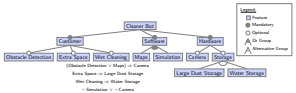
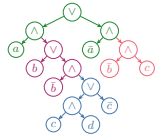


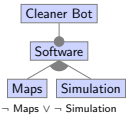
d-DNNF Compilation



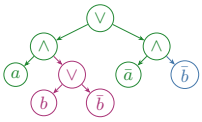
Slicing (Krieter 2016)

Projected d-DNNF Compilation

Slicing



d-DNNF Compilation (Darwiche 2002)



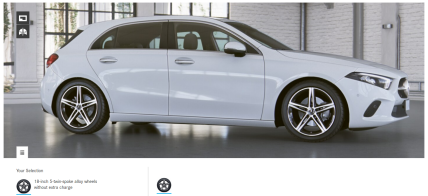
Efficient Slicing of Feature Models via Projected d-DNNF Compilation

ASE'24 Research Track | Chico Sundermann, Jacob Loth, Thomas Thüm | 2024/10/29

universität ulm

PADERBORN UNIVERSITY

Motivation Product Lines



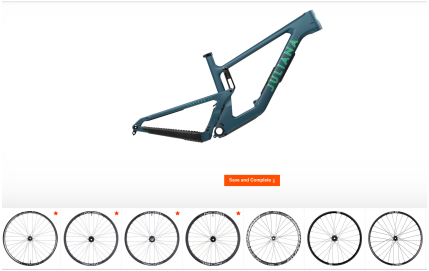
Motivation Product Lines

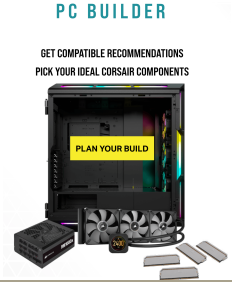
PC BUILDER

GET COMPATIBLE RECOMMENDATIONS

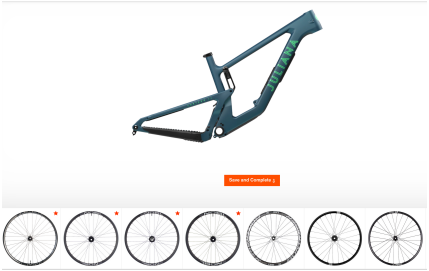
PICK YOUR IDEAL CORSAIR COMPONENTS

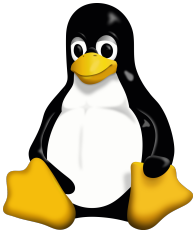
Motivation Product Lines



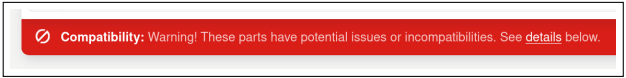


Motivation Product Lines

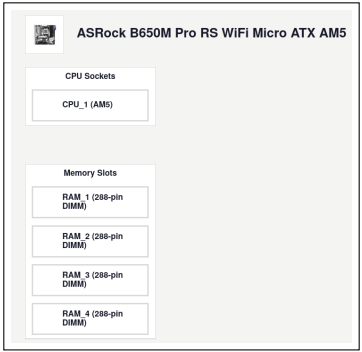




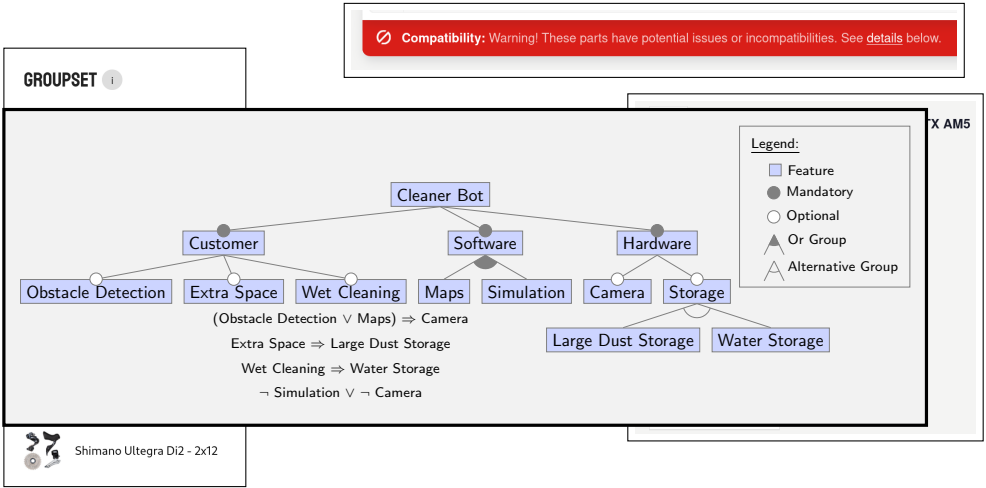
Motivation Feature Dependencies



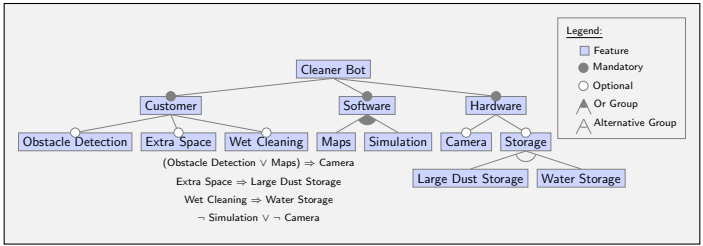

```
config SECURITY_INFINIBAND
            bool "Infiniband Security Hooks"
            depends on SECURITY && INFINIBAND
```



