Recent algorithmic advances for combinatorial optimization in graphical models

Simon de Givry, Thomas Schiex, David Allouche, George Katsirelos, Matthias Zytnicki, MIAT – INRA, Toulouse, France

Abdelkader Ouali, Samir Loudni, GREYC, University of Caen, France
Plan

- Graphical models
  - Examples and definitions
- Local reasoning techniques
  - Bounding, clique cut, pruning
- Complete search methods
  - Hybrid search, iterative search, large neighborhood search
- Experimental results
  - Open-source C++ exact solver **toulbar2 v1.0.0**

https://github.com/toulbar2/toulbar2
Earth Observation Satellite Management (SPOT5)

\[ n \leq 364, \quad d=4, \quad e(2-3) \leq 10,108 \]

(Bensana et al, Constraints 1999 ; IJCAI09)
Radio Link Frequency Assignment (CELAR)

(Cabon et al, Constraints 1999 ; CP97 – AAAI06 – IJCAI07 – IJCAI09 – CP10)

\[ n \leq 458, \ d=44, \ e(2) \leq 5,000 \]
Mendelian error correction in complex pedigree (MendelSoft)

\( n \leq 20,000, \ d \leq 66, \ e(3) \leq 30,000 \)
Genetic Linkage Analysis

(Marinescu & Dechter, AAAI 2006 ; IJCAI11)

\[ n \leq 1,200, \ d \leq 7, \ e(2-5) \leq 2,000 \]
Protein Design

(CP12 – Bioinformatics13 - AIJ14 – JCTC15 – ISMP18)

n ≤ 120, d ≤ 190, e(2) ≤ 7,260
Graph Matching (worms segmentation)

(Kainmueller et al, Med Image Comput 2014 ; Haller et al, AAAI 2018)

\[ n \leq 558, \quad d \leq 128, \quad e(2) \leq 23,407 \]
Definition (Graphical model)

- Let $X = (X_1, \ldots, X_n)$ be a set of variables.
- $X_i$ takes values in $\Lambda_i \subseteq \mathbb{R}$.
- A realization of $X$ is denoted $x = (x_1, \ldots, x_n)$, with $x_i \in \Lambda_i$.
- A graphical model over $X$ is a function $\psi : \prod_i \Lambda_i \to \mathbb{R}$, which writes, $\forall x \in X$:

$$\psi(x) = \bigcirc_{B \in \mathcal{B}} \psi_B(x_B),$$

where $\mathcal{B}$ is a set of subsets of $V = \{1, \ldots, n\}$, $\psi_B : \prod_{i \in B} \Lambda_i \to \mathbb{R}$ and $\bigcirc \in \{\prod, \sum, \min, \max \ldots\}$ is a combination operator.

Slides from Sabbadin’s invited talk at JFRB 2018
Probabilistic Graphical Models

Definition (Markov chain)

- $X = (X_1, \ldots, X_n)$ is a set of variables, with finite domains $\{\Lambda_i\}_{i=1,\ldots,n}$.

\[
P(x_1, \ldots, x_n) = \underbrace{P(x_1)}_{\psi_1(x_1)} \times \underbrace{P(x_2|x_1)}_{\psi_{12}(x_1,x_2)} \times \ldots \times \underbrace{P(x_n|x_{n-1})}_{\psi_{(n-1)n}(x_{n-1},x_n)}
\]
Probabilistic Graphical Models

Definition (Bayesian network)

- $X = (X_1, \ldots, X_n)$ is a set of variables, with finite domains $\{\mathcal{X}_i\}_{i=1,\ldots,n}$.
- $\text{Par}(i) \subseteq \{1, \ldots, i-1\}$, $\forall i = 2, \ldots, n$.

\[
P(x_1, \ldots, x_n) = \frac{P(x_1)}{\psi_1(x_1)} \times \prod_{i=2}^{n} P(x_i | x_{\text{Par}(i)})
\]
Probabilistic Graphical Models

**Definition (Markov Random Field)**

- $G = (V, E)$ is an undirected graph with vertices $V = \{1, \ldots, n\}$, edges $E \subseteq V \times V$ and $C$ is the set of cliques of $G$.
- $\{\psi_C : X_C \to \mathbb{R}^+\}_{C \in C}$ are strictly positive functions.

