Constraint and Cost Function Networks: feasibility, optimization and learning.

Thomas Schiex

Constraint & Cost Function Networks:

Feasibility, optimization and learning

JFPC'2021

ANITI
ARTIFICIAL & NATURAL INTELLIGENCE
TOULOUSE INSTITUTE

T. Schiex (and plenty of colleagues)

Université Fédérale de Toulouse, ANITI, INRAE MIAT, Toulouse, France

June 22, 2021

INRAE
science for people, life & earth

A Constraint Network $\langle \boldsymbol{V}, \Phi \rangle$

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e Boolean functions (or constraints)
- Each $\varphi_{\boldsymbol{S}} \in \Phi$ is a truth function from $D^{\boldsymbol{S}} \to \{t, f\}$

Joint truth function

$$\Phi_{\mathcal{M}} = \bigwedge_{\varphi_{\boldsymbol{S}} \in \Phi} \varphi_{\boldsymbol{S}}$$

The Constraint Satisfaction Problem (NP-complete)

- Is it possible to make $\Phi_{\mathcal{M}} = t$?

A Constraint Network $\langle \boldsymbol{V}, \Phi \rangle$

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e Boolean functions (or constraints)
- Each $\varphi_{\boldsymbol{S}} \in \Phi$ is a truth function from $D^{\boldsymbol{S}} \rightarrow \{t, f\}$

Joint truth function

$$\Phi_{\mathcal{M}} = \bigwedge_{\varphi_{\boldsymbol{S}} \in \Phi} \varphi_{\boldsymbol{S}}$$

The Constraint Satisfaction Problem (NP-complete)

- Is it possible to make $\Phi_{\mathcal{M}} = t$?

A Constraint Network $\langle \boldsymbol{V}, \Phi \rangle$

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e Boolean functions (or constraints)
- Each $\varphi_{\boldsymbol{S}} \in \Phi$ is a truth function from $D^{\boldsymbol{S}} \rightarrow \{t, f\}$

Joint truth function

$$\Phi_{\mathcal{M}} = \bigwedge_{\varphi_{\boldsymbol{S}} \in \Phi} \varphi_{\boldsymbol{S}}$$

The Constraint Satisfaction Problem (NP-complete)

- Is it possible to make $\Phi_{\mathcal{M}} = t$?

A Constraint Network $\langle \boldsymbol{V}, \Phi \rangle$

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e Boolean functions (or constraints)
- Each $\varphi_{\boldsymbol{S}} \in \Phi$ is a truth function from $D^{\boldsymbol{S}} \rightarrow \{t, f\}$

Joint truth function

$$\Phi_{\mathcal{M}} = \bigwedge_{\varphi_{\boldsymbol{S}} \in \Phi} \varphi_{\boldsymbol{S}}$$

The Constraint Satisfaction Problem (NP-complete)

- Is it possible to make $\Phi_{\mathcal{M}} = t$?

A Constraint Network $\langle \boldsymbol{V}, \Phi \rangle$

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e Boolean functions (or constraints)
- Each $\varphi_{\boldsymbol{S}} \in \Phi$ is a truth function from $D^{\boldsymbol{S}} \rightarrow \{t, f\}$

Joint truth function

$$\Phi_{\mathcal{M}} = \bigwedge_{\varphi_{\boldsymbol{S}} \in \Phi} \varphi_{\boldsymbol{S}}$$

The Constraint Satisfaction Problem (NP-complete)

- Is it possible to make $\Phi_{\mathcal{M}} = t$?

Languages for domains and constraints

- Constraint Networks: Boolean tables (tensors) for domains and constraints
- Constraint Programming: interval variables, specialized constraints, control

Tables (or tensors) for $\varphi_{\mathbf{S}}$

- A multidimensional table with a Boolean for every tuple in $D^{\mathbf{S}}$
- Says if it is authorized (t) or not (f)

Pairwise difference (3 values)

$$\begin{bmatrix} f & t & t \\ t & f & t \\ t & t & f \end{bmatrix}$$

Languages for domains and constraints

- Constraint Networks: Boolean tables (tensors) for domains and constraints
- Constraint Programming: interval variables, specialized constraints, control

Global constraints

- Names for specific (useful) constraints

Most famous

AllDifferent$_S$

Languages for domains and constraints

- Constraint Networks: Boolean tables (tensors) for domains and constraints
- Constraint Programming: interval variables, specialized constraints, control

Application domains: NP and beyond

Excel at the analysis of complex perfectly known systems

Digital circuit verification, scheduling and other resource management problems, planning, software verification, theorem proving,…

Biology?

Cost Function Network $\langle \boldsymbol{V}, \Phi, k \rangle$

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e integer cost functions
- Each $\varphi_{\boldsymbol{S}} \in \Phi$ is a numerical function bounded by k (finite or infinite)

Joint cost function using $a +^k b = \min(a + b, k)$

$$\Phi_{\mathcal{M}} = \sum_{\varphi_{\boldsymbol{S}} \in \Phi}^{k} \varphi_{\boldsymbol{S}}$$

The Weighted Constraint Satisfaction Problem (decision NP-complete)

- What is the minimum of $\Phi_{\mathcal{M}}$?

ANITI INRAE

Cost Function Network $\langle \boldsymbol{V}, \Phi, k \rangle$

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e integer cost functions
- Each $\varphi_{\boldsymbol{S}} \in \Phi$ is a numerical function bounded by k (finite or infinite)

Joint cost function using $a +^k b = \min(a + b, k)$

$$\Phi_{\mathcal{M}} = \sum_{\varphi_{\boldsymbol{S}} \in \Phi}^{k} \varphi_{\boldsymbol{S}}$$

The Weighted Constraint Satisfaction Problem (decision NP-complete)

- What is the minimum of $\Phi_{\mathcal{M}}$?