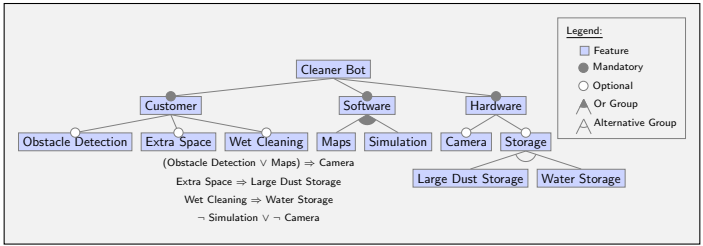
Motivation Feature Dependencies



Motivation Plenty Analyses

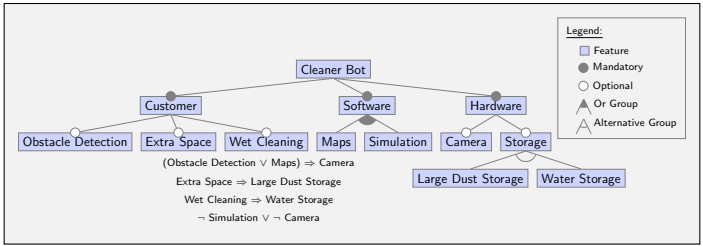


Motivation Plenty Analyses



Can we configure a robot?

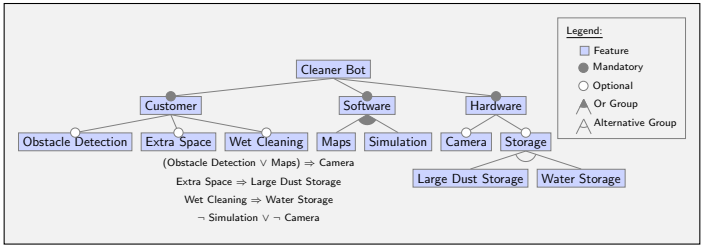
Motivation Plenty Analyses



Can we configure a robot?

Is there a robot with camera?

Motivation Plenty Analyses

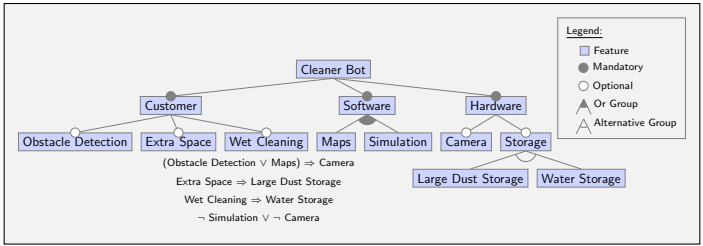


Can we configure a robot?

Is there a robot with camera?

How many robots?

Motivation Plenty Analyses



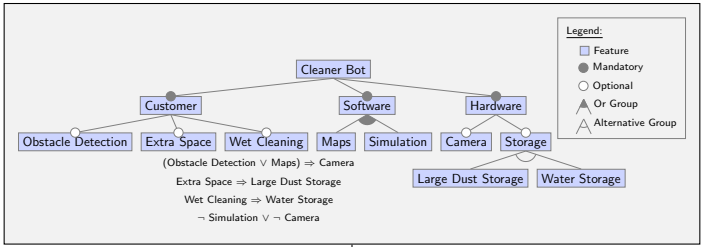
Can we configure a robot?

Is there a robot with camera?

How many robots?

Which robots to test?

Motivation Plenty Analyses



Customer ∧ Software ∧ (Maps ∨ Simulation) ∧ . . .

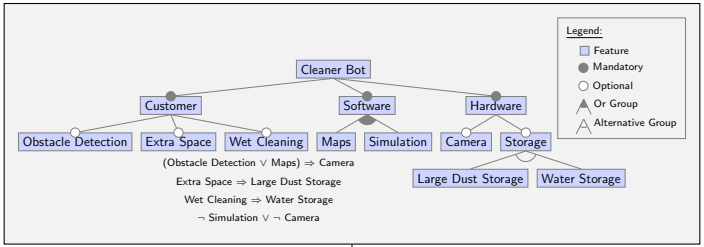
Can we configure a robot?

Is there a robot with camera?

How many robots?

Which robots to test?

Motivation Plenty Analyses



Can we configure a robot?

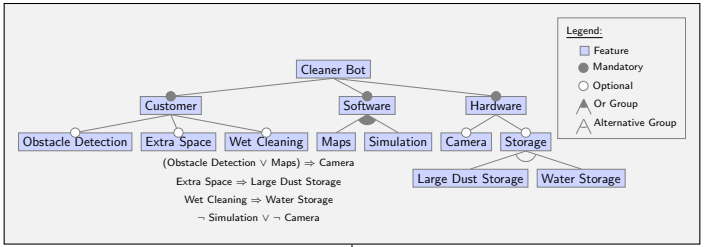
Is there a robot with camera?

How many robots?

Which robots to test?

Customer ∧ Software ∧ (Maps ∨ Simulation) ∧ . . .

Motivation Plenty Analyses



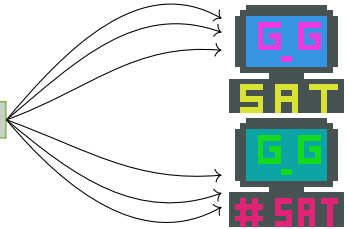
Can we configure a robot?

Is there a robot with camera?

