Integrating Tree Decompositions into Decision Heuristics of Propositional Model Counters

Tuukka Korhonen and Matti Järvisalo

HIIT, Department of Computer Science, University of Helsinki, Finland

CP 2021
Online

Outline

- Problem: Propositional model counting (#SAT): Given a CNF-formula, count the number of solutions
- Approach: Use tree decompositions in the decision heuristic of the model counter SharpSAT
- Results:
  - Significant improvement over state-of-the-art on standard benchmark
  - First places in 3 out of 4 tracks of model counting competition 2021

SharpSAT-TD

SharpSAT [Thurley ’06]

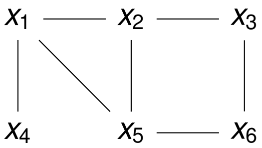
1. Preprocess
2. Count using #DPLL + clause learning + component caching

SharpSAT-TD

1. Preprocess
2. Compute tree decomposition with FlowCutter [Strasser ’17]
3. Integrate tree decomposition into variable scores
4. Count using #DPLL + clause learning + component caching

Tree Decompositions

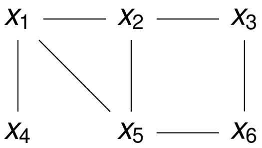
$$(\neg x_2 \vee x_3) \wedge (x_3 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge (x_1 \vee \neg x_2 \vee x_5) \wedge (x_1 \vee \neg x_4)$$



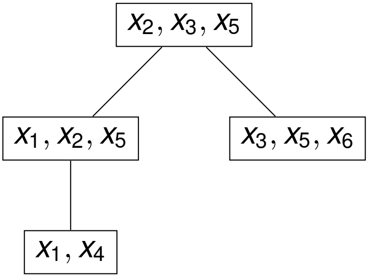
Primal graph

Tree Decompositions

$$(\neg x_2 \vee x_3) \wedge (x_3 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge (x_1 \vee \neg x_2 \vee x_5) \wedge (x_1 \vee \neg x_4)$$



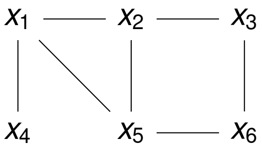
Primal graph



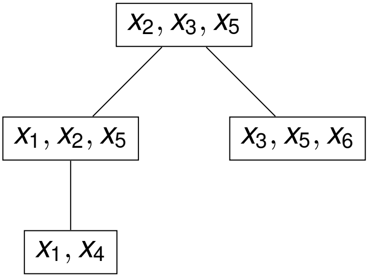
Tree decomposition

Tree Decompositions

$$(\neg x_2 \vee x_3) \wedge (x_3 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge (x_1 \vee \neg x_2 \vee x_5) \wedge (x_1 \vee \neg x_4)$$



Primal graph



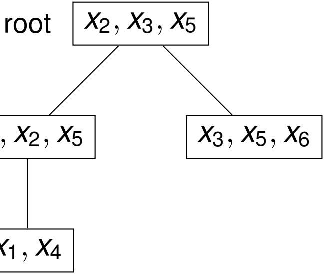
Tree decomposition

- Width of a tree decomposition: Size of the largest bag -1
- Treewidth of a graph/CNF: Minimum width of a tree decomposition

Tree Decomposition Guided Variable Selection

- Select the variable of the active formula that appears the closest to the root in the tree decomposition

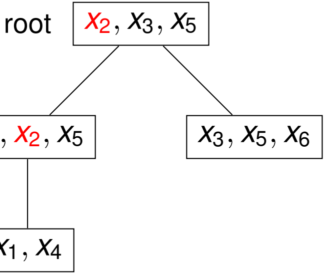
$$(\neg x_2 \vee x_3) \wedge (x_3 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge (x_1 \vee \neg x_2 \vee x_5) \wedge (x_1 \vee \neg x_4)$$



Tree Decomposition Guided Variable Selection

- Select the variable of the active formula that appears the closest to the root in the tree decomposition

$$(x_3) \wedge (x_3 \vee \neg x_6) \wedge (x_5 \vee x_6) \wedge (x_1 \vee x_5) \wedge (x_1 \vee \neg x_4)$$

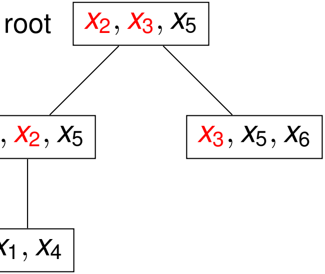


$x_2 = 1,$

Tree Decomposition Guided Variable Selection

- Select the variable of the active formula that appears the closest to the root in the tree decomposition

$$(x_5 \vee x_6) \wedge (x_1 \vee \neg x_2 \vee x_5) \wedge (x_1 \vee \neg x_4)$$

