "\text{SAT}", "\#\text{SAT}"'] & \text{CNF} \arrow[l] \end{tikzcd}$$

A Feature Model *FM*



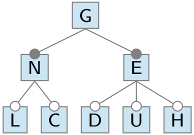
$$\neg(D \land U)$$
$$H \rightarrow U$$
$$D \not\leftrightarrow (U \land H)$$

Analyzing Feature Models with SAT and #SAT Solvers

Feature-Model Analysis

$$\begin{tikzcd} & & \text{FM} \arrow[r, "\Phi"] & \text{Formula} \arrow[d, "\Theta"] \\ \text{Result} & \text{Query} \arrow[l, "\text{SAT}", "\text{\#SAT}"'] & & \text{CNF} \arrow[ll] \end{tikzcd}$$

A Feature Model FM



$\neg(D \land U)$

$H \rightarrow U$

$D \nleftrightarrow (U \land H)$

$\xrightarrow{\Phi}$

As a Formula $\Phi(FM)$

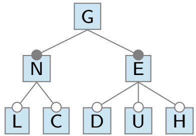
$$\begin{aligned} & G \\ \land\ & (N \leftrightarrow G) \land (E \leftrightarrow G) \\ \land\ & ((L \lor C) \rightarrow N) \\ \land\ & ((D \lor U \lor H) \rightarrow E) \\ \land\ & \neg(D \land U) \land (H \rightarrow U) \\ \land\ & (D \nleftrightarrow (U \land H)) \end{aligned}$$

Analyzing Feature Models with SAT and #SAT Solvers

Feature-Model Analysis

$$\begin{tikzcd} & \text{FM} \arrow[r, "\Phi"] & \text{Formula} \arrow[d, "\Theta"] \\ \text{Result} & \text{Query} \arrow[l, "\text{SAT}"', "\text{\#SAT}"] & \text{CNF} \arrow[l] \end{tikzcd}$$

A Feature Model FM



$\neg(D \wedge U)$

$H \rightarrow U$

$D \not\leftrightarrow (U \wedge H)$

$\xrightarrow{\Phi}$

As a Formula $\Phi(FM)$

$$\begin{aligned} & G \\ \wedge\ & (N \leftrightarrow G) \wedge (E \leftrightarrow G) \\ \wedge\ & ((L \vee C) \rightarrow N) \\ \wedge\ & ((D \vee U \vee H) \rightarrow E) \\ \wedge\ & \neg(D \wedge U) \wedge (H \rightarrow U) \\ \wedge\ & (D \not\leftrightarrow (U \wedge H)) \end{aligned}$$

$\downarrow \Theta$

As a CNF $\Theta(\Phi(FM))$

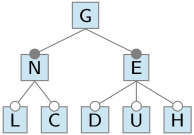
$$\begin{aligned} \{ & \{G\}, \{\neg N, G\}, \{N, \neg G\}, \\ & \{\neg E, G\}, \{E, \neg G\}, \{\neg L, N\}, \\ & \{\neg C, N\}, \{\neg D, E\}, \{\neg U, E\}, \\ & \{\neg H, E\}, \{\neg D, \neg U\}, \{\neg H, U\}, \\ & \{\{D, U\}, \{D, H\}, \{\neg D, \neg U, \neg H\}\}\} \end{aligned}$$

Analyzing Feature Models with SAT and #SAT Solvers

Feature-Model Analysis

$$\begin{tikzcd} & \text{FM} \arrow[r, "\Phi"] & \text{Formula} \arrow[d, "\Theta"] \\ \text{Result} & \text{Query} \arrow[l, "\text{SAT}"', "\text{\#SAT}"] & \text{CNF} \arrow[l] \end{tikzcd}$$

A Feature Model FM



$$\neg(D \land U)$$
$$H \rightarrow U$$
$$D \not\leftrightarrow (U \land H)$$

$\xrightarrow{\Phi}$

As a Formula $\Phi(FM)$

$$\begin{aligned} & G \\ \land\ & (N \leftrightarrow G) \land (E \leftrightarrow G) \\ \land\ & ((L \lor C) \rightarrow N) \\ \land\ & ((D \lor U \lor H) \rightarrow E) \\ \land\ & \neg(D \land U) \land (H \rightarrow U) \\ \land\ & (D \not\leftrightarrow (U \land H)) \end{aligned}$$