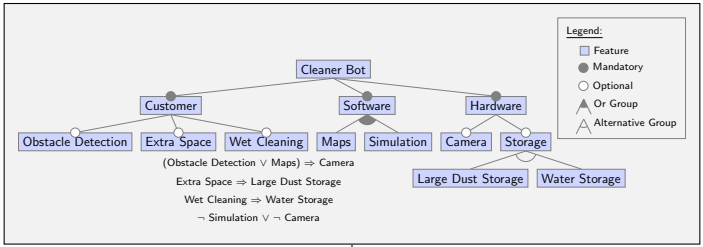
How many robots?

Which robots to test?

Customer ∧ Software ∧ (Maps ∨ Simulation) ∧ . . .

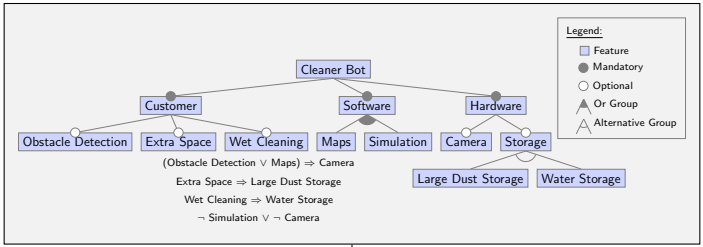


Motivation Knowledge Compilation



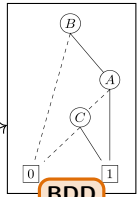
$$\text{Customer} \wedge \text{Software} \wedge (\text{Maps} \vee \text{Simulation}) \wedge \ldots$$

Motivation Knowledge Compilation

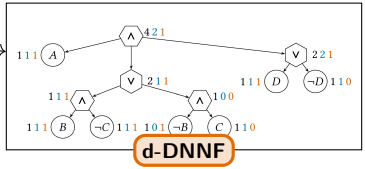


Customer ∧ Software ∧ (Maps ∨ Simulation) ∧ . . .

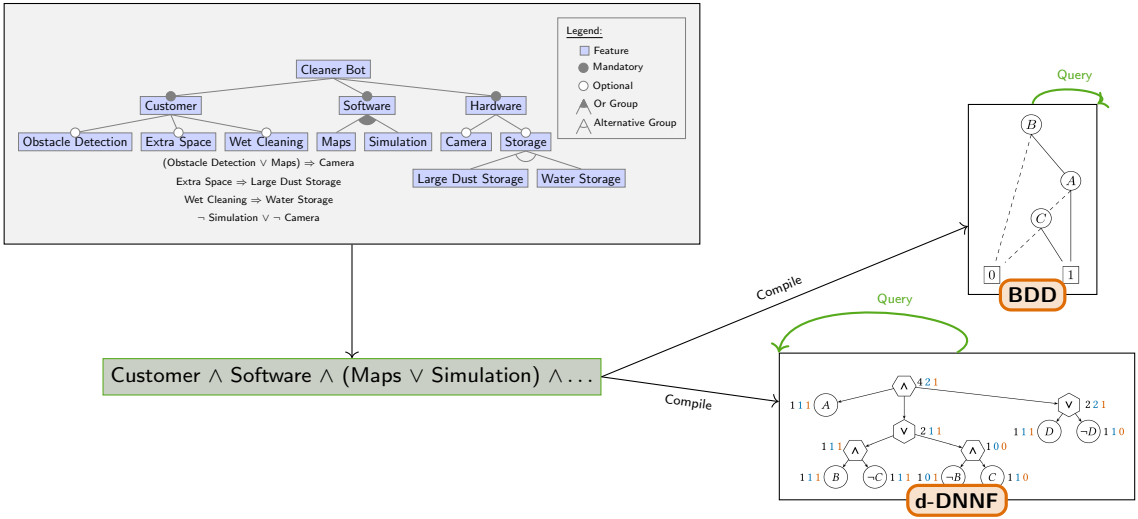
Compile



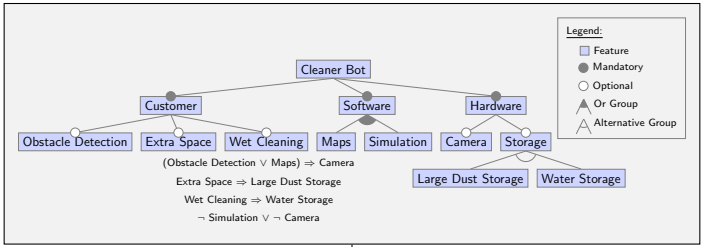
Compile



Motivation Knowledge Compilation

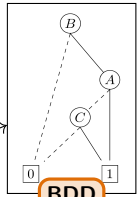


Motivation Knowledge Compilation

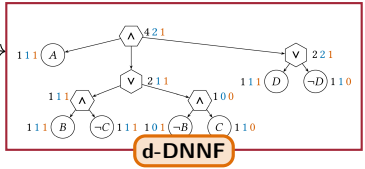


Customer ∧ Software ∧ (Maps ∨ Simulation) ∧ . . .