Cost Function Network $\langle \boldsymbol{V}, \Phi, k \rangle$

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e integer cost functions
- Each $\varphi_{\boldsymbol{S}} \in \Phi$ is a numerical function bounded by k (finite or infinite)

Joint cost function using $a +^k b = \min(a + b, k)$

$$\Phi_{\mathcal{M}} = \sum_{\varphi_{\boldsymbol{S}} \in \Phi}^{k} \varphi_{\boldsymbol{S}}$$

The Weighted Constraint Satisfaction Problem (decision NP-complete)

- What is the minimum of $\Phi_{\mathcal{M}}$?

ANITI INRAE

Cost Function Network $\langle \boldsymbol{V}, \Phi, k\rangle$

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e integer cost functions
- Each $\varphi_{\boldsymbol{S}} \in \Phi$ is a numerical function bounded by k (finite or infinite)

Joint cost function using $a +^k b = \min(a+b, k)$

$$\Phi_{\mathcal{M}} = \sum_{\varphi_{\boldsymbol{S}} \in \Phi}^{k} \varphi_{\boldsymbol{S}}$$

The Weighted Constraint Satisfaction Problem (decision NP-complete)

- What is the minimum of $\Phi_{\mathcal{M}}$?

Cost Function Network $\langle \boldsymbol{V}, \Phi, k \rangle$

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e integer cost functions
- Each $\varphi_{\boldsymbol{S}} \in \Phi$ is a numerical function bounded by k (finite or infinite)

Joint cost function using $a +^k b = \min(a + b, k)$

$$\Phi_{\mathcal{M}} = \sum_{\varphi_{\boldsymbol{S}} \in \Phi}^{k} \varphi_{\boldsymbol{S}}$$

The Weighted Constraint Satisfaction Problem (decision NP-complete)

- What is the minimum of $\Phi_{\mathcal{M}}$?

Tables (or tensors) for φ_S

- A multidimensional table with a number for every tuple in D^S

Global functions

- Names for specific (useful) functions

Soft difference (3 values)

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

A useful one

Knapsack$_S$

Tables (or tensors) for φ_S

- A multidimensional table with a number for every tuple in D^S

Soft difference (3 values)

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Global functions

- Names for specific (useful) functions

A useful one

$\textsc{Knapsack}_S$

Costs and constraints

- We assume non negative integer costs
- A constraint is a cost function that maps to $\{0, k\}$
- $k = 1$ defines a pure Constraint Network

Optimum preserving operations

- scaling: $2^{63} \approx 19$ digits. Fixed decimal point numbers . . . . ok
- shifting: negative numbers and maximization . . . . ok

ANITI INRAE

Extra assumptions inside the solver w/o l.o.g.

- CFNs have all unary functions $\varphi_i, X_i \in \boldsymbol{V}$ (domains)
- CFNs have a constant function $\varphi_\varnothing$

Crucial property

$\varphi_\varnothing$ is a lower bound of the joint function $\Phi_{\mathcal{M}}$

Graph $G = (\boldsymbol{V}, \boldsymbol{E})$ with edge weight function w

- A Boolean variable X_i per vertex $i \in V$
- A cost function per edge $e = (i, j) \in E$: $\varphi_{ij} = w(i, j) \times \mathbb{1}[x_i \neq x_j]$

A simple graph

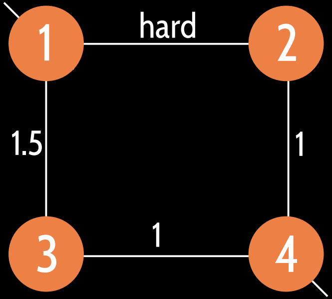
- vertices $\{1, 2, 3, 4\}$
- cut weight 1 or 1.5 $(1, 3)$
- edge $(1, 2)$ hard

Graph $G = (\boldsymbol{V}, \boldsymbol{E})$ with edge weight function w

- A Boolean variable X_i per vertex $i \in V$
- A cost function per edge $e = (i, j) \in E$: $\varphi_{ij} = w(i, j) \times \mathbb{1}[x_i \neq x_j]$

A simple graph

- vertices $\{1, 2, 3, 4\}$
- cut weight 1 or 1.5 $(1, 3)$
- edge $(1, 2)$ hard

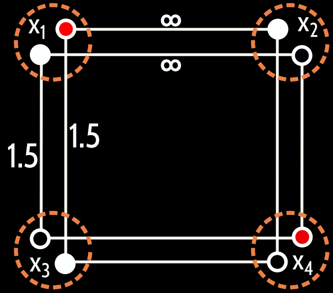


Graph $G = (\boldsymbol{V}, \boldsymbol{E})$ with edge weight function w

- A Boolean variable X_i per vertex $i \in V$
- A cost function per edge $e = (i, j) \in E : \varphi_{ij} = w(i, j) \times \mathbb{1}[x_i \neq x_j]$

A simple graph

- vertices $\{1, 2, 3, 4\}$
- cut weight 1 or 1.5 $(1, 3)$
- edge $(1, 2)$ hard



Min-CUT on 4 variables

```
{
   "problem" :{"name": "MinCut", "mustbe": "<100.0"},
     variables: {"x1": ["l"], "x2": ["l","r"],
                 "x3": ["l","r"], "x4": ["r"]}
     "functions": {
       "cut12": {"scope": ["x1","x2"], "costs": [0.0, 100.0, 100.0, 0.0]},
       "cut13": {"scope": ["x1","x3"], "costs": [0.0,1.5,1.5,0.0]},
       "cut23": {"scope": ["x2","x3"], "costs": [0.0,1.0,1.0,0.0]},
       "cut34": {"scope": ["x3","x4"], "costs": [0.0,1.0,1.0,0.0]}
 }
```