$\downarrow \Theta$

As a CNF $\Theta(\Phi(FM))$

$$\begin{aligned} \{&\{G\}, \{\neg N, G\}, \{N, \neg G\}, \\ &\{\neg E, G\}, \{E, \neg G\}, \{\neg L, N\}, \\ &\{\neg C, N\}, \{\neg D, E\}, \{\neg U, E\}, \\ &\{\neg H, E\}, \{\neg D, \neg U\}, \{\neg H, U\}, \\ &\{\{D, U\}, \{D, H\}, \{\neg D, \neg U, \neg H\}\}\} \end{aligned}$$

$\leftarrow$

Core Feature F?

$$SAT(\Theta(\Phi(FM)) \land \neg F)$$

$\leftarrow$

Core Features

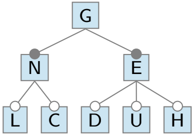
$$\{G, N, E\}$$

Analyzing Feature Models with SAT and #SAT Solvers

Feature-Model Analysis

$$\begin{array}{ccccc} & & \text{FM} & \xrightarrow{\Phi} & \text{Formula} \\ & & & & \downarrow \Theta \\ \text{Result} & \xleftarrow[\#SAT]{SAT} & \text{Query} & \leftarrow & \text{CNF} \end{array}$$

A Feature Model *FM*



$\neg(D \land U)$
$H \rightarrow U$
$D \not\leftrightarrow (U \land H)$

$\xrightarrow{\Phi}$

As a Formula $\Phi(FM)$

$$\begin{aligned} & G \\ \land\ & (N \leftrightarrow G) \land (E \leftrightarrow G) \\ \land\ & ((L \lor C) \rightarrow N) \\ \land\ & ((D \lor U \lor H) \rightarrow E) \\ \land\ & \neg(D \land U) \land (H \rightarrow U) \\ \land\ & (D \not\leftrightarrow (U \land H)) \end{aligned}$$

$\downarrow \Theta$

As a CNF $\Theta(\Phi(FM))$

$$\begin{aligned} \{ & \{G\}, \{\neg N, G\}, \{N, \neg G\}, \\ & \{\neg E, G\}, \{E, \neg G\}, \{\neg L, N\}, \\ & \{\neg C, N\}, \{\neg D, E\}, \{\neg U, E\}, \\ & \{\neg H, E\}, \{\neg D, \neg U\}, \{\neg H, U\}, \\ & \{\{D, U\}, \{D, H\}, \{\neg D, \neg U, \neg H\}\}\} \end{aligned}$$

$\leftarrow$

Core Feature *F*?

$SAT(\Theta(\Phi(FM)) \land \neg F)$

$\leftarrow$

Core Features

$\{G, N, E\}$

$\leftarrow$

Products in *FM*?

$\#SAT(\Theta(\Phi(FM)))$

$\leftarrow$

Feature Model Cardinality

8

Often Overlooked: Conjunctive Normal Form (CNF)

Feature-Model Analysis

$$
\begin{array}{ccc}
 & \text{FM} \xrightarrow{\Phi} & \textbf{Formula} \\
 & & \downarrow \Theta \\
\text{Result} \xleftarrow[\#\text{SAT}]{\text{SAT}} & \text{Query} \leftarrow & \textbf{CNF}
\end{array}
$$

From Formula . . .

$$
\begin{aligned}
& G \\
\wedge\ & (N \leftrightarrow G) \wedge (E \leftrightarrow G) \\
\wedge\ & ((L \vee C) \rightarrow N) \\
\wedge\ & ((D \vee U \vee H) \rightarrow E) \\
\wedge\ & \neg(D \wedge U) \wedge (H \rightarrow U) \\
\wedge\ & (D \not\leftrightarrow (U \wedge H))
\end{aligned}
$$

$\downarrow \Theta$

. . . to CNF

$$
\begin{aligned}
\{ & \{G\}, \{\neg N, G\}, \{N, \neg G\}, \\
& \{\neg E, G\}, \{E, \neg G\}, \{\neg L, N\}, \\
& \{\neg C, N\}, \{\neg D, E\}, \{\neg U, E\}, \\
& \{\neg H, E\}, \{\neg D, \neg U\}, \{\neg H, U\}, \\
& \{\{D, U\}, \{D, H\}, \{\neg D, \neg U, \neg H\}\}\}
\end{aligned}
$$

Often Overlooked: Conjunctive Normal Form (CNF)

Feature-Model Analysis

$$\begin{tikzcd} & \text{FM} \arrow[r, "\Phi"] & \textbf{Formula} \arrow[d, "\Theta"] \\ \text{Result} & \text{Query} \arrow[l, "\text{SAT}", "\#\text{SAT}"'] & \textbf{CNF} \arrow[l] \end{tikzcd}$$

Conjunctive Normal Form

- **conjunction** $\wedge$ of **disjunctions** $\vee$ of **literals** X, $\neg X$
- here: a set of **clauses**, which are sets of literals
- used by almost all solvers