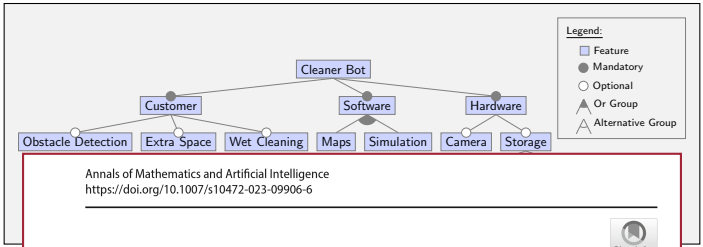
Compile



Compile



Motivation Knowledge Compilation



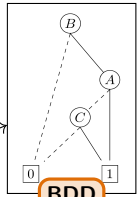
Annals of Mathematics and Artificial Intelligence
https://doi.org/10.1007/s10472-023-09906-6

Check for updates

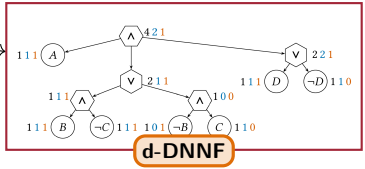
On the benefits of knowledge compilation for feature-model analyses

Chico Sundermann[1] · Elias Kuiter[2] · Tobias Heß[1] · Heiko Raab[1] · Sebastian Krieter[1] · Thomas Thüm[1]

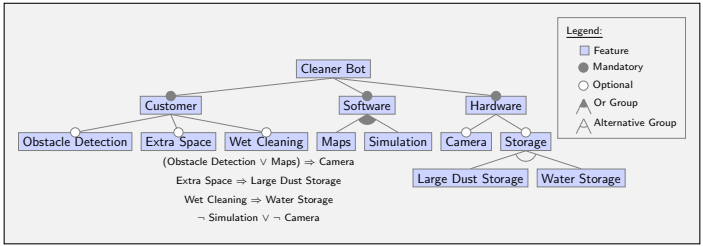
Compile



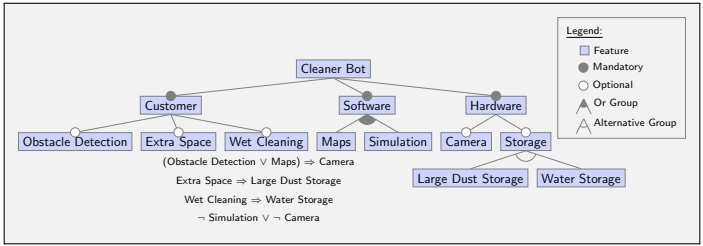
Compile

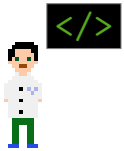


Decomposition for Stakeholders Slicing

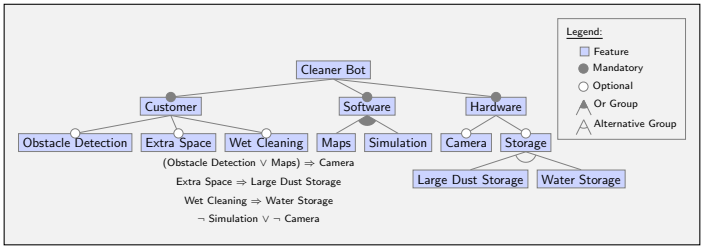


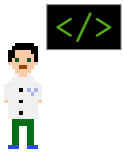
Decomposition for Stakeholders Slicing

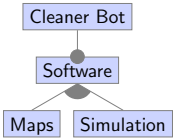




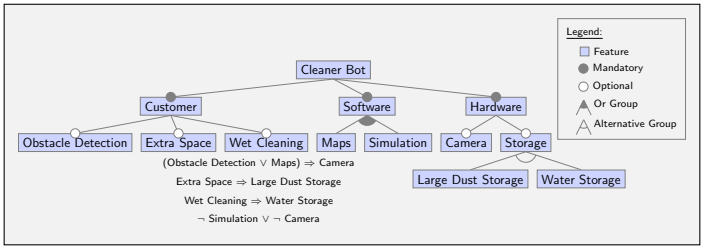
Decomposition for Stakeholders Slicing

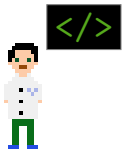


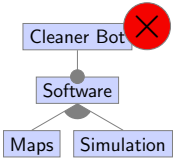


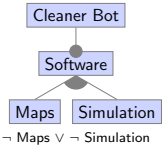


Decomposition for Stakeholders Slicing

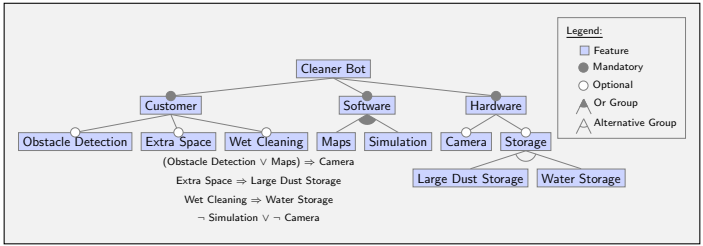








Decomposition for Stakeholders Slicing

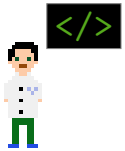


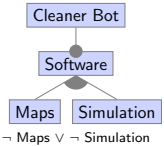
Is there a valid software?

How many software variants?

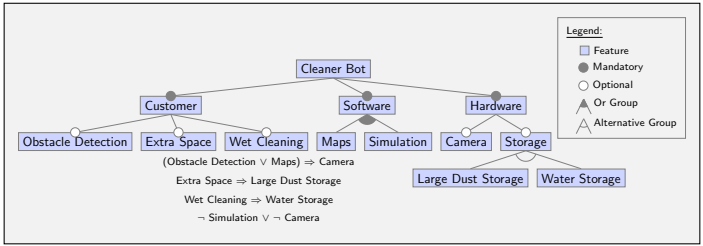
Is there a software with Maps?

