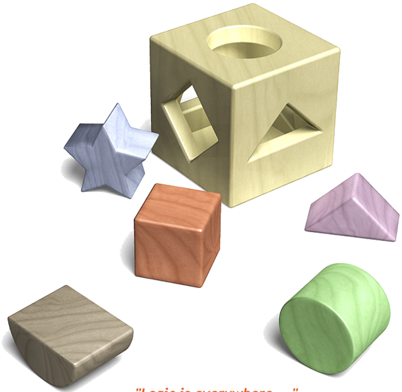
TECHNISCHE UNIVERSITÄT DRESDEN

Visualizing Dynamic Programming On Tree Decompositions

Martin Röbke

International Center for Computational Logic
Technische Universität Dresden
Germany

- WHAT **is this about?**
- WHO **benefits from visualization?**



"Logic is everywhere ..."

About me

Martin Röbke

- **born in Dresden**
- **studying Computer Science Bachelor**
  - **started studying physics at the TU Dresden**
  - **did like logic and visualization more, so switched the faculty** ☺

How did I get to work with my supervisor Johannes Fichte?

Motivation

Previous work:

- **Boolean formulas are very expressive!**
  - Problem with huge instances
- Customized algorithms, data-structures, hardware

Why visualization?

→ trace and document the customization

Outlook:

- Improve and streamline the visualization process
- Implement debug-output in existing solvers
- Even more dynamic possibilities

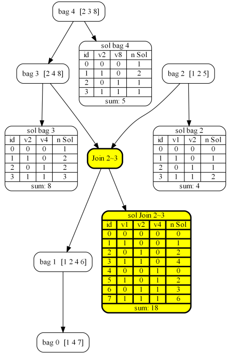


Figure: Example of a #SAT run with DP

Motivation

Previous work:

- **Boolean formulas are very expressive!**
  - **Problem with huge instances**
- **Customized algorithms, data-structures, hardware**

Why visualization?

$\longrightarrow$ trace and document the customization

Outlook:

- **Improve and streamline the visualization process**
- **Implement debug-output in existing solvers**
- **Even more dynamic possibilities**

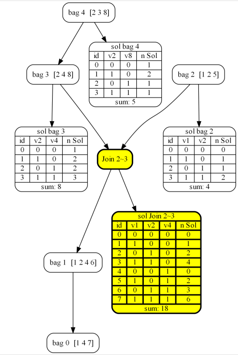


Figure: Example of a #SAT run with DP

Motivation

Previous work:

- Boolean formulas are very expressive!
  - Problem with huge instances
- Customized algorithms, data-structures, hardware

Why visualization?

$\longrightarrow$ trace and document the customization

Outlook:

- Improve and streamline the visualization process
- Implement debug-output in existing solvers
- Even more dynamic possibilities