From Formula . . .

$$\begin{aligned} & G \\ \wedge\ & (N \leftrightarrow G) \wedge (E \leftrightarrow G) \\ \wedge\ & ((L \vee C) \rightarrow N) \\ \wedge\ & ((D \vee U \vee H) \rightarrow E) \\ \wedge\ & \neg(D \wedge U) \wedge (H \rightarrow U) \\ \wedge\ & (D \not\leftrightarrow (U \wedge H)) \end{aligned}$$

$\downarrow \Theta$

. . . to CNF

$$\begin{aligned} \{ & \{G\}, \{\neg N, G\}, \{N, \neg G\}, \\ & \{\neg E, G\}, \{E, \neg G\}, \{\neg L, N\}, \\ & \{\neg C, N\}, \{\neg D, E\}, \{\neg U, E\}, \\ & \{\neg H, E\}, \{\neg D, \neg U\}, \{\neg H, U\}, \\ & \{\{D, U\}, \{D, H\}, \{\neg D, \neg U, \neg H\}\}\} \end{aligned}$$

Often Overlooked: Conjunctive Normal Form (CNF)

Feature-Model Analysis

$$\begin{tikzcd} & \text{FM} \arrow[r, "\Phi"] & \text{Formula} \arrow[d, "\Theta"] \\ \text{Result} & \text{Query} \arrow[l, "\text{SAT}", "\#\text{SAT}"'] & \text{CNF} \arrow[l] \end{tikzcd}$$

Conjunctive Normal Form

- **conjunction** $\land$ of **disjunctions** $\lor$ of **literals** X, $\neg X$
- here: a set of **clauses**, which are sets of literals
- used by almost all solvers

From Formula ...

$$\begin{aligned} & G \\ & \land (N \leftrightarrow G) \land (E \leftrightarrow G) \\ & \land ((L \lor C) \rightarrow N) \\ & \land ((D \lor U \lor H) \rightarrow E) \\ & \land \neg(D \land U) \land (H \rightarrow U) \\ & \land (D \not\leftrightarrow (U \land H)) \end{aligned}$$

$\downarrow \Theta$

... to CNF

$$\begin{aligned} & \{\{G\}, \{\neg N, G\}, \{N, \neg G\}, \\ & \{\neg E, G\}, \{E, \neg G\}, \{\neg L, N\}, \\ & \{\neg C, N\}, \{\neg D, E\}, \{\neg U, E\}, \\ & \{\neg H, E\}, \{\neg D, \neg U\}, \{\neg H, U\}, \\ & \{\{D, U\}, \{D, H\}, \{\neg D, \neg U, \neg H\}\}\} \end{aligned}$$

Our Goal: Raise Awareness for CNF Transformations [ASE'22]

- how to **transform** feature-model formulas **into CNF**?
  $\Rightarrow$ describe and classify CNF transformations
- does this **impact** the work of **practitioners and researchers**?
  $\Rightarrow$ evaluate efficiency and correctness on feature models

CNF Transformations

Distributive $\Theta = D$

apply laws of logic (**De Morgan's laws** and **distributivity**)

$D \not\leftrightarrow (U \wedge H)$

$\xrightarrow{D} (D \vee (U \wedge H)) \wedge (\neg D \vee \neg (U \wedge H))$

$\xrightarrow{D} \{\{D,U\},\{D,H\},\{\neg D,\neg U,\neg H\}\}$

- ✓ **equivalence**
  SAT ✓, $\#SAT = 4$
- ✓ easy to implement
- ✗ **exponential** complexity

CNF Transformations

Distributive $\Theta = D$

apply laws of logic (**De Morgan's laws** and **distributivity**)

$$D \not\leftrightarrow (U \land H)$$
$$\xrightarrow{D} (D \lor (U \land H)) \land (\neg D \lor \neg (U \land H))$$
$$\xrightarrow{D} \{\{D, U\}, \{D, H\}, \{\neg D, \neg U, \neg H\}\}$$

✓ **equivalence**
SAT ✓, $\#SAT = 4$

✓ easy to implement

✗ **exponential** complexity

Tseitin $\Theta = T$ ['83]

abbreviate a subformula ϕ with an auxiliary variable $x_\phi \leftrightarrow \phi$

$$D \not\leftrightarrow (U \land H)$$
$$\xrightarrow{T} (D \not\leftrightarrow x) \land x \leftrightarrow (U \land H)$$
$$\xrightarrow{D} \{\{D, x\}, \{\neg D, \neg x\}, \{\neg x, U\}, \{\neg x, H\}, \{\neg U, \neg H, x\}\}$$