Which software variants to test?





Decomposition for Stakeholders Slicing

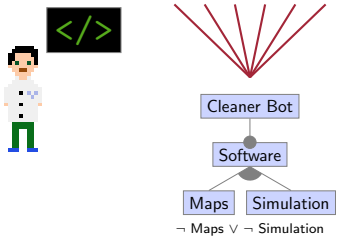


Is there a valid software?

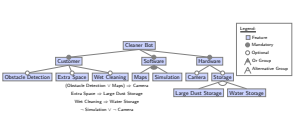
How many software variants?

Is there a software with Maps?

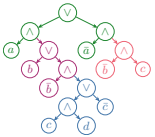
Which software variants to test?



Projected d-DNNF Compilation



d-DNNF Compilation

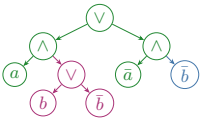


Slicing (Krieter 2016)

Slicing

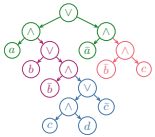
Cleaner Bot
Software
Maps
Simulation
¬ Maps ∨ ¬ Simulation

d-DNNF Compilation (Darwiche 2002)



Projected d-DNNF Compilation

d-DNNF Compilation



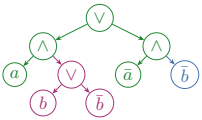
Slicing (Krieter 2016)

Projected d-DNNF Compilation

Slicing

Cleaner Bot
Software
Maps
Simulation
¬ Maps ∨ ¬ Simulation

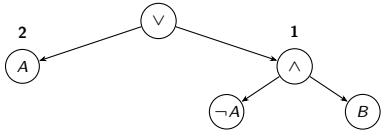
d-DNNF Compilation (Darwiche 2002)



How does this even work? d-DNNFs (Darwiche 2000)

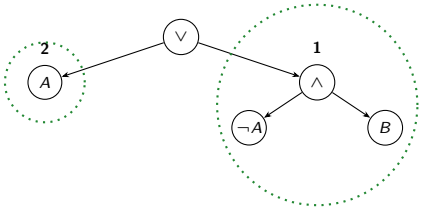
deterministic
decomposable
negation
normal
form

How does this even work? d-DNNFs (Darwiche 2000)



deterministic
decomposable
negation
normal
form

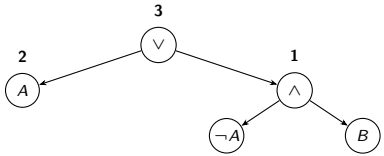
How does this even work? d-DNNFs (Darwiche 2000)



No shared solutions

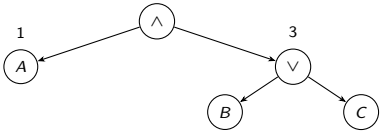
deterministic
decomposable
negation
normal
form

How does this even work? d-DNNFs (Darwiche 2000)



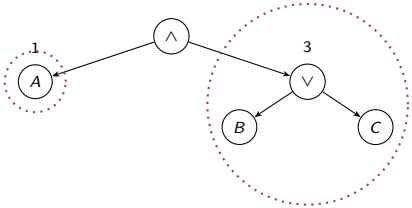
deterministic (Sum for ∨)
decomposable
negation
normal
form

How does this even work? d-DNNFs (Darwiche 2000)



deterministic (Sum for ∨)
decomposable
negation
normal
form

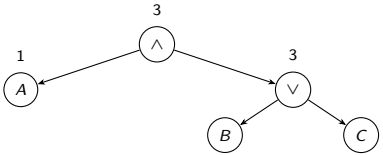
How does this even work? d-DNNFs (Darwiche 2000)



Independent
$\Rightarrow$ All pairs valid

deterministic (Sum for $\vee$)
decomposable
negation
normal
form

How does this even work? d-DNNFs (Darwiche 2000)

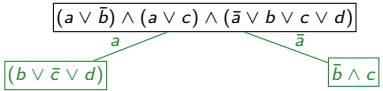


deterministic (Sum for $\vee$)
decomposable (Product for $\wedge$)
negation
normal
form

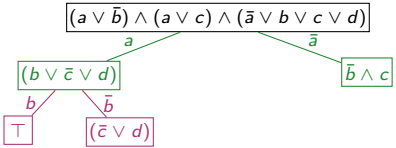
Regular d-DNNF Compilation

$$\boxed{(a \vee \bar{b}) \wedge (a \vee c) \wedge (\bar{a} \vee b \vee c \vee d)}$$