Min-CUT on 4 variables

```
import pytoulbar2
myCFN = pytoulbar2.CFN(100,1)   # ub, resolution (optional)
for i in range(4):
    myCFN.AddVariable("x"+str(i+1),["l", "r"]) # returns an index
myCFN.AddFunction(["x1"],[0,100])
myCFN.AddFunction(["x4"],[100,0])
myCFN.AddFunction(["x1","x3"], [0,1.5,1.5,0])
...
sol = myCFN.Solve()  # returns a triple (sol, cost, _)
```

Definition

- Variables X_{ij} for cell (i, j) has domain $\{1, \cdots, 9\}$
- Set R_i (resp. C_j) contains all variables of row i (resp. column j)
- Set S_i contains all variables in sub-cell i
- There is an All-Different constraint on each of these
- or a clique of pairwise different constraints

Example

Let's have a look at the pytoulbar2 code.

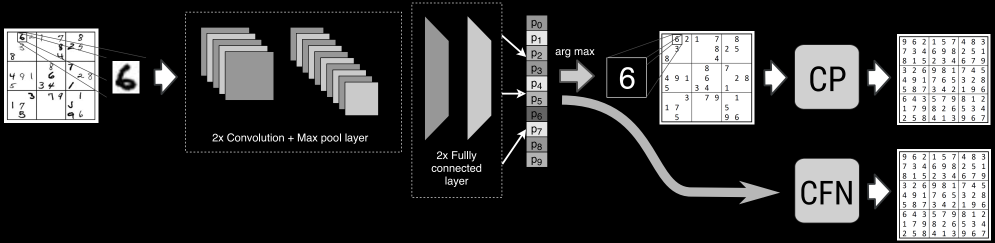
```
myCFN = pytoulbar2.CFN(1)   # k = 1, so CSP

for i in range(9):
  for j in range(9):
      vIdx = myCFN.AddVariable("X"+str(i+1)+"."+str(j+1),range(1,10))
      columns[j].append(vIdx)
      rows[i].append(vIdx)
      cells[(i//3)*3+(j//3)].append(vIdx)

for scope in rows+columns+cells:
  addCliqueAllDiff(myCFN,scope)   # Adds a clique of pairwise difference

for v,h in enumerate(grid):
  if h:  myCFN.AddFunction([v],[0 if i == h else 1 for i in range(1,10)])
```

Numbers: interfacing with DL



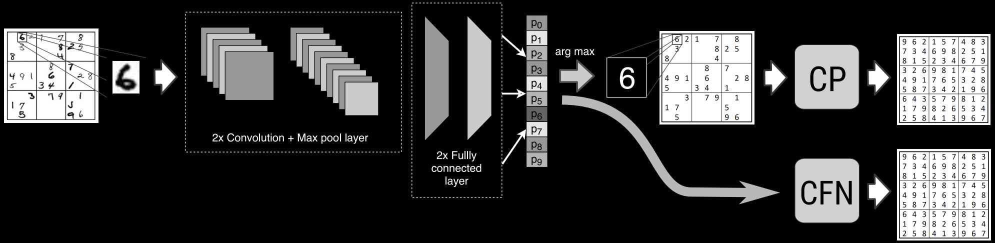
The Boolean way

Thanks to Tias Gun for the picture above

1. Assign the cell variable with the prediction
2. LeNet has 99.2% accuracy, SAT-Net dataset 36.2 hints (avg): . . . . . . 74.7% max. accuracy

The Numbers way

1. Add LeNet output tensor (negated) as a cost function
2. $(\min \sum -\log) \equiv (\max \prod)$ probabilities . . . . >99% acc.



The Boolean way

Thanks to Tias Gun for the picture above

1. Assign the cell variable with the prediction
2. LeNet has 99.2% accuracy, SAT-Net dataset 36.2 hints (avg): . . . . . . 74.7% max. accuracy

The Numbers way

1. Add LeNet output tensor (negated) as a cost function
2. $(\min \sum -\log) \equiv (\max \prod)$ probabilities . . . . >99% acc.

```
myCFN = pytoulbar2.CFN(1000000,6)

for i in range(9):
  for j in range(9):
      vIdx = myCFN.AddVariable("X"+str(i+1)+"."+str(j+1),range(1,10))
      columns[j].append(vIdx)
      rows[i].append(vIdx)
      cells[(i//3)*3+(j//3)].append(vIdx)

for scope in rows+columns+cells:
  addCliqueAllDiff(myCFN,scope)   # Adds a clique of pairwise difference

for v, h in enumerate(grid):
    if h: myCFN.AddFunction([v],-MNIST_output(csol,v,h))
```

ANITI INRAE

CFN compared to a COP approach[1]

- COP (OR-Tools) + global All-Different
- CFN (toulbar2) + pairwise differences

Tight links with (I)LP

Let's look at the primal connection

[1] Maxime Mulamba et al. "Hybrid Classification and Reasoning for Image-based Constraint Solving". In: *Proc. of CPAIOR'20, also in arXiv preprint arXiv:2003.11001*. 2020, pp. 364–380.

