A Compiler for Weak Decomposable Negation Normal Form

Petr Illner and Petr Kučera

Department of Theoretical Computer Science and Mathematical Logic, Faculty of Mathematics and Physics
Charles University, Czech Republic

illner3@gmail.com, kucerap@ktiml.mff.cuni.cz

Knowledge representation

How to represent propositional theories

Representations (aka languages) are studied from the following perspectives:

1) Universality: Can the language represent any Boolean function?

2) Succinctness[1]: How compactly can the language represent Boolean functions with respect to other languages?

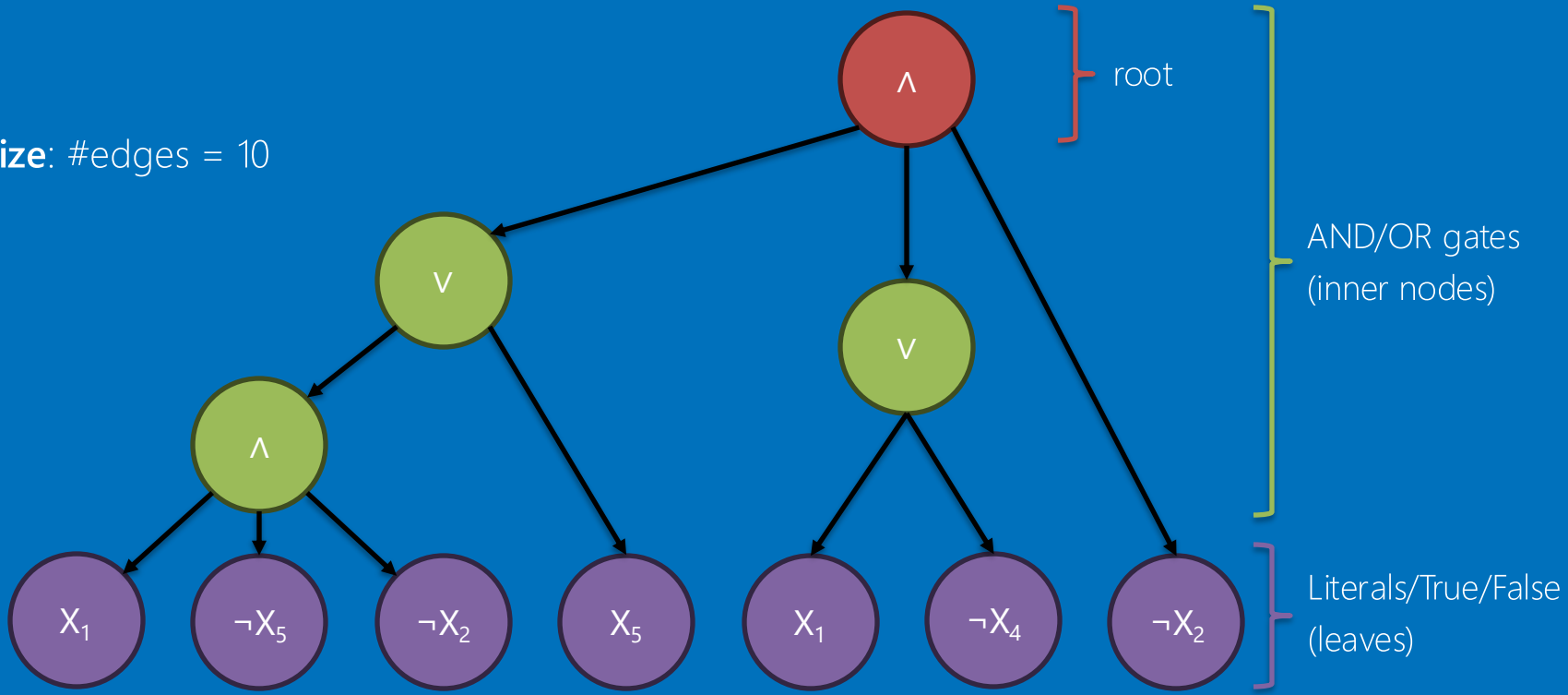
3) Tractable operations: What operations (that is, queries and transformations) does the language satisfy?