$$P(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{C \in C} \psi_C(x_C)$$

$\psi_0$, normalizing constant
Deterministic Graphical Model

Definition (Cost Functions networks)

- \( \{w_C : X_C \rightarrow \mathbb{R}^+\}_{C \in \mathcal{C}} \) are positive functions.

\[
w(x_1, \ldots, x_n) = \sum_{c \in \mathcal{C}} w_C(x_C)
\]
Deterministic Graphical Model

Definition (Cost Functions networks)

- \( \{w_C : X_C \rightarrow \mathbb{R}^+ \}_{C \in C} \) are positive functions.

\[
  w(x_1, \ldots, x_n) = \sum_{c \in C} w_C(x_C)
\]

Minimization task: \( \min w(X_1, \ldots, X_n) \) \hspace{1cm} NP-hard problem

\[
  w_{123} = -\log \varphi_{123}
\]

\[
  w_{124} = -\log \varphi_{124}
\]

Energy minimization task is equivalent to finding the most probable explanation
Example

In JSON compatible toolbar2 cfn format

```json
{
    "problem": { "name": "maximization", mustbe: ">=-5.0" },
    "variables": { "X1": ["a", "b"], "X2": ["c", "d"] },
    "functions": {
        "w0": { "scope": [], costs: [-6.0] },
        "w1": { "scope": ["X1"], costs: [1.0, 0.5] },
        "w2": { "scope": ["X2"], costs: [1.0, 0.5] },
        "w12": { "scope": ["X1", "X2"], costs: [-1.0, 0.5, -2.0, 5.5] }
    }
}
```
Micro-Structure

```
{
  problem: { name: "maximization", mustbe: ">-5.0"},
  variables: { "X1": ["a", "b"], "X2": ["c", "d"] },
  functions: {
    "w0": {scope: [], costs: [-6.0]},
    "w1": {scope: ["X1"], costs: [1.0, 0.5]},
    "w2": {scope: ["X2"], costs: [1.0, 0.5]},
    "w12": {scope: ["X1", "X2"], costs: [-1.0, 0.5, -2.0, 5.5]}
  }
}
```
Minimization with non-negative integer costs

```json
{
  problem: { name: "maximization", mustbe: ">-5.0"},
  variables: { "X1": ["a", "b"], "X2": ["c", "d"] },
  functions: {
    "w0": {scope: [], costs: [-6.0]},
    "w1": {scope: ["X1"], costs: [1.0, 0.5]},
    "w2": {scope: ["X2"], costs: [1.0, 0.5]},
    "w12": {scope: ["X1", "X2"], costs: [-1.0, 0.5, -2.0, 5.5]}
  }
}
```

\[ w_\emptyset = 60 \]

UB < 125
Constraints are Cost Functions

{ 
  problem: { name: "maximization", mustbe: ">-5.0"},
  variables: { "X1": ["a", "b"], "X2": ["c", "d"] },
  functions: { 
    "w0": {scope: [], costs: [-6.0]},
    "w1": {scope: ["X1"], costs: [1.0, 0.5]},
    "w2": {scope: ["X2"], costs: [1.0, 0.5]},
    "w12": {scope: ["X1", "X2"], costs: [-1.0, 0.5, -2.0, 5.5]}
  }
}
Constraints are Cost Functions

```json
{
    problem: { name: "maximization", mustbe: ">-5.0"},
    variables: { "X1": ["a", "b"], "X2": ["c", "d"] },
    functions: {
        "w0": {scope: [], costs: [-6.0]},
        "w1": {scope: ["X1"], costs: [1.0, 0.5]},
        "w2": {scope: ["X2"], costs: [1.0, 0.5]},
        "w12": {scope: ["X1", "X2"], costs: [-1.0, 0.5, -2.0, 5.5]}
    }
}
```

\[ w_\emptyset = 60 \]