CFN compared to a COP approach[1]

- COP (OR-Tools) + global All-Different . . . . 0.79"
- CFN (toulbar2) + pairwise differences . . . . 0.05"

Tight links with (I)LP

Let's look at the primal connection

[1] Maxime Mulamba et al. "Hybrid Classification and Reasoning for Image-based Constraint Solving". In: *Proc. of CPAIOR'20, also in arXiv preprint arXiv:2003.11001.* 2020, pp. 364–380.

CFN compared to a COP approach[1]

- COP (OR-Tools) + global All-Different . . . . 0.79"
- CFN (toulbar2) + pairwise differences . . . . 0.05"
- 99.6% of all problems are solved backtrack-free by toulbar2

Tight links with (I)LP

Let's look at the primal connection

[1] Maxime Mulamba et al. "Hybrid Classification and Reasoning for Image-based Constraint Solving". In: *Proc. of CPAIOR'20, also in arXiv preprint arXiv:2003.11001.* 2020, pp. 364–380.

CFN compared to a COP approach[1]

- COP (OR-Tools) + global All-Different . . . . 0.79"
- CFN (toulbar2) + pairwise differences . . . . 0.05"
- 99.6% of all problems are solved backtrack-free by toulbar2
- CFN bounds way tighter than COP bounds [LL12]

Tight links with (I)LP

Let's look at the primal connection

[1] Maxime Mulamba et al. "Hybrid Classification and Reasoning for Image-based Constraint Solving". In: *Proc. of CPAIOR'20, also in arXiv preprint arXiv:2003.11001*. 2020, pp. 364–380.

ANITI INRAE

The "local polytope" [Sch76; Kos99; Wer07] (without eq. (1))

Minimize $\sum_{i,a} \varphi_i(a) \cdot x_{ia} + \sum_{\substack{\varphi_{ij} \in \Phi \\ a \in D^i, b \in D^j}} \varphi_{ij}(a,b) \cdot y_{iajb}$ such that

$$\sum_{a \in D^i} x_{ia} = 1 \qquad \forall i \in \{1, \ldots, n\}$$

$$\sum_{b \in D^j} y_{iajb} = x_{ia} \qquad \forall \varphi_{ij} \in \Phi, \forall a \in D^i$$

$$\sum_{a \in D^i} y_{iajb} = x_{jb} \qquad \forall \varphi_{ij} \in \Phi, \forall b \in D^j$$

$$x_{ia} \in \{0, 1\} \qquad \forall i \in \{1, \ldots, n\} \quad (1)$$

$nd + ed^2$ variables, $n + 2ed$ constraints: a strong but expensive bound

Systematic tree search — Time $O(d^n)$, linear space

- If all $|D^X| = 1$ obvious minimum — update k to $\Phi_{\mathcal{M}}(\boldsymbol{v})$
- Else choose $X \in \boldsymbol{V}$ s.t. $|D^X| > 1$ and $u \in D^X$ and reduce to
  1. one query where we set $X = u$
  2. one where u is removed from D^X
- Return the minimum

Optimization — Branch and Bound [LW66]

If the $\underbrace{\text{local lower bound}}_{\varphi_\varnothing}$ reaches the $\underbrace{\text{global upper bound}}_{k}$ — Prune!

Systematic tree search — Time $O(d^n)$, linear space

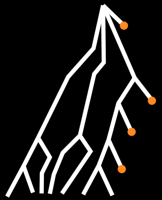
- If all $|D^X| = 1$ obvious minimum — update k to $\Phi_{\mathcal{M}}(\boldsymbol{v})$
- Else choose $X \in \boldsymbol{V}$ s.t. $|D^X| > 1$ and $u \in D^X$ and reduce to
  1. one query where we set $X = u$
  2. one where u is removed from D^X
- Return the minimum

Optimization — Branch and Bound [LW66]

If the $\underbrace{\text{local lower bound}}_{\varphi_\varnothing}$ reaches the $\underbrace{\text{global upper bound}}_{k}$ — Prune!

Hybrid Best First Search [All+15] — **Anyspace**

- Uses Depth-First Search for a bounded amount of backtracks
- Pending nodes are pushed onto a list of Open nodes
- The next DFS starts from the best Open node
- Tree-decomposition friendly (BTD [GSV06]/AND-OR search [MD09])

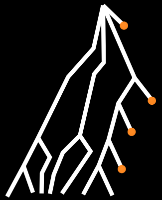


Nice properties

- Good upper bounds quickly (DFS)
- A constantly improving global lower bound (optimality gap)
- Implicit restarts, easy parallelization

Hybrid Best First Search [All+15] Anyspace

- Uses Depth-First Search for a bounded amount of backtracks
- Pending nodes are pushed onto a list of Open nodes
- The next DFS starts from the best Open node
- Tree-decomposition friendly (BTD [GSV06]/AND-OR search [MD09])



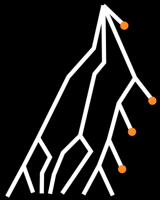
Nice properties

- Good upper bounds quickly (DFS)
- A constantly improving global lower bound (optimality gap)
- Implicit restarts, easy parallelization

Depth First (CP) or Best First (ILP)?