4) **Knowledge compiler**: Is there a knowledge compiler for the language?

[1] GOGIC, Goran, et al. The comparative linguistics of knowledge representation. In: IJCAI (1). 1995. p. 862-869.

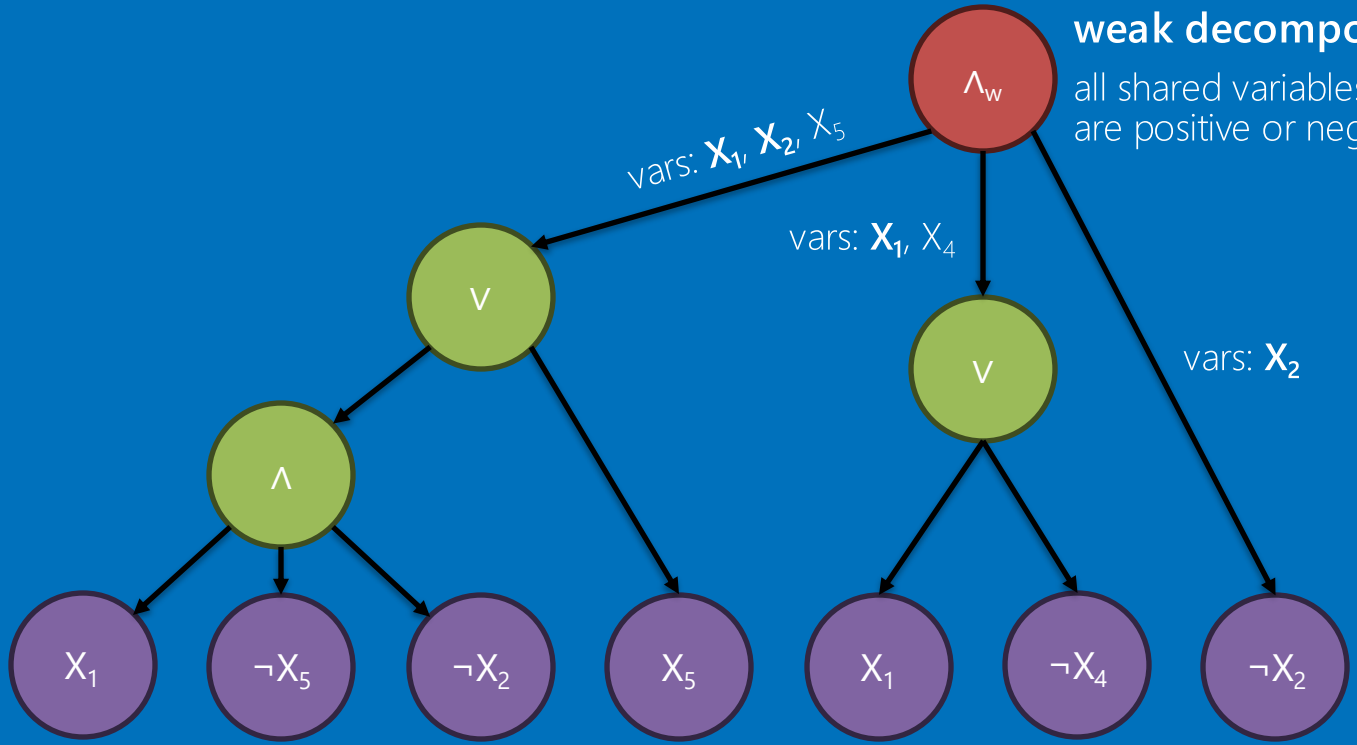
Weak decomposable **negation normal form** (wD**NNF**) **circuits**[1]

Size: #edges = 10



[1] AKSHAY, S., et al. Knowledge compilation for Boolean functional synthesis. In: 2019 Formal Methods in Computer Aided Design (FMCAD). IEEE, 2019. p. 161-169.