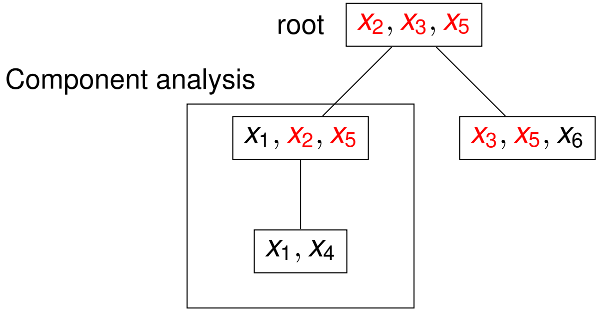


$x_2 = 1, \quad x_3 = 1,$

Tree Decomposition Guided Variable Selection

- Select the variable of the active formula that appears the closest to the root in the tree decomposition

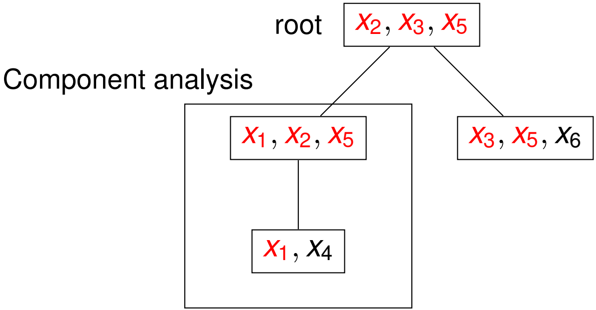
$$(x_1 \vee \neg x_4)$$



$x_2 = 1, \quad x_3 = 1, \quad x_5 = 1,$

Tree Decomposition Guided Variable Selection

- Select the variable of the active formula that appears the closest to the root in the tree decomposition



$x_2 = 1, \quad x_3 = 1, \quad x_5 = 1, \quad x_1 = 1$

Theoretical Background

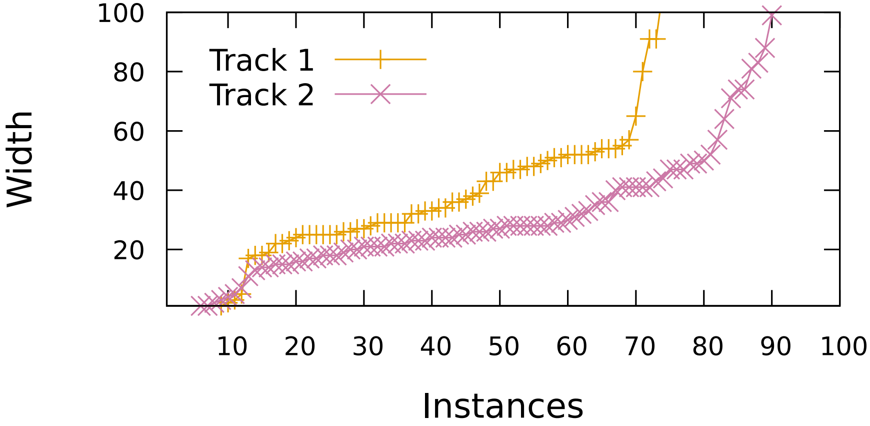
Proposition ([BDP03, Dar01])

Standard #DPLL algorithm, with component analysis and component caching, works in $2^w \mathrm{poly}(|\phi|)$ time when using a tree decomposition of width w for variable selection.

Theoretical Background

Proposition ([BDP03, Dar01])

Standard #DPLL algorithm, with component analysis and component caching, works in $2^w \text{poly}(|\phi|)$ time when using a tree decomposition of width w for variable selection.



Implementation of Variable Selection

Variable x with highest $\texttt{score}(x)$ is selected.

Standard SharpSAT:

$$\texttt{score}(x) = \texttt{act}(x) + \texttt{freq}(x)$$

Where

- $\texttt{act}(x)$ is VSIDS-like activity score
- $\texttt{freq}(x)$ is the number of occurrences of x in the current formula

Implementation of Variable Selection

Variable x with highest $\texttt{score}(x)$ is selected.

Standard SharpSAT:

$$\texttt{score}(x) = \texttt{act}(x) + \texttt{freq}(x)$$

SharpSAT-TD:

$$\texttt{score}(x) = \texttt{act}(x) + \texttt{freq}(x) - C \cdot d(x)$$

Where

- $\texttt{act}(x)$ is VSIDS-like activity score
- $\texttt{freq}(x)$ is the number of occurrences of x in the current formula
- $d(x)$ is the distance from root of tree decomposition to closest bag containing x
- C is some positive constant

Implementation of Variable Selection

Variable x with highest $\texttt{score}(x)$ is selected.