Hybrid Best First Search [All+15] **Anyspace**

- Uses Depth-First Search for a bounded amount of backtracks
- Pending nodes are pushed onto a list of Open nodes
- The next DFS starts from the best Open node
- Tree-decomposition friendly (BTD [GSV06]/AND-OR search [MD09])

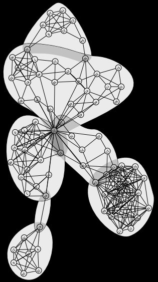


Nice properties

- Good upper bounds quickly (DFS)
- A constantly improving global lower bound (optimality gap)
- Implicit restarts, easy parallelization

Hybrid Best First Search [All+15] — Anyspace

- Uses Depth-First Search for a bounded amount of backtracks
- Pending nodes are pushed onto a list of Open nodes
- The next DFS starts from the best Open node
- Tree-decomposition friendly (BTD [GSV06]/AND-OR search [MD09])



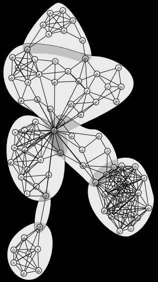
Nice properties

- Good upper bounds quickly (DFS)
- A constantly improving global lower bound (optimality gap)
- Implicit restarts, easy parallelization

Depth First (CP) or Best First (ILP)?

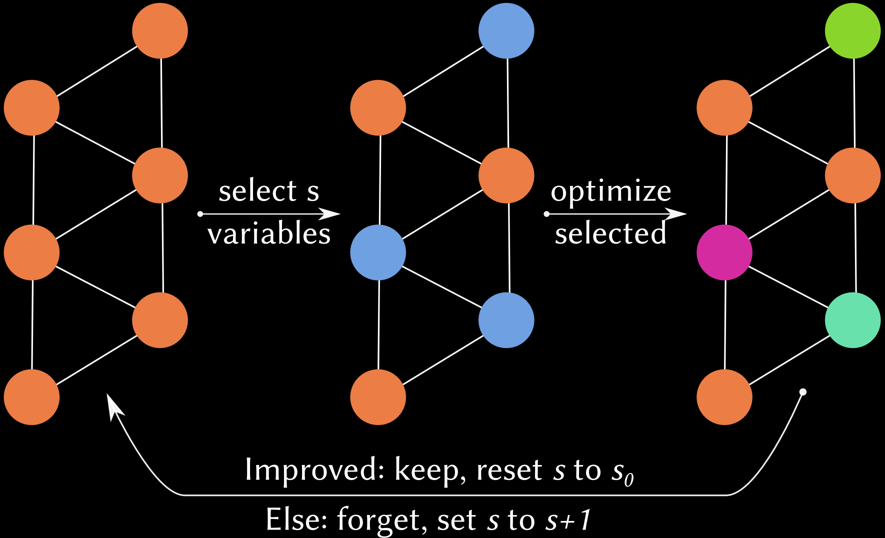
Hybrid Best First Search [All+15] — Anyspace

- Uses Depth-First Search for a bounded amount of backtracks
- Pending nodes are pushed onto a list of Open nodes
- The next DFS starts from the best Open node
- Tree-decomposition friendly (BTD [GSV06]/AND-OR search [MD09])



Nice properties

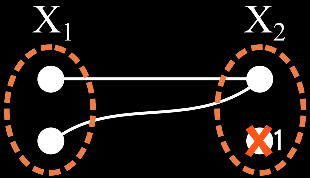
- Good upper bounds quickly (DFS)
- A constantly improving global lower bound (optimality gap)
- Implicit restarts, easy parallelization


select s
variables
optimize
selected
Improved: keep, reset *s* to s_0
Else: forget, set *s* to *s+1*

Presentation Outline

Filtering by Arc Consistency (support)

A value $u \in D^i$ with no value $v \in D^j$ such that $\varphi_{ij}(u, v) = 0$ can be deleted, leaving the problem equivalent.

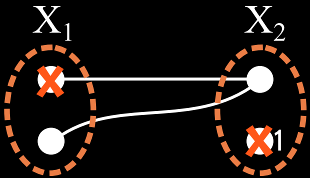


Properties

- Combine φ_{ij} and φ_j
- Project on X_i
- Combine with φ_i
- Unique fixpoint (monotonic), polynomial time (inconsistency detection)

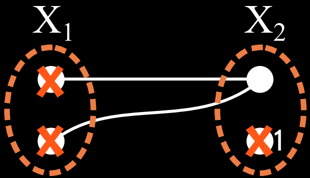
Filtering by Arc Consistency (support)

A value $u \in D^i$ with no value $v \in D^j$ such that $\varphi_{ij}(u, v) = 0$ can be deleted, leaving the problem equivalent.



Properties

- Combine φ_{ij} and φ_j
- Project on X_i
- Combine with φ_i
- Unique fixpoint (monotonic), polynomial time (inconsistency detection)



Filtering by Arc Consistency (support)

A value $u \in D^i$ with no value $v \in D^j$ such that $\varphi_{ij}(u, v) = 0$ can be deleted, leaving the problem equivalent.

Properties

- Combine φ_{ij} and φ_j
- Project on X_i
- Combine with φ_i
- Unique fixpoint (monotonic), polynomial time (inconsistency detection)

General CFN case [Sch00; LS03; LS04; CS04; Coo+10]

Obvious issue

One cannot add functions to the CFN: loss of equivalence, meaningless result

Equivalence Preserving Transformations with $-^k$ $(\alpha -^k \beta) \equiv ((\alpha = k) \,?\, k : \alpha - \beta)$

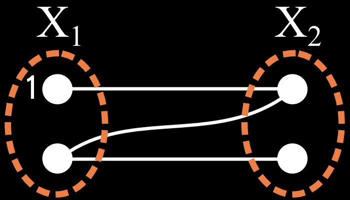
- Add the projection to φ_j with $+^k$
- Subtract it from its source using $-^k$