Weak decomposable negation normal form (wDNNF) circuits[1]



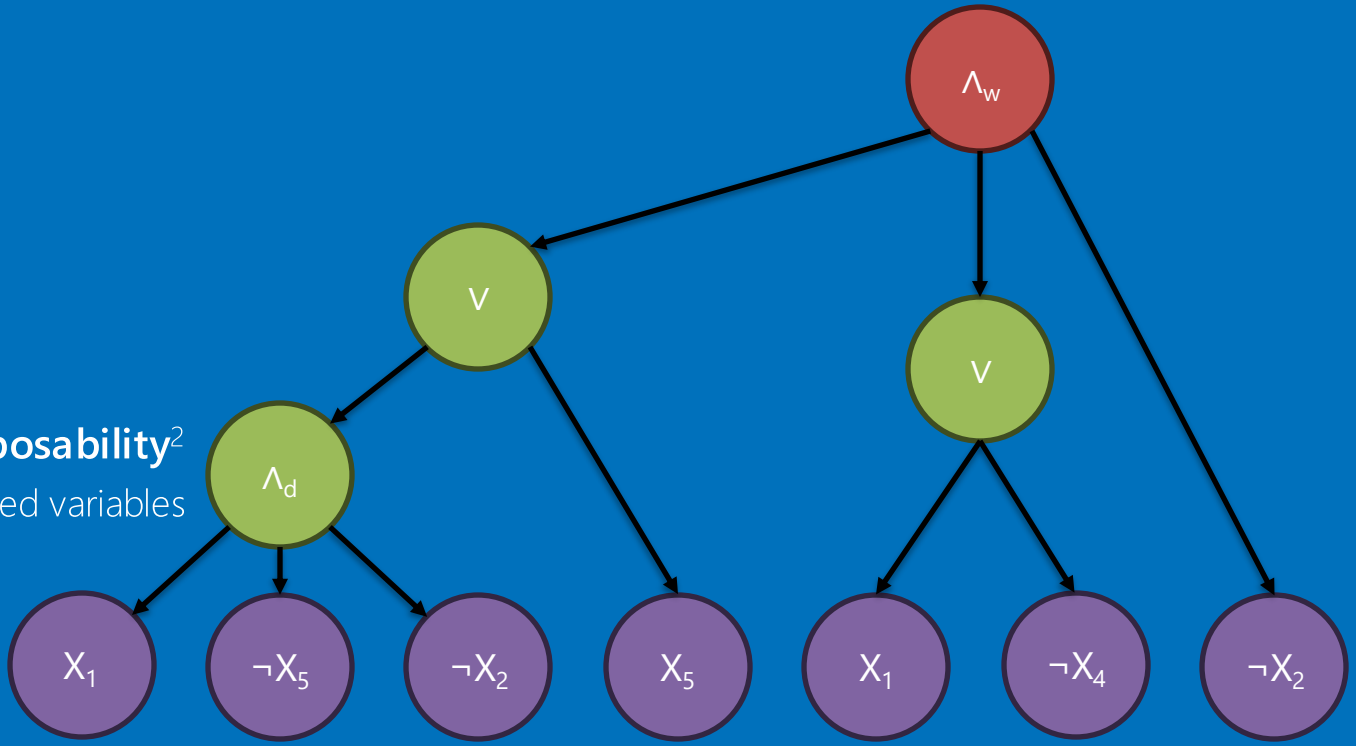
weak decomposability[1]

all shared variables (that is, X_1, and X_2) are positive or negative in the subcircuit

[1] AKSHAY, S., et al. Knowledge compilation for Boolean functional synthesis. In: 2019 Formal Methods in Computer Aided Design (FMCAD). IEEE, 2019. p. 161-169.