\[ UB < 125 \]
**Other equivalent formulations**

*In various toulbar2 input formats*

<table>
<thead>
<tr>
<th>WCSP</th>
<th>MRF</th>
<th>Max-SAT</th>
<th>QPBO</th>
</tr>
</thead>
<tbody>
<tr>
<td>wcsp 2 2 4 125 2 2 0 1 0 4 0 0 65 0 1 50 1 0 75 1 1 0 1 0 125 2 0 0 1 5 1 1 125 2 0 0 1 5 0 60 0</td>
<td>MARKOV 2 2 4 2 0 1 0 1 1 0 1 0 4 0 00341454887383 0 0215443469003 0 0001 1 0 2 1 0 0 541169526546 2 1 0 0 541169526546 2 0 0063095734448 0 0063095734448</td>
<td>p wcnf 2 7 125 65 1 2 0 50 1 -2 0 75 -1 2 0 5 -1 0 5 -2 0 60 1 0 60 -1 0</td>
<td>4 13 1 3 32.5 1 4 25 2 3 37.5 2 2 5 4 4 5 1 1 60 2 2 60 1 1 -1000 2 2 -1000 1 2 1000 3 3 -1000 4 4 -1000 3 4 1000</td>
</tr>
</tbody>
</table>
Local reasoning techniques

Cost Function Propagation
Reparameterization and pruning

(Schiex, CP 2000; Larrosa, AAAI 2002; Cooper, FSS 2003; IJCAI05; IJCAI07; AAAI08; AIJ10)

$w_{\phi} = 60$

UB < 125
Reparameterization and pruning

(Schiex, CP 2000 ; Larrosa, AAAI 2002 ; Cooper, FSS 2003 ; IJCAI05 ; IJCAI07 ; AAAI08 ; AIJ10)
Reparameterization and pruning

(Schiex, CP 2000; Larrosa, AAAI 2002; Cooper, FSS 2003; IJCAI05; IJCAI07; AAAI08; AIJ10)
Reparameterization and pruning

(Schiex, CP 2000; Larrosa, AAAI 2002; Cooper, FSS 2003; IJCAI05; IJCAI07; AAAI08; AIJ10)

\[ w_\emptyset = 60 \]

\[ w_\emptyset = 60 \]

\[ UB < 125 \]
Reparameterization and pruning

(Shiex, CP 2000 ; Larrosa, AAAI 2002 ; Cooper, FSS 2003 ; IJCAI05 ; IJCAI07 ; AAAI08 ; AIJ10)

\[ w_\phi = 60 \]

\[ \text{UB} < 125 \]
Reparameterization and pruning

(Schiex, CP 2000; Larrosa, AAAI 2002; Cooper, FSS 2003; IJCAI05; IJCAI07; AAAI08; AIJ10)
Reparameterization and pruning

(Schiex, CP 2000; Larrosa, AAAI 2002; Cooper, FSS 2003; IJCAI05; IJCAI07; AAAI08; AIJ10)
Reparameterization and pruning

(Schiex, CP 2000; Larrosa, AAAI 2002; Cooper, FSS 2003; IJCAI05; IJCAI07; AAAI08; AIJ10)
Reparameterization and pruning

(Schiex, CP 2000; Larrosa, AAAI 2002; Cooper, FSS 2003; IJCAI05; IJCAI07; AAAI08; AIJ10)

\[ w_\emptyset = 70 \]

\[ \text{UB} < 125 \]
Reparameterization and pruning

(Schiex, CP 2000 ; Larrosa, AAAI 2002 ; Cooper, FSS 2003 ; IJCAI05 ; IJCAI07 ; AAAI08 ; AIJ10)

- Reparameterization produces a feasible solution of the dual of a strong LP relaxation
- We use a sequence of reparameterizations
  - Faster than LP
  - Not optimal: weaker dual bounds than LP
  - Many fixpoints

and domain value pruning


### Direct LP formulation

Minimize

\[ +50 \, t_{0_0_1_1} + 75 \, t_{0_1_1_0} + 65 \, t_{0_0_1_0} - 5 \, d_{0_0} - 5 \, d_{1_0} + 60 \, t + 10 \, t \]

Subject to:

\[ +1 \, d_{0_0} - 1 \, d_{1_0} - t_{0_0_1_1} \leq 0 \]
\[ -1 \, d_{0_0} + 1 \, d_{1_0} - t_{0_1_1_0} \leq 0 \]
\[ +1 \, d_{0_0} + 1 \, d_{1_0} - t_{0_0_1_0} \leq 1 \]

Bounds

\[ t_{0_0_1_0} \leq 1 \]
\[ t_{0_0_1_1} \leq 1 \]
\[ t_{0_1_1_0} \leq 1 \]
\[ t = 1 \]