✓ **quasi-equivalence**
SAT ✓, $\#SAT = 4$

✓ **linear** complexity

✗ take care of new variables

CNF Transformations

Distributive $\Theta = D$

apply laws of logic (**De Morgan's laws** and **distributivity**)

$$D \not\leftrightarrow (U \land H)$$
$$\xrightarrow{D} (D \lor (U \land H)) \land (\neg D \lor \neg(U \land H))$$
$$\xrightarrow{D} \{\{D, U\}, \{D, H\}, \{\neg D, \neg U, \neg H\}\}$$

- ✓ **equivalence** SAT ✓, $\#SAT = 4$
- ✓ easy to implement
- ✗ **exponential** complexity

Tseitin $\Theta = T$ ['83]

abbreviate a subformula ϕ with an auxiliary variable $x_\phi \leftrightarrow \phi$

$$D \not\leftrightarrow (U \land H)$$
$$\xrightarrow{T} (D \not\leftrightarrow x) \land x \leftrightarrow (U \land H)$$
$$\xrightarrow{D} \{\{D, x\}, \{\neg D, \neg x\}, \{\neg x, U\}, \{\neg x, H\}, \{\neg U, \neg H, x\}\}$$

- ✓ **quasi-equivalence** SAT ✓, $\#SAT = 4$
- ✓ **linear** complexity
- ✗ take care of new variables

Plaisted-Greenbaum $\Theta = PG$ ['86]

abbreviate a subformula ϕ with an auxiliary variable $x_\phi \rightarrow \phi$

$$D \not\leftrightarrow (U \land H)$$
$$\xrightarrow{PG} (D \not\leftrightarrow x) \land x \rightarrow (U \land H)$$
$$\xrightarrow{D} \{\{D, x\}, \{\neg D, \neg x\}, \{\neg x, U\}, \{\neg x, H\}\}$$

- ✓ **equi-assignability** SAT ✓
- ✓ **linear** complexity $< T$
- ✗ **equi-countability** $\#SAT = 5$

Evaluation

Research Questions

RQ 1 **efficiency** of CNF transformations?

RQ 2 CNF transformation $\rightarrow$ **efficiency** of analyses?

RQ 3 CNF transformation $\rightarrow$ **correctness** of analyses?

Evaluation

Research Questions

RQ 1 **efficiency** of CNF transformations?

RQ 2 CNF transformation → **efficiency** of analyses?

RQ 3 CNF transformation → **correctness** of analyses?

Experimental Setup

- 22 configurable software systems
- 3 CNF transformation tools
- 23 SAT and #SAT solvers

Evaluation

Research Questions

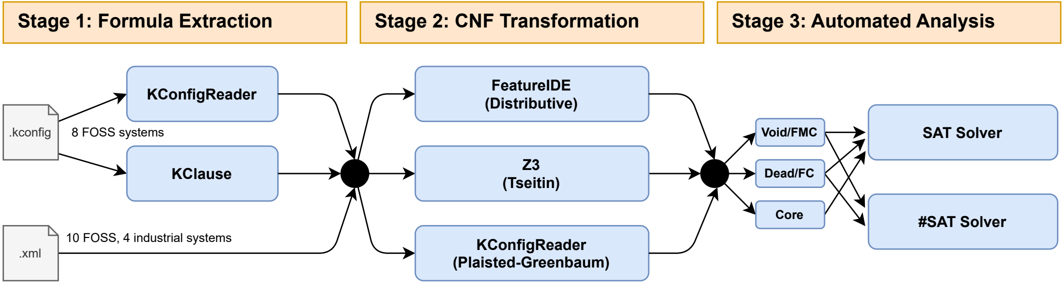
RQ 1 efficiency of CNF transformations?

RQ 2 CNF transformation → efficiency of analyses?

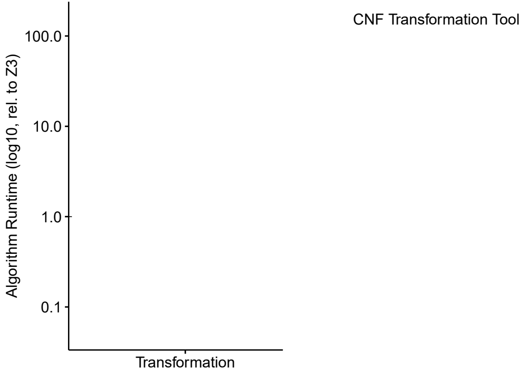
RQ 3 CNF transformation → correctness of analyses?