Obvious issue

One cannot add functions to the CFN: loss of equivalence, meaningless result

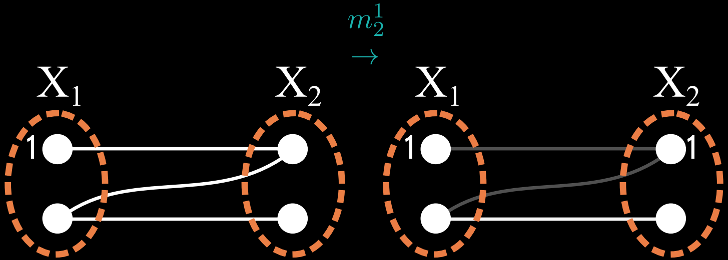
Equivalence Preserving Transformations with $-^k$ $(\alpha -^k \beta) \equiv ((\alpha = k)\ ?\ k : \alpha - \beta)$

- Add the projection to φ_j with $+^k$
- Subtract it from its source using $-^k$



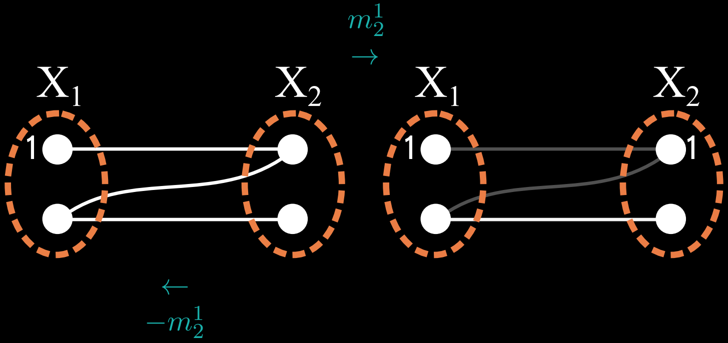
(Loss of) properties

Preserves equivalence but non-monotonic and fixpoints may be non unique (or may not exist)



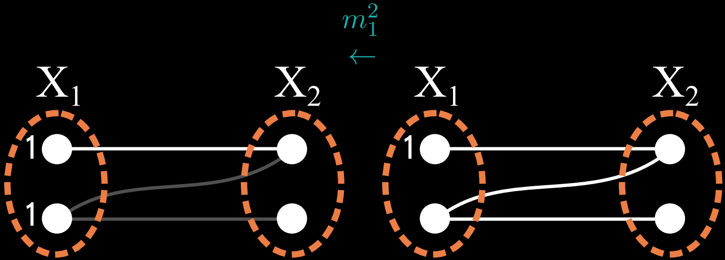
(Loss of) properties

Preserves equivalence but non-monotonic and fixpoints may be non unique (or may not exist)



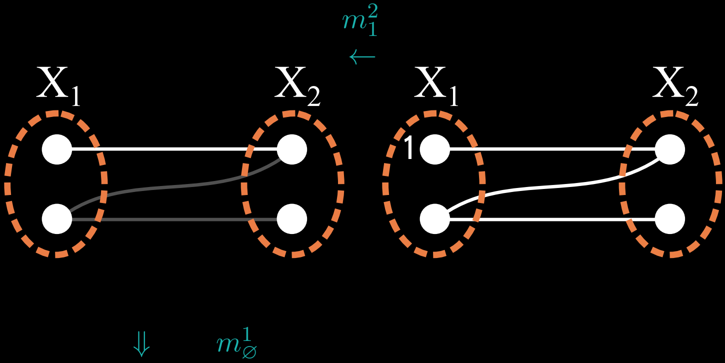
(Loss of) properties

Preserves equivalence but non-monotonic and fixpoints may be non unique (or may not exist)



(Loss of) properties

Preserves equivalence but non-monotonic and fixpoints may be non unique (or may not exist)



$\varphi_\varnothing = 1$

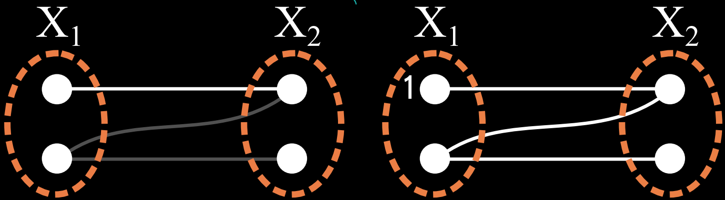
(Loss of) properties

Preserves equivalence but non-monotonic and fixpoints may be non unique (or may not exist)

Example with $k > 1$

m_1^2

$\leftarrow$



$\Downarrow \quad m_{\varnothing}^1$

$\varphi_{\varnothing} = 1$

(Loss of) properties

Preserves equivalence but non-monotonic and fixpoints may be non unique (or may not exist)

Many way to avoid loops (enforce fixpoint existence)

The many "soft ACs" — One paper to read: [Coo+10]

- NC+AC+DAC (FDAC): binary & unary (+ direction)[Sch00; Lar02; Coo03] — Full Supports
- +Existential AC: EDAC, a star (variable incident functions) [Lar+05] — EAC supports
- +Virtual AC: any spanning tree [Coo+08; Coo+10] — VAC supports

Supports provide value ordering heuristics

- EAC: $\varphi_i(u) = 0$ can be extended for free on X_i's star
- VAC: $\varphi_i(u) = 0$ can be extended for free on any spanning tree [Kol06; Coo+08; Coo+10]

NC provides reduced cost-based pruning (back-propagation)

If $(\varphi_\varnothing \overset{k}{+} \varphi_i(u)) = k$, NC deletes u

ANITI INRAE

The many "soft ACs" — One paper to read: [Coo+10]

- NC+AC+DAC (FDAC): binary & unary (+ direction)[Sch00; Lar02; Coo03] — Full Supports
- +Existential AC: EDAC, a star (variable incident functions) [Lar+05] — EAC supports
- +Virtual AC: any spanning tree [Coo+08; Coo+10] — VAC supports