Binary

\[ d_{0_0} \quad d_{1_0} \]

End

### Stronger LP formulation

Minimize

\[ +50 \, t_{0_0_1_1} + 75 \, t_{0_1_1_0} + 65 \, t_{0_0_1_0} - 5 \, d_{0_0} - 5 \, d_{1_0} + 60 \, t + 10 \, t \]

Subject to:

\[ +1 \, t_{0_0_1_0} + 1 \, t_{0_0_1_1} - 1 \, d_{0_0} = 0 \]
\[ +1 \, t_{0_1_1_0} + 1 \, t_{0_1_1_1} + 1 \, d_{0_0} = 1 \]
\[ +1 \, t_{0_0_1_0} + 1 \, t_{0_1_1_0} - 1 \, d_{1_0} = 0 \]
\[ +1 \, t_{0_0_1_1} + 1 \, t_{0_1_1_1} + 1 \, d_{1_0} = 1 \]

Bounds

\[ t_{0_0_1_0} \leq 1 \]
\[ t_{0_0_1_1} \leq 1 \]
\[ t_{0_1_1_0} \leq 1 \]
\[ t_{0_1_1_1} \leq 1 \]
\[ t = 1 \]

Binary

\[ d_{0_0} \quad d_{1_0} \]

End

(CPAIOR16 – Constraints16)
## Uncapacitated Warehouse Location Problem

(Kratica et al., RAIRO OR 2001)

### Search nodes

<table>
<thead>
<tr>
<th>Instance</th>
<th>cplex 12.7.1</th>
<th>toulbar2 1.0.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capmo1 100x100</td>
<td>155</td>
<td>7,581</td>
</tr>
<tr>
<td>Capmo2 100x100</td>
<td>25</td>
<td>2,024</td>
</tr>
<tr>
<td>Capmo3 100x100</td>
<td>93</td>
<td>5,439</td>
</tr>
<tr>
<td>Capmo4 100x100</td>
<td>23</td>
<td>4,055</td>
</tr>
<tr>
<td>Capmo5 100x100</td>
<td>28</td>
<td>2,664</td>
</tr>
</tbody>
</table>

### CPU time (sec. on PC i7 3GHz)

<table>
<thead>
<tr>
<th>Instance</th>
<th>cplex 12.7.1</th>
<th>toulbar2 1.0.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capmo1 100x100</td>
<td>13.01</td>
<td>20.13</td>
</tr>
<tr>
<td>Capmo2 100x100</td>
<td>3.06</td>
<td>3.02</td>
</tr>
<tr>
<td>Capmo3 100x100</td>
<td>13.32</td>
<td>11.40</td>
</tr>
<tr>
<td>Capmo4 100x100</td>
<td>3.26</td>
<td>7.45</td>
</tr>
<tr>
<td>Capmo5 100x100</td>
<td>2.68</td>
<td>4.62</td>
</tr>
</tbody>
</table>
Clique cuts

Given a set $S$

$$x_i + x_j \leq 1 \quad \forall x_i, x_j \in S$$

$\Rightarrow$ Satisfied by $x_i = 0.5$

But we can get

$$\sum_{x_i \in S} x_i \leq 1$$
Clique cuts in CFN

Straightforward generalization
Given a set $S$ of $\langle X_i, v_i \rangle$ with

- $c_{ij}(v_i, v_j) = \infty$

Then derive

$$\sum_{ij \in S} X_{ij} \leq 1$$
Reparameterization for clique

\( w_\emptyset = 0 \)

(\text{CP17})
Reparameterization for clique

\[ w_{123}(b,d,f) \rightarrow 2 \]

\( w_\emptyset = 3 \)

(CP17)
Reparameterization for cliques

$w_{123}(b,d,f) \rightarrow 2$

$w_\emptyset = 3$
Reparameterization for cliques

$w_{123}(b,d,f) \rightarrow 2$

$w_{234}(d,f,v) \rightarrow 1$

$w_{\emptyset} = 4$

Propagating C1 before C2
Reparameterization for cliques

Propagating \( C_2 \) before \( C_1 \)

Select the clique with the largest lower bound increase first
Experimental Results

(* Including bounded clique detection with Bron-Kerbosch algorithm in preprocessing *)