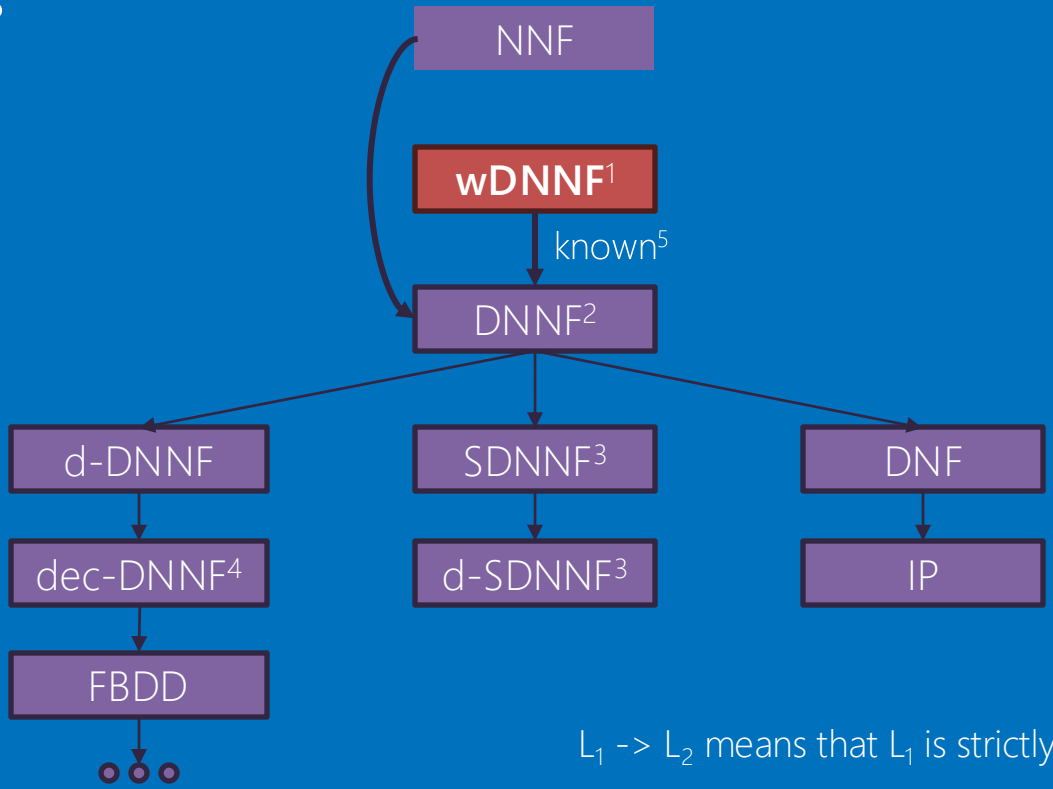
Weak decomposable negation normal form (wDNNF) circuits[1]

(classical) **decomposability**[2]
no shared variables



[1] AKSHAY, S., et al. Knowledge compilation for Boolean functional synthesis. In: 2019 Formal Methods in Computer Aided Design (FMCAD). IEEE, 2019. p. 161-169.
[2] DARWICHE, Adnan. Compiling knowledge into decomposable negation normal form. In: IJCAI. 1999. p. 284-289.