Supports provide value ordering heuristics

- EAC: $\varphi_i(u) = 0$ can be extended for free on X_i's star
- VAC: $\varphi_i(u) = 0$ can be extended for free on any spanning tree [Kol06; Coo+08; Coo+10]

NC provides reduced cost-based pruning (back-propagation)

If $(\varphi_\varnothing \overset{k}{+} \varphi_i(u)) = k$, NC deletes u

ANITI INRAE

The many "soft ACs" — One paper to read: [Coo+10]

- NC+AC+DAC (FDAC): binary & unary (+ direction)[Sch00; Lar02; Coo03] — Full Supports
- +Existential AC: EDAC, a star (variable incident functions) [Lar+05] — EAC supports
- +Virtual AC: any spanning tree [Coo+08; Coo+10] — VAC supports

Supports provide value ordering heuristics

- EAC: $\varphi_i(u) = 0$ can be extended for free on X_i's star
- VAC: $\varphi_i(u) = 0$ can be extended for free on any spanning tree [Kol06; Coo+08; Coo+10]

NC provides reduced cost-based pruning (back-propagation)

If $(\varphi_\varnothing \overset{k}{+} \varphi_i(u)) = k$, NC deletes u

Properties

- Proper extension of classical NC/DAC or AC respectively ($k = 1$)
- Polynomial time, $O(ed)$ space (Generalized ACs)
- Incremental, strengthens $\varphi_\varnothing$ (NC $\leq$ AC $\leq$ FDAC $\leq$ EDAC$\leq$ VAC)
- Stronger bounds than AC in COP [LL12]

Set of rational EPTs — OSAC [Sch76; Coo07; Wer07; Coo+10]

Maximizing $\varphi_\varnothing$ is in P (local polytope dual + AC for k)

ANITI INRAE

Properties

- Proper extension of classical NC/DAC or AC respectively ($k = 1$)
- Polynomial time, $O(ed)$ space (Generalized ACs)
- Incremental, strengthens $\varphi_\varnothing$ (NC $\leq$ AC $\leq$ FDAC $\leq$ EDAC$\leq$ VAC)
- Stronger bounds than AC in COP [LL12]

Set of rational EPTs OSAC [Sch76; Coo07; Wer07; Coo+10]

Maximizing $\varphi_\varnothing$ is in P (local polytope dual + AC for k)

Variables for a binary CFN, no constraints [Sch76; Kos99; CGS07; Wer07; Coo+10]

1. u_i: amount of cost shifted from φ_i to $\varphi_\varnothing$
2. p_{ija}: amount of cost shifted from φ_{ij} to $\varphi_i(a)$
3. p_{jib}: amount of cost shifted from φ_{ij} to $\varphi_j(b)$

OSAC

Maximize $\sum_{i=1}^{n} u_i$ subject to

$$\varphi_i(a) - u_i + \sum_{(\varphi_{ij} \in C)} p_{ija} \geq 0 \qquad \forall i \in \{1, \ldots, n\}, \forall a \in D^i$$

$$\varphi_{ij}(a, b) - p_{ija} - p_{jib} \geq 0 \qquad \forall \varphi_{ij} \in C, \forall (a, b) \in D^{ij}$$

Variables for a binary CFN, no constraints [Sch76; Kos99; CGS07; Wer07; Coo+10]

1. u_i: amount of cost shifted from φ_i to $\varphi_\varnothing$
2. p_{ija}: amount of cost shifted from φ_{ij} to $\varphi_i(a)$
3. p_{jib}: amount of cost shifted from φ_{ij} to $\varphi_j(b)$

OSAC

$$\text{Maximize } \sum_{i=1}^{n} u_i \qquad \text{subject to}$$

$$\varphi_i(a) - u_i + \sum_{(\varphi_{ij} \in C)} p_{ija} \geq 0 \qquad \forall i \in \{1, \ldots, n\},\ \forall a \in D^i$$

$$\varphi_{ij}(a,b) - p_{ija} - p_{jib} \geq 0 \qquad \forall \varphi_{ij} \in C, \forall (a,b) \in D^{ij}$$

ANITI INRAE

Problems solved [Coo+10; KZ17]

- Tree-structured problems
- Permutated submodular problems (e.g. Min-Cut)

OSAC empirically too expensive compared to VAC

- CFN Arc consistencies provide fast approximate LP bounds
- and deal with constraints seamlessly

CFN Local Consistencies

Enhance CP with fast incremental approximate Linear Programming dual bounds

Problems solved [Coo+10; KZ17]

- Tree-structured problems
- Permutated submodular problems (e.g. Min-Cut)

OSAC empirically too expensive compared to VAC

- CFN Arc consistencies provide fast approximate LP bounds
- and deal with constraints seamlessly

CFN Local Consistencies

Enhance CP with fast incremental approximate Linear Programming dual bounds

Problems solved [Coo+10; KZ17]

- Tree-structured problems
- Permutated submodular problems (e.g. Min-Cut)

OSAC empirically too expensive compared to VAC

- CFN Arc consistencies provide fast approximate LP bounds
- and deal with constraints seamlessly

CFN Local Consistencies

Enhance CP with fast incremental approximate Linear Programming dual bounds

Additional algorithmic ingredients

- Value ordering (for free): existential or virtual supports
- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- (On the fly) variable elimination [Lar00]
- Dominance analysis (substitutability/DEE) [Fre91; Des+92; DPO13; All+14]
- Function decomposition [Fav+11]
- Some global cost functions (weighted Regular, All-Diff, Among...) [LL12; All+16]
- Incremental solving, guaranteed diverse solutions [Ruf+19]
- Unified (Parallel) Decomposition Guided VNS/LDS (UPDGVNS [Oua+20])