<table>
<thead>
<tr>
<th>problem</th>
<th>TOULBAR2</th>
<th>TOULBAR2^{clq}</th>
<th>CPLEX</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>solv.</td>
<td>time</td>
<td>solv.</td>
</tr>
<tr>
<td>Auction/path</td>
<td>86</td>
<td>59</td>
<td>86</td>
</tr>
<tr>
<td>Auction/sched</td>
<td>84</td>
<td>110</td>
<td>84</td>
</tr>
<tr>
<td>MaxClique</td>
<td>31</td>
<td>1871</td>
<td>37</td>
</tr>
<tr>
<td>SPOT5</td>
<td>4</td>
<td>2884</td>
<td>6</td>
</tr>
</tbody>
</table>
Complete tree search methods

Hybrid search
DFS

Depth First

Diagram of a tree:

```
  1
  / \   \
 2   9
  |   /   \   \
 3   6   ...   ...
  |   |   \   \
 4   5   7   8
```
DFS

Depth First Advantages

- Incrementality
Depth First

Advantages

- Incrementality
- Anytime (sort of)
DFS

Depth First

Advantages

- Incrementality
- Anytime (sort of)

But

- Thrashing
DFS

Depth First Advantages

- Incrementality
- Anytime (sort of)

But

- Thrashing
- No global lower bounds
BFS

Best first

- Memory requirements
BFS

Best first
- Memory requirements
- No incrementality or even greater memory cost
BFS

Best first
- Memory requirements
- No incrementality or even greater memory cost
- Not anytime
BFS

Best first
• Memory requirements
• No incrementality or even greater memory cost
• Not anytime
but
• Theoretical guarantees
BFS

Best first
- Memory requirements
- No incrementality or even greater memory cost
- Not anytime

but
- Theoretical guarantees
- Global lower bounds
HBFS

BFS with DFS probes*
HBFS

BFS with DFS probes*  
  • Improved anytime behavior
HBFS

BFS with DFS probes*
- Improved anytime behavior
- Incrementality without memory overhead
HBFS

BFS with DFS probes*
- Improved anytime behavior
- Incrementality without memory overhead
- Lower bounds
HBFS

BFS with DFS probes*

- Improved anytime behavior
- Incrementality without memory overhead
- Lower bounds
- Some of the advantages of restarting
HBFS

BFS with DFS probes*

- Improved anytime behavior
- Incrementality without memory overhead
- Lower bounds
- Some of the advantages of restarting

* With adaptive heuristic for probe size
# Benchmark

- **MRF**: Probabilistic Inference Challenge 2011 (uai format)
- **CVPR**: Computer Vision and Pattern Recognition OpenGM2 (uai)
- **CFN**: MaxCSP 2008 Competition and CFLib (wcsp format)
- **WPMS**: Weighted Partial MaxSAT Evaluation 2013 (wcnf format)
- **CP**: MiniZinc Challenge 2012 & 2013 (minizinc format)

## Number of instances and their total compressed (gzipped) size:

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Nb.</th>
<th>UAI</th>
<th>WCSP</th>
<th>LP(direct)</th>
<th>LP(tuple)</th>
<th>WCNF(direct)</th>
<th>WCNF(tuple)</th>
<th>MINIZINC</th>
</tr>
</thead>
<tbody>
<tr>
<td>MRF</td>
<td>319</td>
<td>187MB</td>
<td>475MB</td>
<td>2.4G</td>
<td>2.0GB</td>
<td>518MB</td>
<td>2.9GB</td>
<td>473MB</td>
</tr>
<tr>
<td>CVPR</td>
<td>1461</td>
<td>430MB</td>
<td>557MB</td>
<td>9.8GB</td>
<td>11GB</td>
<td>3.0GB</td>
<td>15GB</td>
<td>N/A</td>
</tr>
<tr>
<td>CFN</td>
<td>281</td>
<td>43MB</td>
<td>122MB</td>
<td>300MB</td>
<td>3.5GB</td>
<td>389MB</td>
<td>5.7GB</td>
<td>69MB</td>
</tr>
<tr>
<td>MaxCSP</td>
<td>503</td>
<td>13MB</td>
<td>24MB</td>
<td>311MB</td>
<td>660MB</td>
<td>73MB</td>
<td>999MB</td>
<td>29MB</td>
</tr>
<tr>
<td>WPMS</td>
<td>427</td>
<td>N/A</td>
<td>387MB</td>
<td>433MB</td>
<td>N/A</td>
<td>717MB</td>
<td>N/A</td>
<td>631MB</td>
</tr>
<tr>
<td>CP</td>
<td>35</td>
<td>7.5MB</td>
<td>597MB</td>
<td>499MB</td>
<td>1.2GB</td>
<td>378MB</td>
<td>1.9GB</td>
<td>21KB</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>3026</strong></td>
<td><strong>0.68G</strong></td>
<td><strong>2.2G</strong></td>
<td><strong>14G</strong></td>
<td><strong>18G</strong></td>
<td><strong>5G</strong></td>
<td><strong>27G</strong></td>
<td><strong>1.2G</strong></td>
</tr>
</tbody>
</table>