Standard SharpSAT:

$$\texttt{score}(x) = \texttt{act}(x) + \texttt{freq}(x)$$

SharpSAT-TD:

$$\texttt{score}(x) = \texttt{act}(x) + \texttt{freq}(x) - C \cdot d(x)$$

Where

- $\texttt{act}(x)$ is VSIDS-like activity score
- $\texttt{freq}(x)$ is the number of occurrences of x in the current formula
- $d(x)$ is the distance from root of tree decomposition to closest bag containing x
- C is some positive constant
  - If C is large, selection is purely by tree decomposition
  - If C is small, selection is same as in standard SharpSAT

Implementation of Variable Selection

Variable x with highest $\texttt{score}(x)$ is selected.

Standard SharpSAT:

$$\texttt{score}(x) = \texttt{act}(x) + \texttt{freq}(x)$$

SharpSAT-TD:

$$\texttt{score}(x) = \texttt{act}(x) + \texttt{freq}(x) - C \cdot d(x)$$

Where

- $\texttt{act}(x)$ is VSIDS-like activity score
- $\texttt{freq}(x)$ is the number of occurrences of x in the current formula
- $d(x)$ is the distance from root of tree decomposition to closest bag containing x
- C is some positive constant
  - If C is large, selection is purely by tree decomposition
  - If C is small, selection is same as in standard SharpSAT
  - C chosen per-instance based on the width of the tree decomposition

Experimental setting

- Set of 2424 instances merged from http://www.cril.univ-artois.fr/KC/benchmarks.html and https://github.com/dfremont/counting-benchmarks
- Time limit of 7200 seconds
- Memory limit of 16GB

Experimental setting

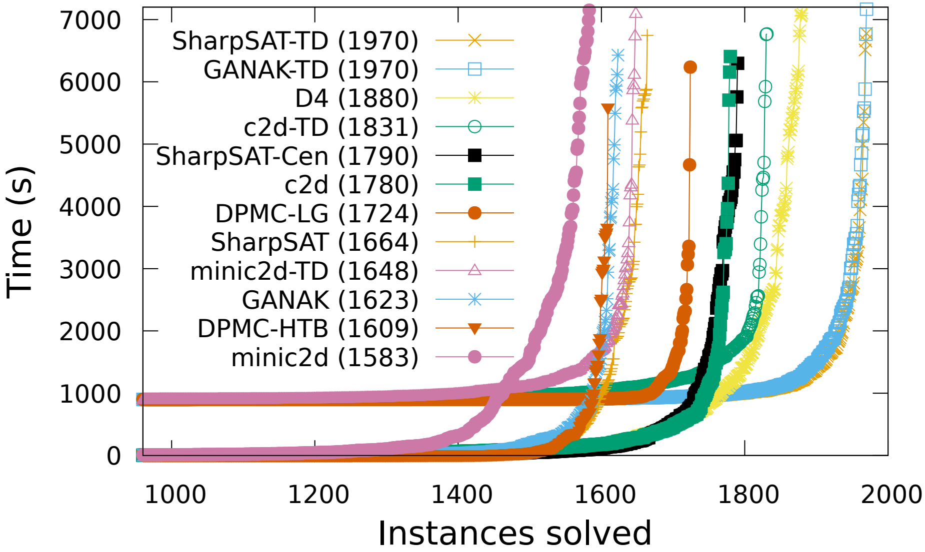
- Set of 2424 instances merged from http://www.cril.univ-artois.fr/KC/benchmarks.html and https://github.com/dfremont/counting-benchmarks
- Time limit of 7200 seconds
- Memory limit of 16GB
- 900 seconds used for computing a tree decomposition with FlowCutter

Experimental setting

- Set of 2424 instances merged from http://www.cril.univ-artois.fr/KC/benchmarks.html and https://github.com/dfremont/counting-benchmarks
- Time limit of 7200 seconds
- Memory limit of 16GB
- 900 seconds used for computing a tree decomposition with FlowCutter
- (60 seconds would yield very similar results)

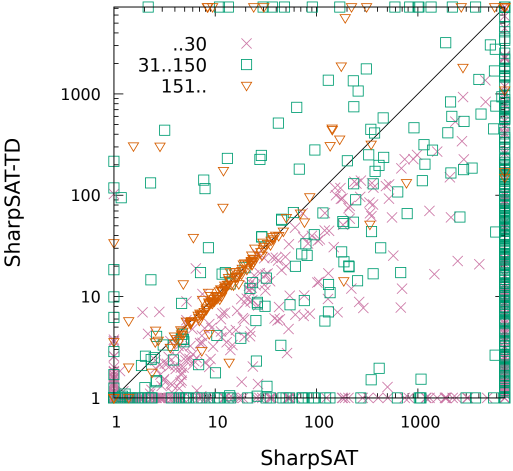
Overall Comparison

Solvers with *-TD use tree decomposition from FlowCutter, others have default settings