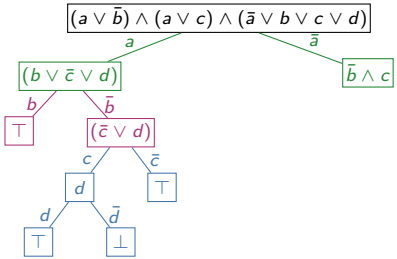
Regular d-DNNF Compilation



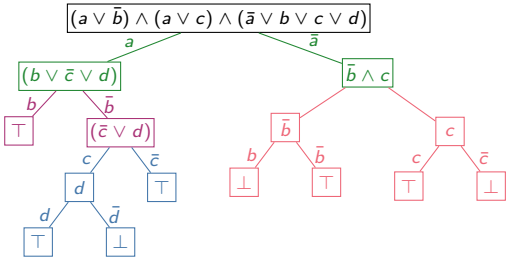
Regular d-DNNF Compilation



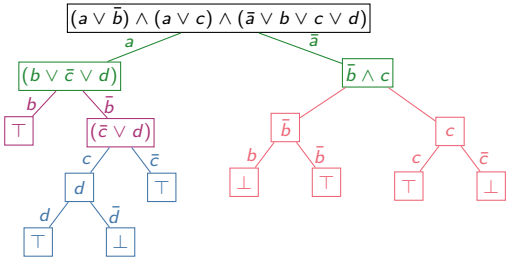
Regular d-DNNF Compilation

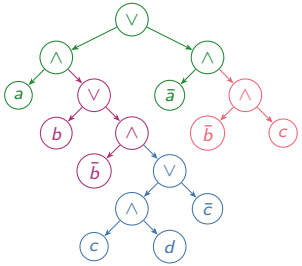


Regular d-DNNF Compilation



Regular d-DNNF Compilation





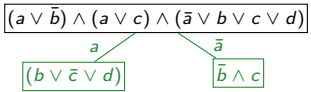
Projected d-DNNF Compilation

Slice c and d

$$(a \vee \bar{b}) \wedge (a \vee c) \wedge (\bar{a} \vee b \vee c \vee d)$$

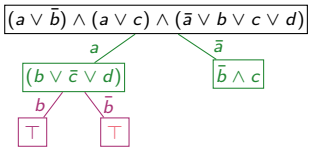
Projected d-DNNF Compilation

Slice c and d



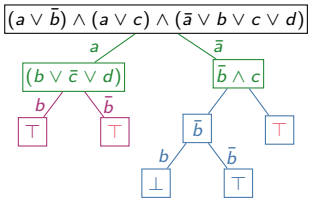
Projected d-DNNF Compilation

Slice c and d



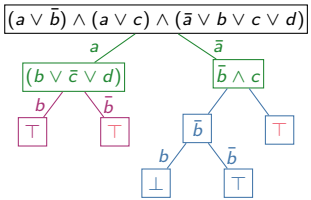
Projected d-DNNF Compilation

Slice c and d



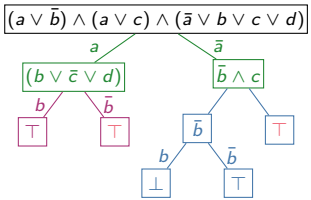
Projected d-DNNF Compilation

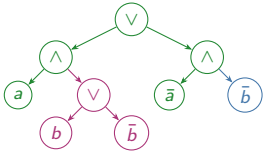
Slice c and d



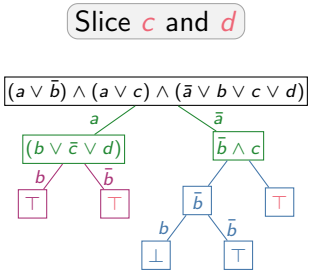
Projected d-DNNF Compilation

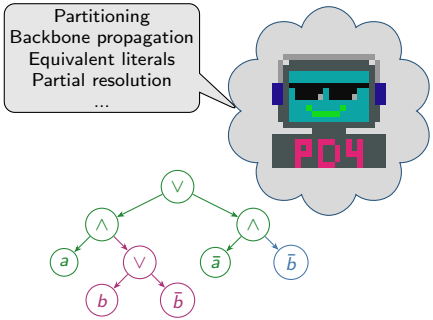
Slice c and d





Projected d-DNNF Compilation





Projected d-DNNF Compilation — Evaluation

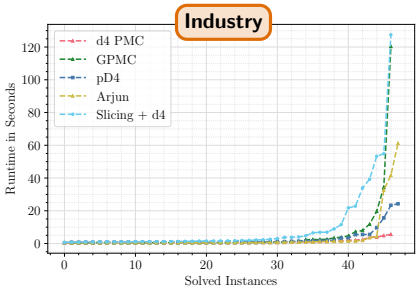
Instances

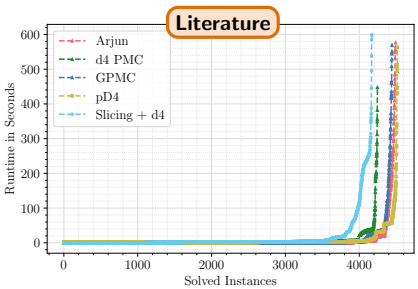
Track	Instances	Variables	Clauses
Industrial Models & Slices	$12 \cdot 4$	376–3,286	1,015–51,450
Literature Models & Random Slices	$49 \cdot 100$	17–62,482	16–350,221
Model Counting Competition Instances	200	100–3,423,788	545–6,163,392

Tools

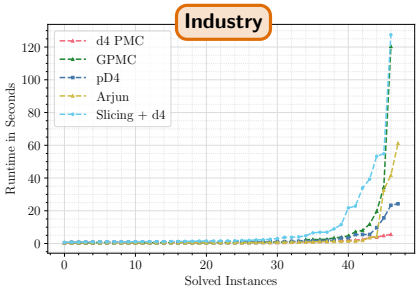
Name	Type	Origin
pd4	Projected d-DNNF Compiler	This work
Slicing FeatureIDE (v3.10)	Slicer via Resolution	[Krieter et al.(2016)]
d4 (v2)	d-DNNF Compiler	[Lagniez and Marquis(2017)]
GPMC (v1.1.1)	Projected Model Counter	[Fichte et al.(2021)]
d4-pmc (v2)	Projected Model Counter	[Lagniez and Marquis(2017)]
arjun (MCC Version)	Projected Model Counter	[Sharma et al.(2019), Soos and Meel(2022)]