Experimental Setup

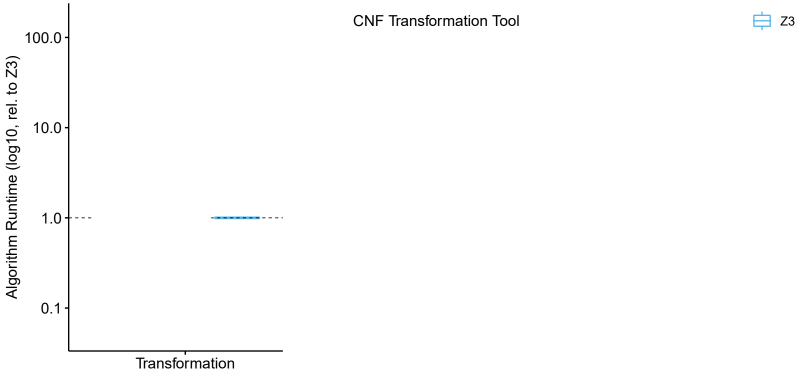
- 22 configurable software systems
- 3 CNF transformation tools
- 23 SAT and #SAT solvers



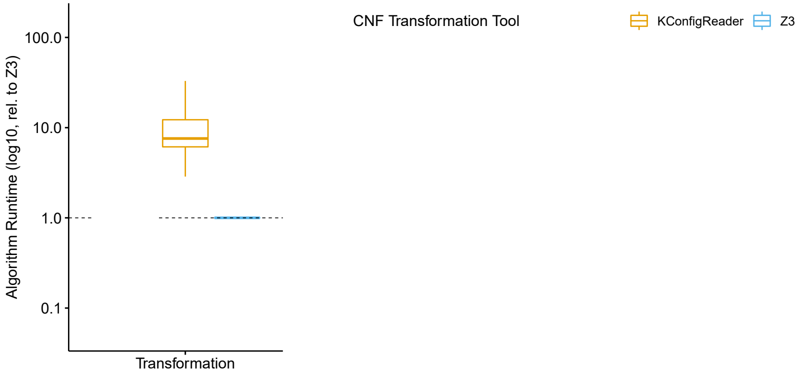
Efficiency of CNF Transformations (RQ 1) and Analyses (RQ 2)



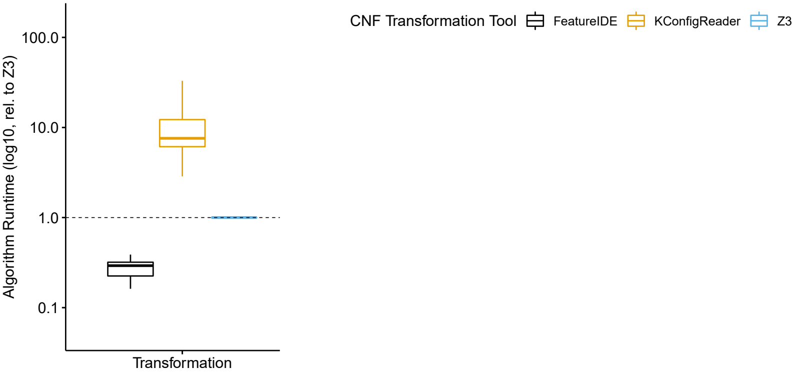
Efficiency of CNF Transformations (RQ 1) and Analyses (RQ 2)



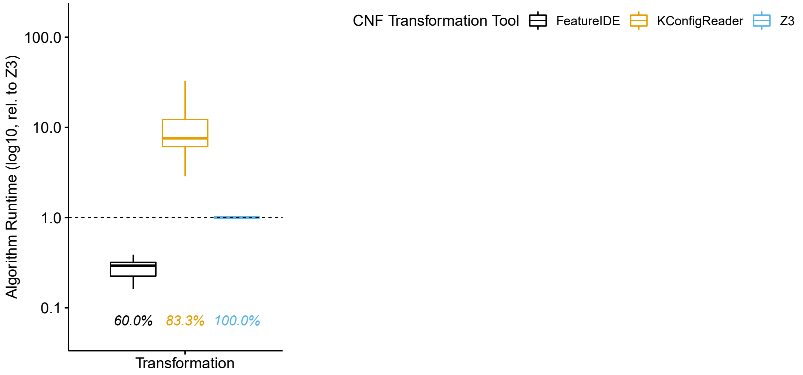
Efficiency of CNF Transformations (RQ 1) and Analyses (RQ 2)



Efficiency of CNF Transformations (RQ 1) and Analyses (RQ 2)