SharpSAT vs SharpSAT-TD

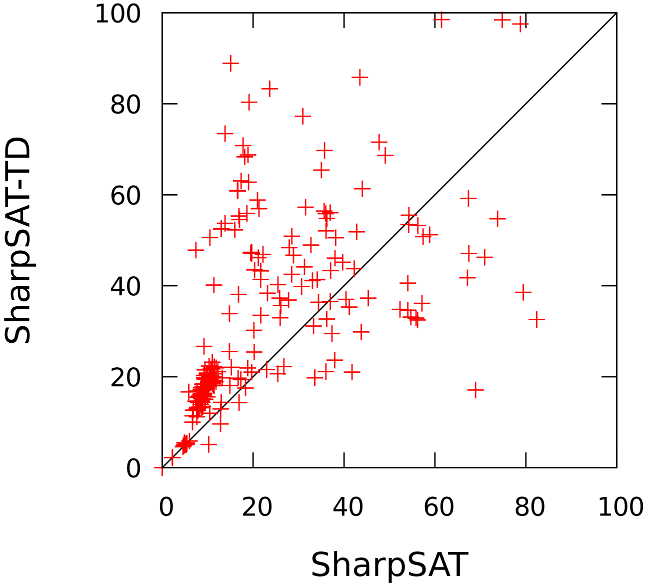
Comparison of SharpSAT and SharpSAT-TD grouped by the width of the used tree decomposition. Time used in computing tree decomposition excluded.



Width	#Ins	S	S-TD
≤ 20	810	798	**810**
21...30	526	405	**524**
31...50	378	173	**302**
51...100	259	101	**152**
101...150	57	25	**26**
151...200	128	114	**115**
201...300	43	**31**	26
$301 \leq$	223	**17**	15
Total	2424	1664	1970

Component cache hit rate

Comparison of component cache hit % in SharpSAT and SharpSAT-TD



Model Counting Competition 2021

Track 1, model counting:

#	Submission	Authors	From	solved
1	**SharpSAT-TD**	Tuukka Korhonen Matti Järvisalo	Helsinki	**78**
2	**nus-narasimha (2021)**	Sharma, Lai, Xu, Roy, Yap, Soos, Meel	Singapore, Kanpur, Changchun	**61**
3	d4 (2021)	Jean-Marie Lagniez Pierre Marquis	Lens	**51**

Track 2, weighted model counting:

#	Submission	Authors	From	solved
1	**SharpSAT-TD**	Tuukka Korhonen Matti Järvisalo	Helsinki	**90**
2	**d4 (v2021)**	Jean-Marie Lagniez Pierre Marquis	Lens	**80**
3	c2d (v3.0.0 MC2021)	Adnan Darwiche	LA	79

Track 4, approximate model counting:

#	Submission	From	solved
1	**SharpSAT-TD**	Helsinki	**68**
2	**Nus-narasimha (2021)**	Singapore, Kanpur, Changchun	**65**
3	d4 (2021)	Lens	53

The end

Thank you for your attention!

Comparison with gpusat and NestHDB

Width	#Ins	VBS	gpusat	NestHDB	SharpSAT-TD
≤ 30	1232	1232	1232	1232	1232
$31 \ldots 50$	21	14	1	10	**14**
$51 \ldots 100$	15	10	0	7	**9**
$101 \ldots 200$	18	16	0	16	16
$201 \ldots 266$	21	11	0	8	**10**
$267 \leq$	187	0	0	0	0
Total	1494	1283	1233	1273	**1281**

Bibliography

Gilles Audemard and Laurent Simon.
Predicting learnt clauses quality in modern SAT solvers.
In *IJCAI*, pages 399–404, 2009.

F. Bacchus, S. Dalmao, and T. Pitassi.
Algorithms and complexity results for #SAT and Bayesian inference.
In *FOCS*, pages 340–351. IEEE, 2003.

A. Darwiche.
Decomposable negation normal form.
J. ACM, 48(4):608–647, 2001.

J. Lagniez, E. Lonca, and P. Marquis.
Improving model counting by leveraging definability.
In *IJCAI*, pages 751–757. IJCAI/AAAI Press, 2016.

S. Sharma, S. Roy, M. Soos, and K. S. Meel.
GANAK: A scalable probabilistic exact model counter.
In *IJCAI*, pages 1169–1176. ijcai.org, 2019.