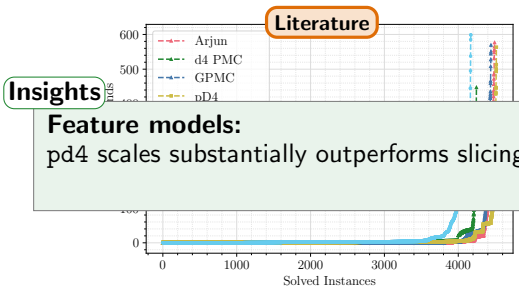
Projected d-DNNF Compilation Performance





Projected d-DNNF Compilation Performance



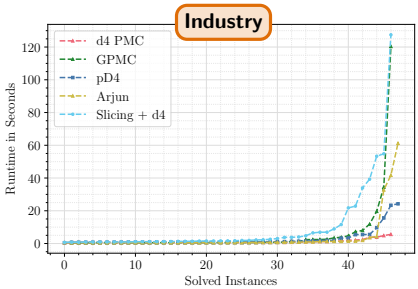


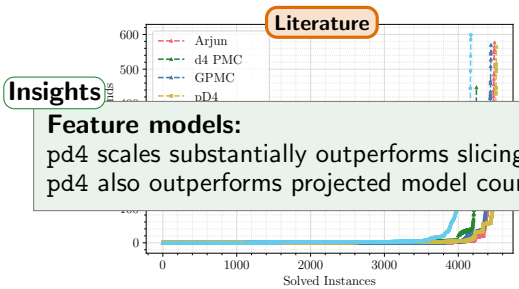
Insights

Feature models:
pd4 scales substantially outperforms slicing

Projected d-DNNF Compilation

Performance

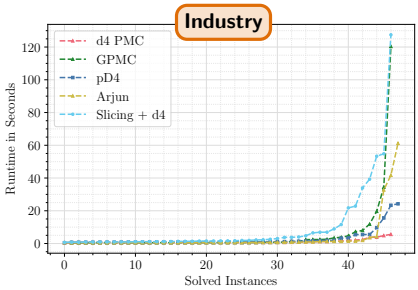


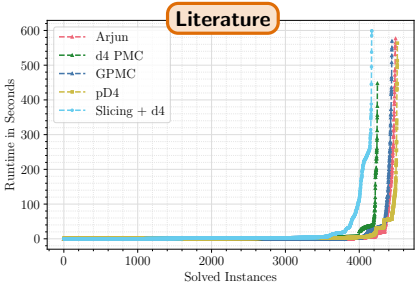


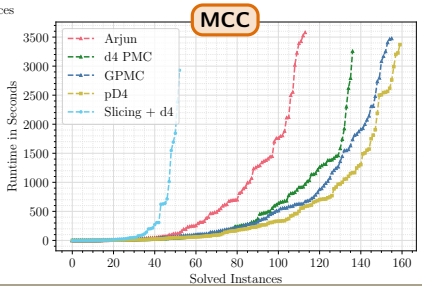
Insights

Feature models:
pd4 scales substantially outperforms slicing
pd4 also outperforms projected model counters

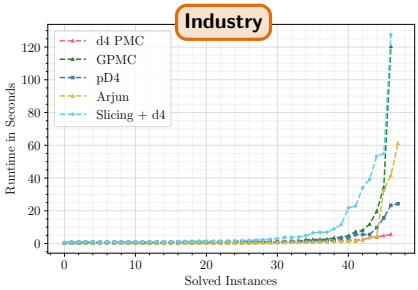
Projected d-DNNF Compilation Performance

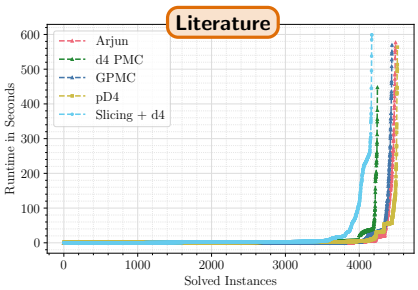


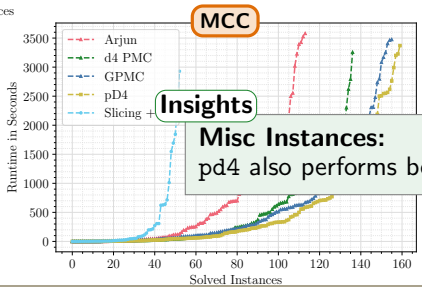




Projected d-DNNF Compilation Performance



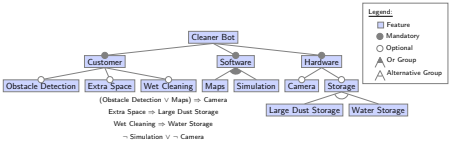




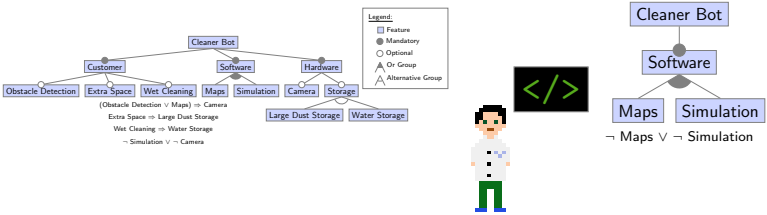
Insights

Misc Instances:
pd4 also performs best

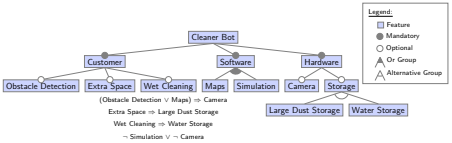
Efficient Slicing of Feature Models via Projected d-DNNF Compilation