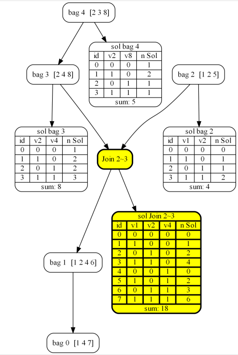


Figure: Example of a #SAT run with DP

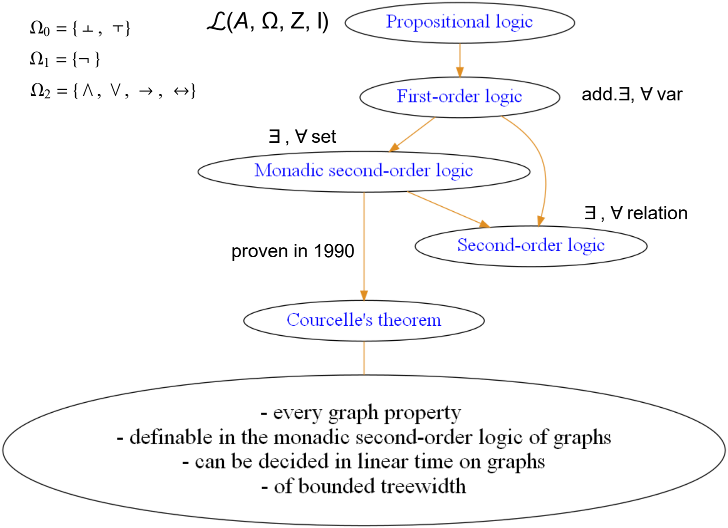
Background

$\Omega_0 = \{\perp, \top\}$

$\Omega_1 = \{\neg\}$

$\Omega_2 = \{\wedge, \vee, \rightarrow, \leftrightarrow\}$

$\mathcal{L}(A, \Omega, Z, I)$



Example: Vertex-Cover problem

For this graph **G** we want to compute a set of vertices so that from every edge **(*u*, *v*)** there is at least one of ***u*** or ***v*** in that “cover” of **G**.

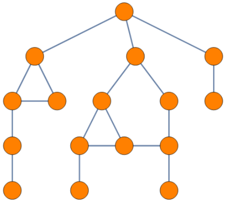


Figure: Example undirected graph **G**

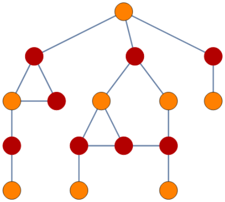


Figure: Minimal VC of **G**

$$\exists \boldsymbol{S} : \forall \boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{E} : (\boldsymbol{x} \in \boldsymbol{S} \vee \boldsymbol{y} \in \boldsymbol{S})$$

Finding the smallest of these sets is the optimization version of a NP-complete decision problem.

Courcelle's theorem

- **statement:**

  Every graph property definable in MSOL is decidable in linear time on graphs of bounded treewidth.

  Courcelle, Bruno (Bordeaux I University 1990)

- **drawback:** still expensive (2^{k*tw}, $2^{2^{\#quant}}$, large constants)
- **usage:**

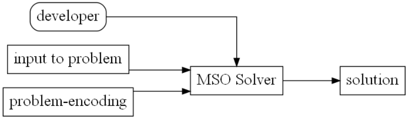


Figure: Implementation of the theorem

Courcelle's theorem

- **statement:**

  Every graph property definable in MSOL is decidable in linear time on graphs of bounded treewidth.

  Courcelle, Bruno (Bordeaux I University 1990)

- **drawback:** still expensive (2^{k*tw}, $2^{2^{\#quant}}$, large constants)
- **usage:**

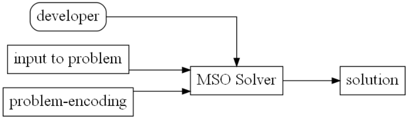


Figure: Implementation of the theorem

Courcelle's theorem

- **statement:**

  Every graph property definable in MSOL is decidable in linear time on graphs of bounded treewidth.

  Courcelle, Bruno (Bordeaux I University 1990)

- **drawback:** **still expensive (2^{k*tw}, $2^{2^{\#quant}}$, large constants)**
- **usage:**

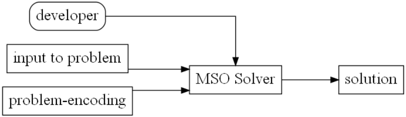


Figure: Implementation of the theorem

Courcelle's theorem

- **statement:**

  Every graph property definable in MSOL is decidable in linear time on graphs of bounded treewidth.

  Courcelle, Bruno (Bordeaux I University 1990)

- **drawback:** still expensive (2^{k*tw}, $2^{2^{\#quant}}$, large constants)
- **usage:**

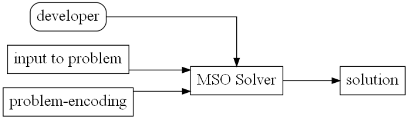


Figure: Implementation of the theorem

(Weighted) Model-Counting

The #SAT Problem

Input: A boolean formula **F**
Question: How many assignments do the occurring variables satisfy in **F**

The Weighted Model Counting Problem

- $w(lit) \in [0,1], \quad w(\neg lit) = 1 - w(lit)$
- $w(assignment) = \prod_{lit} w(lit)$
- $WMC(formula) = \sum_{satisfying\ \ assignments} w(assignment)$