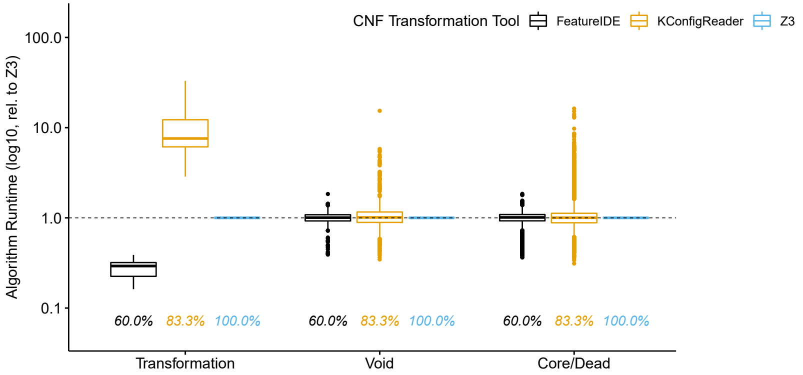


Efficiency of CNF Transformations (RQ 1) and Analyses (RQ 2)



RQ 1: *D* **often fails**, tools **differ significantly**

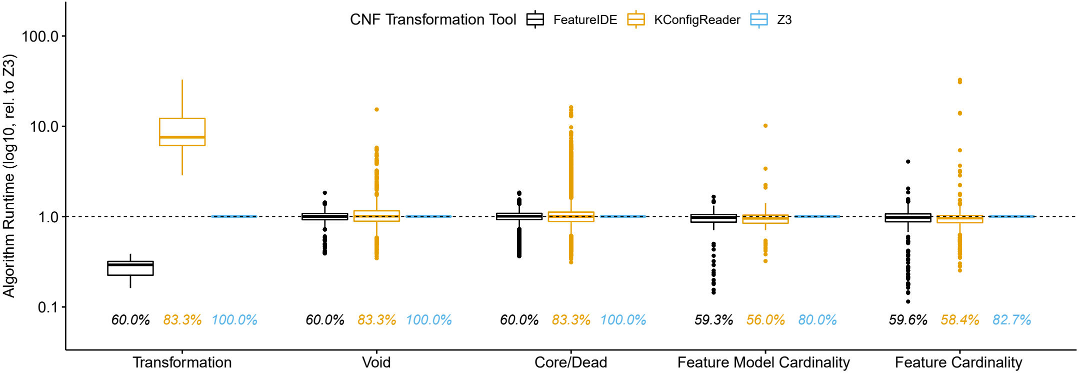
Efficiency of CNF Transformations (RQ 1) and Analyses (RQ 2)



RQ 1: *D* **often fails**,
tools **differ significantly**

RQ 2 (SAT): almost all calls succeed,
solve time varies by factor 0.31–16.27

Efficiency of CNF Transformations (RQ 1) and Analyses (RQ 2)



RQ 1: *D* **often fails**,
tools **differ significantly**

RQ 2 (SAT): almost all calls succeed,
solve time varies by factor 0.31–16.27

RQ 2 (#SAT): 81.6% of calls succeed,
solve time varies by factor 0.11–32.7

Correctness of #SAT-Based Analyses (RQ 3)

How Many Valid Configurations in BusyBox 1.35.0?

FeatureIDE (Distributive) says:

47842046044873008384 13517649496919484532 17980737275928522342 35800557238486733859 78971326945465595845 72908124465341304467 84732350200161989505 38440744692509401678 99136000000000000000 000000000

Tseitin (Z3) says:

47842046044873008384 13517649496919484532 17980737275928522342 35800557238486733859 78971326945465595845 72908124465341304467 84732350200161989505 38440744692509401678 99136000000000000000 000000000

KConfigReader (Plaisted-Greenbaum) says:

15751357446718468213 90135655996554596226 77965648288591932216 37368937605749145888 80850342078354075798 38471914912986177301 71318442740266744344 68038795993960163378 18607616000000000000 000000000 **0** $\Rightarrow$ **off by factor 3.292**

Correctness of #SAT-Based Analyses (RQ 3)

How Many Valid Configurations in BusyBox 1.35.0?