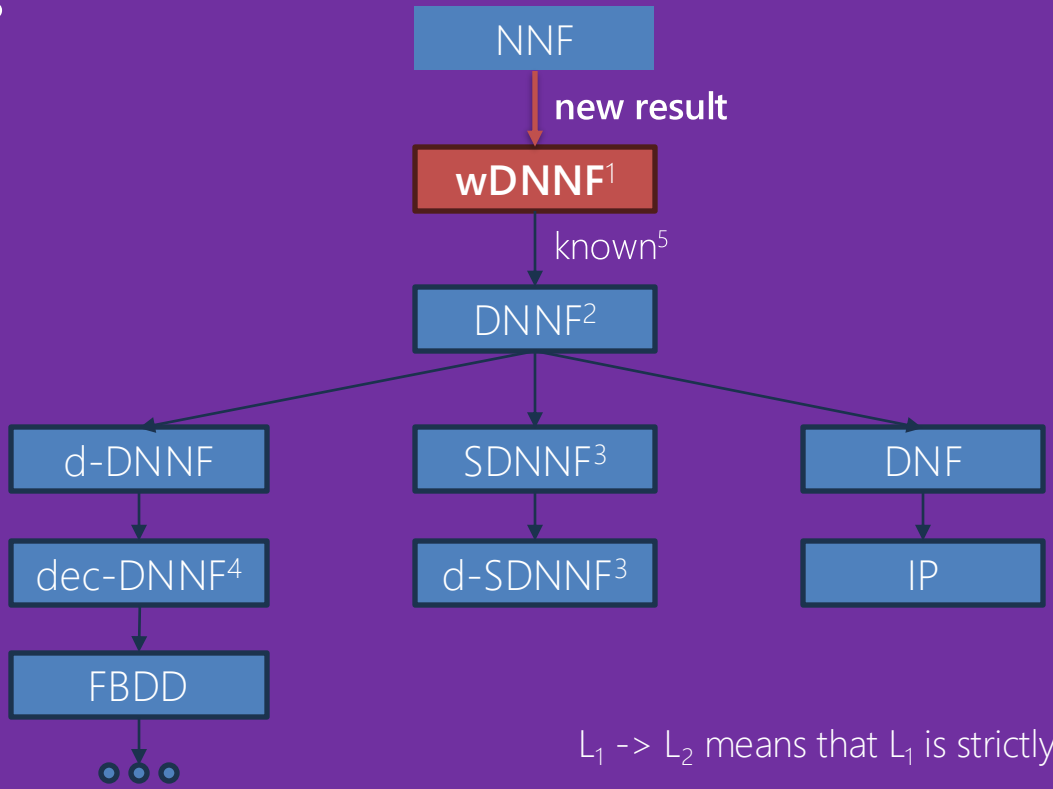
Succinctness



L_1 -> L_2 means that L_1 is strictly more succinct than L_2

[1] AKSHAY, S., et al. Knowledge compilation for Boolean functional synthesis. In: 2019 Formal Methods in Computer Aided Design (FMCAD). IEEE, 2019. p. 161-169.
[2] DARWICHE, Adnan. Compiling knowledge into decomposable negation normal form. In: IJCAI. 1999. p. 284-289.
[3] PIPATSRISAWAT, Knot; DARWICHE, Adnan. New Compilation Languages Based on Structured Decomposability. In: AAAI. 2008. p. 517-522.
[4] HUANG, Jinbo; DARWICHE, Adnan. The language of search. Journal of Artificial Intelligence Research, 2007, 29: 191-219.
[5] DE COLNET, Alexis; MENGEL, Stefan. A Compilation of Succinctness Results for Arithmetic Circuits. arXiv preprint arXiv:2110.13014, 2021.

Succinctness



L_1 -> L_2 means that L_1 is strictly more succinct than L_2

[1] AKSHAY, S., et al. Knowledge compilation for Boolean functional synthesis. In: 2019 Formal Methods in Computer Aided Design (FMCAD). IEEE, 2019. p. 161-169.
[2] DARWICHE, Adnan. Compiling knowledge into decomposable negation normal form. In: IJCAI. 1999. p. 284-289.
[3] PIPATSRISAWAT, Knot; DARWICHE, Adnan. New Compilation Languages Based on Structured Decomposability. In: AAAI. 2008. p. 517-522.
[4] HUANG, Jinbo; DARWICHE, Adnan. The language of search. Journal of Artificial Intelligence Research, 2007, 29: 191-219.
[5] DE COLNET, Alexis; MENGEL, Stefan. A Compilation of Succinctness Results for Arithmetic Circuits. arXiv preprint arXiv:2110.13014, 2021.