Graphs for Boolean Formulas

Transforming the formula into CNF-form

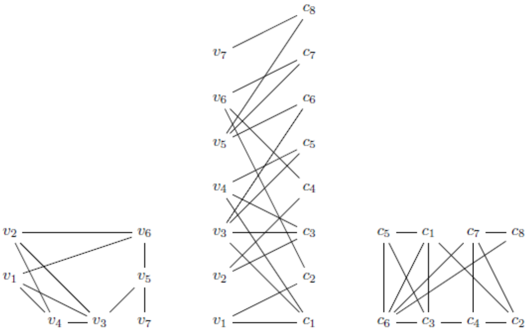


Figure 2.1: The primal (left), incidence (middle) and dual (right) graph for the SAT formula in Example 2.1

Tree Decompositions

Parameterized Complexity and its Applications in Practice
From Foundations to Implementations
Johannes K. Fichte
TU Dresden, Germany
Jakarta, Indonesia
Summer 2019 (May 6th - May 16th)
pages 162-174
Backup: VC tree vs graph - example

gpuSAT1

graphic / github

- **OpenCL**
- **Incidence + Primal Graph**

gpuSAT2

graphic / github

- **OpenCL**
- **Only primal graph - simpler solving DP**
- **adapted memory-management**
- **improved precision handling**
- **customized tree decompositions**

dpdb

graphic / github

- **using databases for intermediate results**
- **SAT**
- **#SAT**
- **Vertex Cover**

TODO - DA

τ_1 :

ID	c_1	c_5	c_6	n Sol
0	0	0	0	48
1	1	0	0	4
2	0	1	0	16
3	1	1	0	0
4	0	0	1	15
5	1	0	1	1
6	0	1	1	10
7	1	1	1	0

τ_4 :

ID	c_1	c_6	n Sol
0	0	0	32
1	1	0	4
2	0	1	10
3	1	1	1

JOIN $\longrightarrow$

τ_2 :

ID	c_1	c_5	n Sol
0	0	0	12
1	1	0	2
2	0	1	4
3	1	1	0

JOIN $\longleftarrow$

IF $\uparrow$

τ_5 :

ID	c_1	c_2	c_6	n Sol
0	0	0	0	36
1	1	0	0	6
2	0	1	0	4
3	1	1	0	2
4	0	0	1	12
5	1	0	1	2
6	0	1	1	2
7	1	1	1	1

$\uparrow$

τ_3 :

ID	c_1	c_4	c_5	n Sol
0	0	0	0	16
1	1	0	0	4
2	0	1	0	4
3	1	1	0	2
4	0	0	1	4
5	1	0	1	0
6	0	1	1	0
7	1	1	1	0

IF $\uparrow$

τ_6 :

ID	c_1	c_2	n Sol
0	0	0	12
1	1	0	2
2	0	1	2
3	1	1	1

JOIN $\longrightarrow$

τ_7 :

ID	c_2	c_6	n Sol
0	0	0	12
1	1	0	2
2	0	1	4
3	1	1	1

JOIN $\longleftarrow$

$\uparrow$

τ_9 :

ID	c_1	c_2	c_3	n Sol
0	0	0	0	16
1	1	0	0	4
2	0	1	0	4
3	1	1	0	2
4	0	0	1	4
5	1	0	1	2
6	0	1	1	2
7	1	1	1	1

IF $\uparrow$

$\uparrow$

τ_8 :

ID	c_2	c_6	c_7	n Sol
0	0	0	0	16
1	1	0	0	4
2	0	1	0	4
3	1	1	0	1
4	0	0	1	4
5	1	0	1	2
6	0	1	1	0
7	1	1	1	0

IF $\uparrow$

Figure:

TODO - presentation_gpusat

Creating Visualization for:

Improving

- **documentation**
- **debugging complex ds and parallel sync**
- **hotspots**

Generalizing the underlying graph

Outlook

- customizable output and interactive visualization
- ref. impl. in CUDA of gpuSAT2

Further Information

-

References

-