[http://genoweb.toulouse.inra.fr/~degivry/evalgm](http://genoweb.toulouse.inra.fr/~degivry/evalgm)
Normalized lower and upper bounds on 1208 difficult instances as time passes.
Results exploiting cliques

 normalized lower and upper bounds on 252 instances as time passes

(CP17)
Limited Discrepancy Search (Ginsberg 95)

- Small example with 3 variables and 2 values per domain
Limited Discrepancy Search

- Small example with 3 variables and 2 values per domain
Limited Discrepancy Search (Ginsberg 95)
\[ l_{\text{max}} = n \cdot (d - 1) \quad \text{in this case, } l_{\text{max}} = 3 \cdot (2 - 1) = 3 \]

Limited Discrepancy Search (Ginsberg 95)

In practice, it occurs before \( l_{\text{max}} \) thanks to bounding and pruning.
Variable Neighborhood Search (Hansen 97)

1. Select randomly and uniformly a local set of $k$ variables

LDS SEARCH with given discrepancy

3. If $E' < E$ then intensification : $S = S'$ and $k = k_{init}$ (small)
Else diversification : $k = k + 1$
UDGVNS: Exploration of both k and l dimensions

LDS

\[ \text{l=0} \quad \text{l=1} \quad \text{l=2} \quad \cdots \quad \text{l=}_{\text{max}} \]

- \( k_{\text{init}} = 4 \)
- \( k = 5 \)
- \( k = \ldots \)
- \( k_{\text{max}} \)

DSF
Step 1: Initial solution

Greedy assignment
NEW SOLUTION WITH BETTER $E_{\text{best}}$ → RESTART
Proof of Optimality

IFF $ub = lb(\text{problem})$ can be before $k_{max}$
Proof of Optimality

In the worst case \( l \geq \max \text{ number of right branches} \)

\[ l_{\text{max}} = |x|^*(D_{\text{max}}-1) \]

Iff \( k = k_{\text{max}} = \text{problem size} \)
Proof of Optimality

In the worst case $l \geq \text{max number of right branches}$

$Iff \; k = k_{\text{max}} = \text{problem size}$
Neighborhoods using problem structure

Radio Link Frequency Assignment

CELAR SCEN-07r
(*Constraints 4*(1), 1999)

Earth Observation Satellite Management

SPOT5 #509 (*Constraints 4*(3), 1999)

Mendelian Error Detection

langladeM7 sheep pedigree
(*Constraints 13*(1), 2008)

Tag SNP Selection

HapMap chr01 \( r^2 \geq 0.8 \) #14481
(*Bioinformatics 22*(2), 2006)
Cluster visit in a topological order:

Graph treewidth ($W$)
Results

(UAI17)

CPU time (in seconds)

Number of solved instances
Parallel VNS

Unified Parallel Decomposition Guided VNS (UPDGVNS)
Results

(UAI17)

The figure shows the normalized upper bounds for various algorithms over time. The algorithms compared include UPDGVNS (30 cores), UPDGVNS (10 cores), UDGVNS, incop+toulbar2, and daoopt (1200sec setting). The x-axis represents wall-clock real time in minutes, while the y-axis represents the normalized upper bounds.
References

- Cooper et al. *Soft arc consistency revisited*. Artificial Intelligence, 2010
- Katsirelos et al. *Clique Cuts in Weighted Constraint Satisfaction*. In Proc. of CP-17, Melbourne, Australia, 2017

https://github.com/toulbar2/toulbar2