Queries

Query[1]	NNF	**wDNNF**	=	DNNF	d-DNNF
Consistency (CO)	✗	✓		✓	✓
Validity	✗	✗*		✗	✓
Clausal entailment (CE)	✗	✓*		✓	✓
Implicant	✗	✗*		✗	✓
Equivalence	✗	✗*		✗	?
Sentential entailment	✗	✗*		✗	✗
Model counting	✗	✗*		✗	✓
Model enumeration (ME)	✗	✓*		✓	✓

✓ polytime ✓ polynomial delay ✗ not polytime unless $P = NP$? unknown * **new result**

[1] DARWICHE, Adnan; MARQUIS, Pierre. A knowledge compilation map. Journal of Artificial Intelligence Research, 2002, 17: 229-264.

Transformations

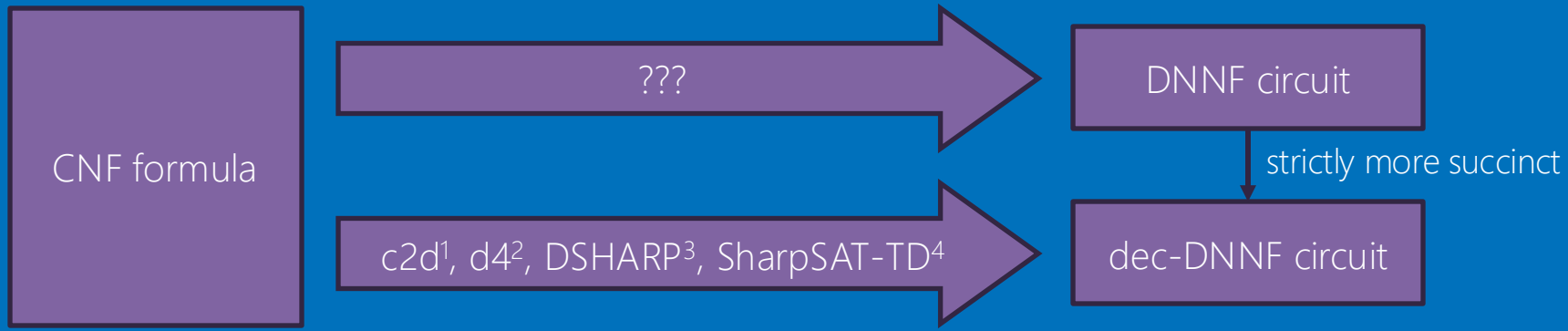
Transformation[1]	NNF	**wDNNF**	=	DNNF	d-DNNF
Conditioning (CD)	✓	✓		✓	✓
Singleton forgetting (SFO)	✓	✓		✓	✗
Forgetting (FO)	✗	✓		✓	✗
Conjunction	✓	✗*		✗	✗
Bounded conjunction	✓	✗*		✗	✗
Disjunction	✓	✓*		✓	✗
Bounded disjunction	✓	✓*		✓	✗
Negation	✓	✗*		✗	?

✓ polytime ✗ not polytime unless $\mathbf{P} = \mathbf{NP}$? unknown * **new result**

[1] DARWICHE, Adnan; MARQUIS, Pierre. A knowledge compilation map. Journal of Artificial Intelligence Research, 2002, 17: 229-264.

Knowledge compilers

Currently, if we need any of the operations satisfied by DNNF circuits, we must use decision-DNNF circuits, which also satisfy harder operations (for example, model counting).