FeatureIDE (Distributive) says:

47842046044873008384 13517649496919484532 17980737275928522342 35800557238486733859
78971326945465595845 72908124465341304467 84732350200161989505 38440744692509401678
99136000000000000000 000000000

Tseitin (Z3) says:

47842046044873008384 13517649496919484532 17980737275928522342 35800557238486733859
78971326945465595845 72908124465341304467 84732350200161989505 38440744692509401678
99136000000000000000 000000000

KConfigReader (Plaisted-Greenbaum) says:

15751357446718468213 90135655996554596226 77965648288591932216 37368937605749145888
80850342078354075798 38471914912986177301 71318442740266744344 68038795993960163378
18607616000000000000 000000000 **0** $\Rightarrow$ **off by factor 3.292**

RQ 3

- with *PG*, $\approx$ 70% of #SAT calls return **incorrect results**
- incorrect by factor ≈ 3 (median)
- incorrect by factor $\approx 10^{77}$ (worst)

Correctness of #SAT-Based Analyses (RQ 3)

How Many Valid Configurations in BusyBox 1.35.0?

FeatureIDE (Distributive) says:

47842046044873008384 13517649496919484532 17980737275928522342 35800557238486733859 78971326945465595845 72908124465341304467 84732350200161989505 38440744692509401678 99136000000000000000 000000000

Tseitin (Z3) says:

47842046044873008384 13517649496919484532 17980737275928522342 35800557238486733859 78971326945465595845 72908124465341304467 84732350200161989505 38440744692509401678 99136000000000000000 000000000

KConfigReader (Plaisted-Greenbaum) says:

15751357446718468213 90135655996554596226 77965648288591932216 37368937605749145888 80850342078354075798 38471914912986177301 71318442740266744344 68038795993960163378 18607616000000000000 000000000 **0** $\Rightarrow$ **off by factor 3.292**

RQ 3

- with *PG*, $\approx$ 70% of #SAT calls return **incorrect results**
- incorrect by factor ≈ 3 (median)
- incorrect by factor $\approx 10^{77}$ (worst)

Our Recommendations

RQ 1 *D* for small, *T* for large models

RQ 2 largely depends on the model $\Rightarrow$ future work

RQ 3 do not use *PG* for #SAT

Conclusion

The Impact of CNF Transformations on Feature-Model Analyses

Distributive	Tseitin	Plaisted-Greenbaum
apply laws of logic	abbreviate ϕ with $x_\phi \leftrightarrow \phi$	abbreviate ϕ with $x_\phi \rightarrow \phi$
✓ **equivalence**	✓ **quasi-equivalence**	✓ **equi-assignability**
✓ easy to implement	✓ **linear** complexity	✓ **linear** complexity
× **exponential** complexity	× take care of new variables	× **equi-countability**

FeatureIDE	Z3	KConfigReader
often fails on large models	succeeds correctly on all models	often incorrect for #SAT calls

Conclusion

The Impact of CNF Transformations on Feature-Model Analyses

Distributive

apply laws of logic

- ✓ **equivalence**
- ✓ easy to implement
- ✗ **exponential** complexity

Tseitin

abbreviate ϕ with $x_\phi \leftrightarrow \phi$

- ✓ **quasi-equivalence**
- ✓ **linear** complexity
- ✗ take care of new variables

Plaisted-Greenbaum

abbreviate ϕ with $x_\phi \rightarrow \phi$

- ✓ **equi-assignability**
- ✓ **linear** complexity
- ✗ **equi-countability**

FeatureIDE

often fails on large models

Z3

succeeds correctly on all models

KConfigReader

often incorrect for #SAT calls

Tseitin or not Tseitin?

Conclusion

The Impact of CNF Transformations on Feature-Model Analyses

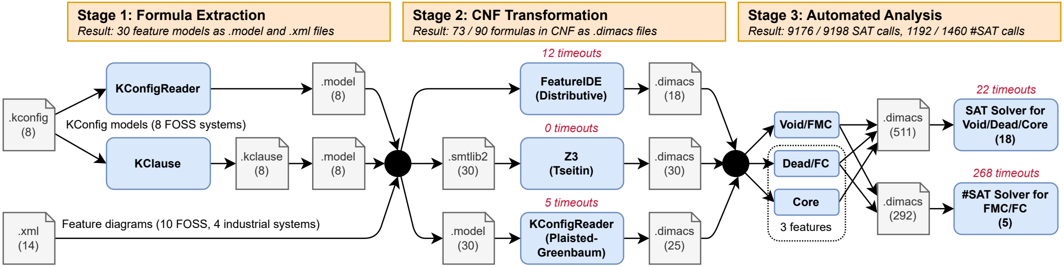
Distributive	Tseitin	Plaisted-Greenbaum
apply laws of logic	abbreviate ϕ with $x_\phi \leftrightarrow \phi$	abbreviate ϕ with $x_\phi \rightarrow \phi$
✓ **equivalence** ✓ easy to implement ✗ **exponential** complexity	✓ **quasi-equivalence** ✓ **linear** complexity ✗ take care of new variables	✓ **equi-assignability** ✓ **linear** complexity ✗ **equi-countability**