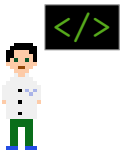


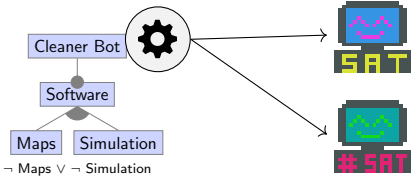
Efficient Slicing of Feature Models via Projected d-DNNF Compilation



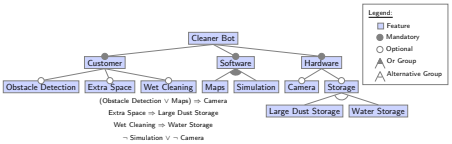
Efficient Slicing of Feature Models via Projected d-DNNF Compilation

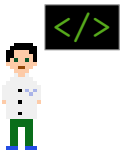


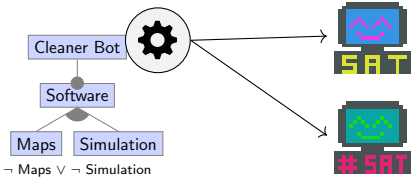


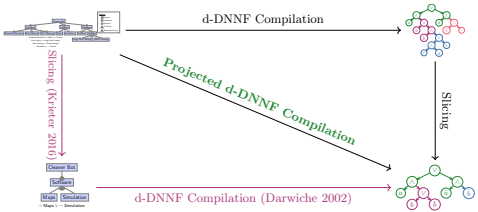


Efficient Slicing of Feature Models via Projected d-DNNF Compilation

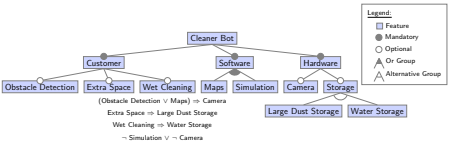


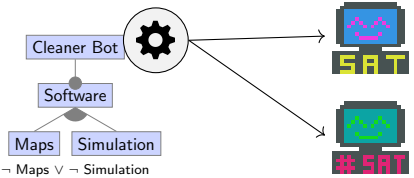


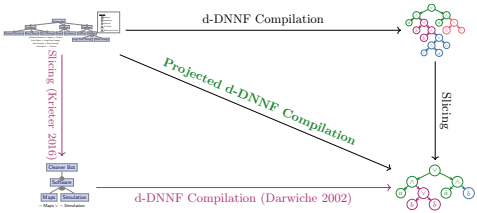


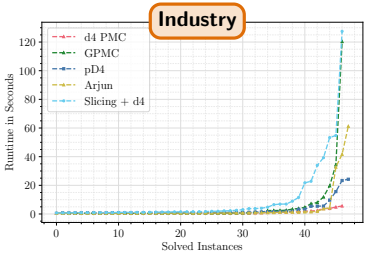


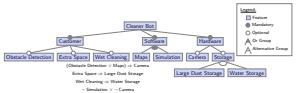
Efficient Slicing of Feature Models via Projected d-DNNF Compilation



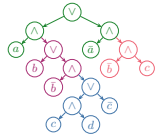








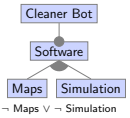
d-DNNF Compilation



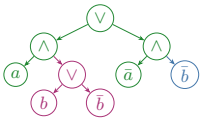
Slicing (Krieter 2016)

Projected d-DNNF Compilation

Slicing



d-DNNF Compilation (Darwiche 2002)



Efficient Slicing of Feature Models via Projected d-DNNF Compilation

ASE'24 Research Track | Chico Sundermann, Jacob Loth, Thomas Thüm | 2024/10/29

universität ulm

PADERBORN UNIVERSITY


Johannes K. Fichte, Markus Hecher, and Florim Hamiti. 2021.
The Model Counting Competition 2020.
ACM J. of Experimental Algorithmics (JEA) 26, Article 13 (2021), 26 pages.


Sebastian Krieter, Reimar Schröter, Thomas Thüm, Wolfram Fenske, and Gunter Saake. 2016.
Comparing Algorithms for Efficient Feature-Model Slicing. In *Proc. Int'l Systems and Software Product Line Conf. (SPLC)*. ACM, 60–64.


Jean-Marie Lagniez and Pierre Marquis. 2017.
An Improved Decision-DNNF Compiler. In *Proc. Int'l Joint Conf. on Artificial Intelligence (IJCAI)*. International Joint Conferences on Artificial Intelligence, 667–673.


Shubham Sharma, Subhajit Roy, Mate Soos, and Kuldeep S Meel. 2019.
GANAK: A Scalable Probabilistic Exact Model Counter. In *Proc. Int'l Joint Conf. on Artificial Intelligence (IJCAI)*, Vol. 19. AAAI Press, 1169–1176.


Mate Soos and Kuldeep S. Meel. 2022.
Arjun: An Efficient Independent Support Computation Technique and its Applications to Counting and Sampling. In *Proc. Int'l Conf. on Computer-Aided Design (ICCAD)*. ACM, Article 71.

Efficient Slicing of Feature Models via Projected d-DNNF Compilation