[1] DARWICHE, Adnan, et al. New advances in compiling CNF to decomposable negation normal form. In: Proc. of ECAI. Citeseer, 2004. p. 328-332.
[2] LAGNIEZ, Jean-Marie; MARQUIS, Pierre. An Improved Decision-DNNF Compiler. In: IJCAI. 2017. p. 667-673.
[3] MUISE, Christian, et al. Fast d-DNNF Compilation with sharpSAT. In: Proceedings of the 8th AAAI Conference on Abstraction, Reformulation, and Approximation. 2010. p. 54-60.
[4] KIESEL, Rafael; EITER, Thomas. Knowledge compilation and more with SharpSAT-TD. In: Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning. 2023.

Open question:

Is there a knowledge compiler for DNNF circuits?

Open question:

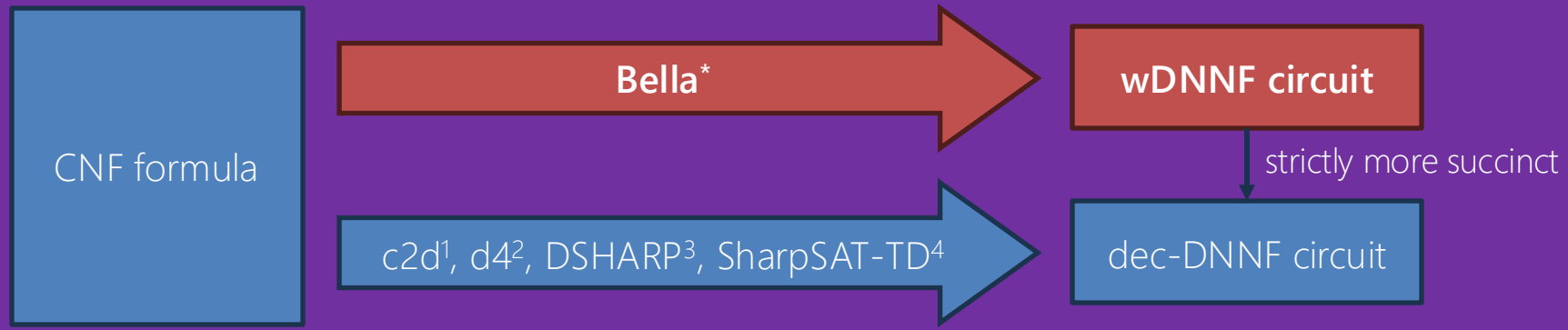
Is there a knowledge compiler for DNNF circuits?

can be reformulated as

Is there a knowledge compiler for a circuit type with the same properties as DNNF circuits?

Knowledge compilers

Instead of DNNF circuits, we can use wDNNF circuits and our new knowledge compiler Bella.