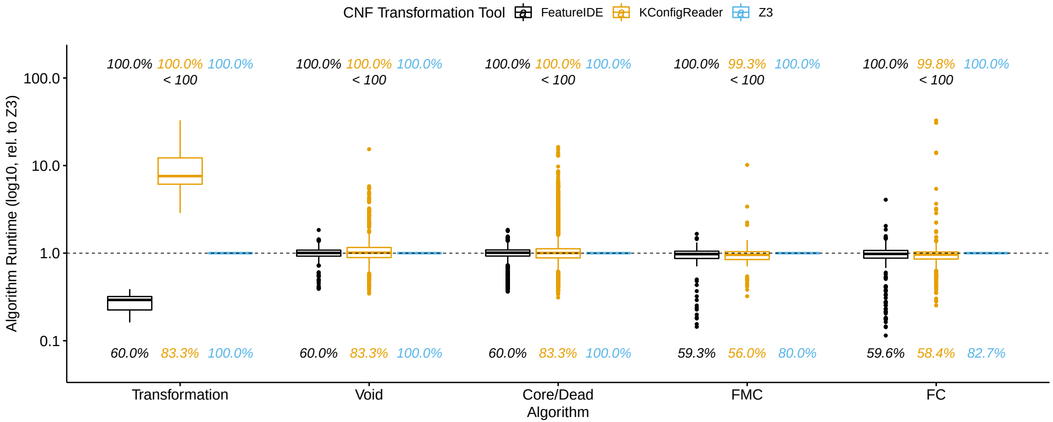
FeatureIDE	Z3	KConfigReader
often fails on large models	succeeds correctly on all models	often incorrect for #SAT calls

Tseitin or not Tseitin? $\Rightarrow$ Yes!

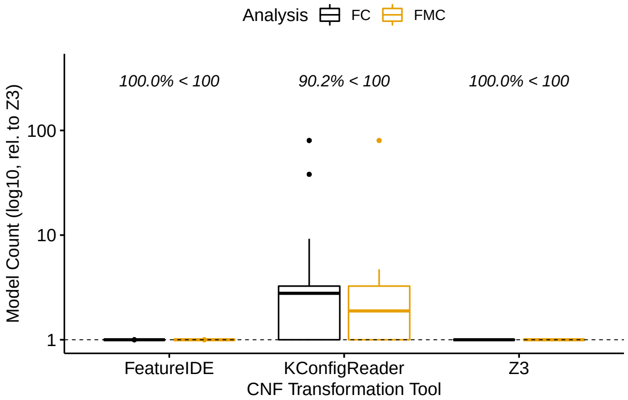
find out more:

github.com/ekuiter/tseitin-or-not-tseitin


Stage 1: Formula Extraction
Result: 30 feature models as .model and .xml files
Stage 2: CNF Transformation
Result: 73 / 90 formulas in CNF as .dimacs files
Stage 3: Automated Analysis
Result: 9176 / 9198 SAT calls, 1192 / 1460 #SAT calls
.kconfig (8)
KConfigReader
KClause
KConfig models (8 FOSS systems)
.model (8)
.kclause (8)
.model (8)
.xml (14)
Feature diagrams (10 FOSS, 4 industrial systems)
.smtlib2 (30)
.model (30)
12 timeouts
FeatureIDE (Distributive)
0 timeouts
Z3 (Tseitin)
5 timeouts
KConfigReader (Plaisted-Greenbaum)
.dimacs (18)
.dimacs (30)
.dimacs (25)
Void/FMC
Dead/FC
Core
3 features
.dimacs (511)
.dimacs (292)
22 timeouts
SAT Solver for Void/Dead/Core (18)
268 timeouts
#SAT Solver for FMC/FC (5)



CNF Transformation Tool
FeatureIDE
KConfigReader
Z3
Algorithm Runtime (log10, rel. to Z3)
100.0
10.0
1.0
0.1
100.0% 100.0% 100.0%
< 100
100.0% 100.0% 100.0%
< 100
100.0% 100.0% 100.0%
< 100
100.0% 99.3% 100.0%
< 100
100.0% 99.8% 100.0%
< 100
60.0% 83.3% 100.0%
60.0% 83.3% 100.0%
60.0% 83.3% 100.0%
59.3% 56.0% 80.0%
59.6% 58.4% 82.7%
Transformation
Void
Core/Dead
FMC
FC
Algorithm


Analysis
FC
FMC
100.0% < 100
90.2% < 100
100.0% < 100
Model Count (log10, rel. to Z3)
100
10
1
FeatureIDE
KConfigReader
Z3
CNF Transformation Tool