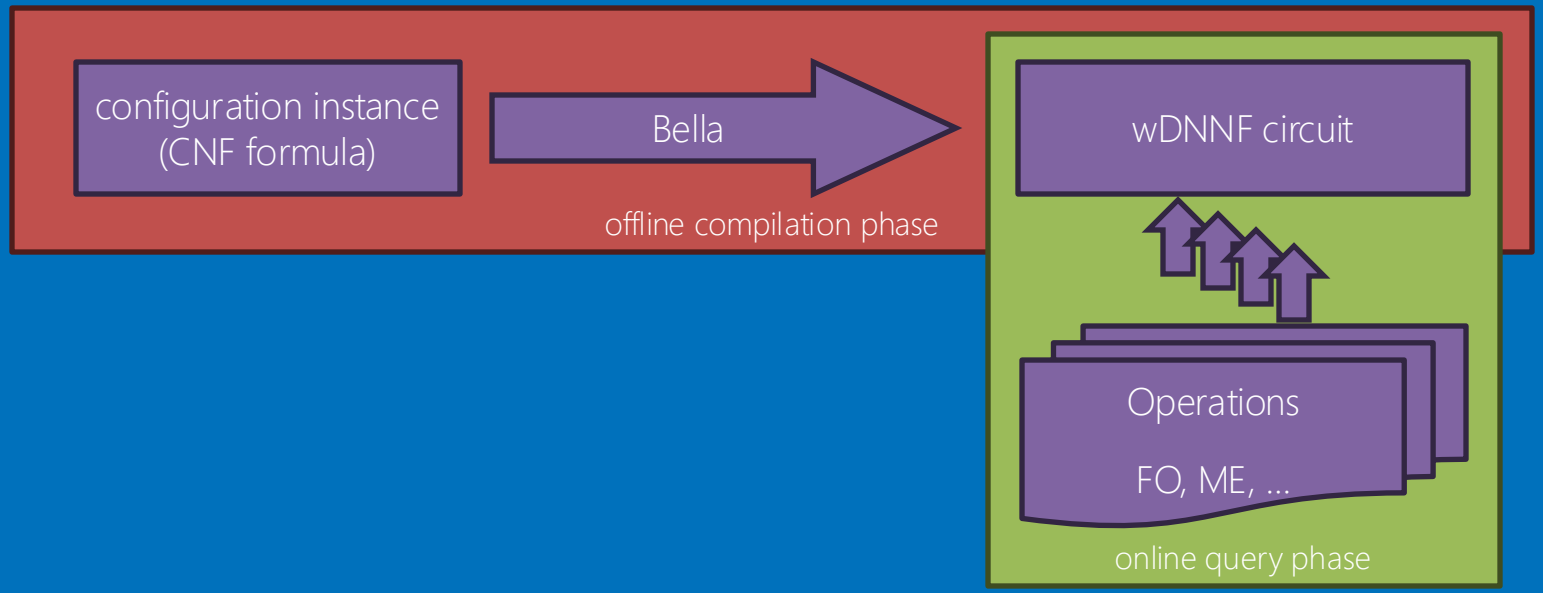
* **new knowledge compiler**

[1] DARWICHE, Adnan, et al. New advances in compiling CNF to decomposable negation normal form. In: Proc. of ECAI. Citeseer, 2004. p. 328-332.
[2] LAGNIEZ, Jean-Marie; MARQUIS, Pierre. An Improved Decision-DNNF Compiler. In: IJCAI. 2017. p. 667-673.
[3] MUISE, Christian, et al. Fast d-DNNF Compilation with sharpSAT. In: Proceedings of the 8th AAAI Conference on Abstraction, Reformulation, and Approximation. 2010. p. 54-60.
[4] KIESEL, Rafael; EITER, Thomas. Knowledge compilation and more with SharpSAT-TD. In: Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning. 2023.

Applications

Problem: **Enumeration of valid partial configurations**

Required operations: Forgetting (FO), and model enumeration (ME)



Experimental results

Configuration instance[1]	**wDNNF circuit**		decision-DNNF circuit				
	Bella		d4		SharpSAT-TD		c2d
	time (s)	size	time (s)	size	time (s)	size	size
C202_FS	**66.3**	**4 245 485**	1 256.2	154 862 147	176.1	341 407 473	169 398 649
C202_FW	**70.7**	**7 344 961**	---	---	151.4	253 123 753	195 320 974
C210_FS	**33.2**	**4 300 692**	1 065.3	240 087 456	239.6	425 732 421	71 522 262
C210_FW	**56.7**	**6 429 027**	4 048.2	715 624 258	339.9	609 298 328	118 653 033

The compilation times (in seconds), and the circuit sizes of the four most challenging configuration instances.

[1] https://www.cril.univ-artois.fr/kc/benchmarks.html

Main contributions

The main contributions of this paper are:

1) We studied wDNNF circuits in terms of the knowledge compilation map.

2) We showed that NNF circuits are strictly more succinct than wDNNF circuits, even if $\mathbf{P} = \mathbf{NP}$.

3) We presented and evaluated our knowledge compiler for wDNNF circuits.

4) We demonstrated that wDNNF circuits are suitable for efficient enumeration of valid partial configurations.