More information

github.com/toulbar2/toulbar2 miat.inrae.fr/toulbar2

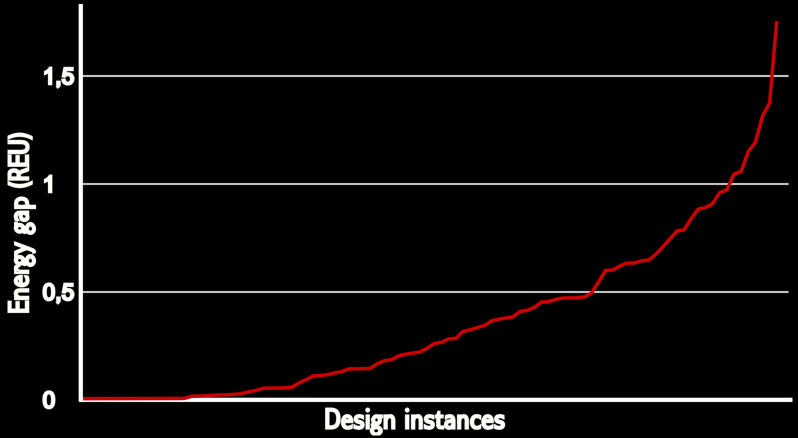
VAC vs. LP on Protein design problems

CPLEX V12.4.0.0

```
Problem '3e4h.LP' read.
Root relaxation solution time =  811.28 sec.
...
MIP - Integer optimal solution:  Objective =  150023297067
Solution time =  864.39 sec.
```

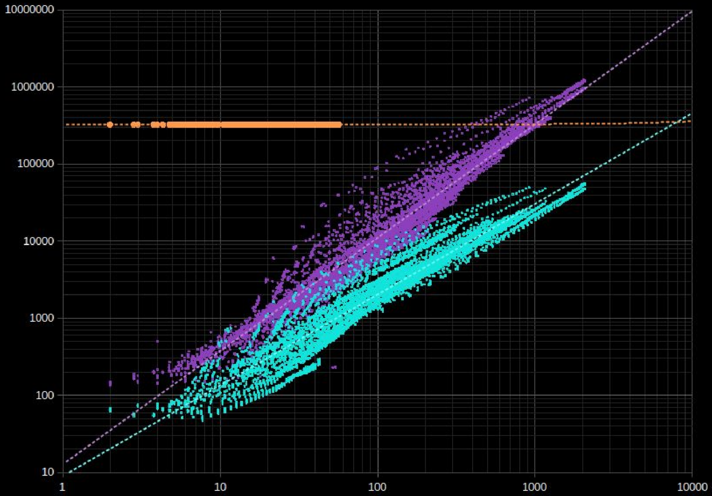
tb2 and VAC (AC3 based)

```
loading CFN file: 3e4h.wcsp
Lb after VAC: 150023297067
Preprocessing time: 9.13 seconds.
Optimum: 150023297067 in 129 backtracks, 129 nodes and 9.38 seconds.
```



Optimality gap of the Simulated annealing solution as problems get harder

[2] David Simoncini et al. "Guaranteed Discrete Energy Optimization on Large Protein Design Problems". In: *Journal of Chemical Theory and Computation* 11.12 (2015), pp. 5980–5989. DOI: 10.1021/acs.jctc.5b00594.



DWave approximations *kcal/mol*

gap > 1.16 90% of the time > 4.35, 50% of the time > 8.45, 10% of the time

Kind words from Protein Designers[3]

The Toulbar[2] package for WCSPs significantly improved the state-of-the-art efficiency for protein design in the discrete pairwise model.

Kind words from OpenGM2 developpers (image processing)

"ToulBar2 variants were superior to CPLEX variants in all our tests"[4]

[3] Mark A Hallen and Bruce R Donald. "Protein design by provable algorithms". In: *Communications of the ACM* 62.10 (2019), pp. 76–84.

[4] Stefan Haller, Paul Swoboda, and Bogdan Savchynskyy. "Exact MAP-Inference by Confining Combinatorial Search with LP Relaxation". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

Kind words from Protein Designers[3]

The Toulbar[2] package for WCSPs significantly improved the state-of-the-art efficiency for protein design in the discrete pairwise model.

Kind words from OpenGM2 developpers (image processing)

"ToulBar2 variants were superior to CPLEX variants in all our tests"[4]

[3] Mark A Hallen and Bruce R Donald. "Protein design by provable algorithms". In: *Communications of the ACM* 62.10 (2019), pp. 76–84.

[4] Stefan Haller, Paul Swoboda, and Bogdan Savchynskyy. "Exact MAP-Inference by Confining Combinatorial Search with LP Relaxation". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

Data mining, bioinformatics

Given a matrix of arbitrary real numbers, find a subset C of columns and R of rows such that the sum of numbers in the submatrix is maximized.

Dedicated global constraint

Presented in [BSD17; Der+19], dominates MILP and MIQCP.

Data mining, bioinformatics

Given a matrix of arbitrary real numbers, find a subset C of columns and R of rows such that the sum of numbers in the submatrix is maximized.

Dedicated global constraint

Presented in [BSD17; Der+19], dominates MILP and MIQCP.

```
def generate_model(path):
  m = pandas.read_csv(path, sep='\t', header=None)
  r, c = m.shape
  model = pytoulbar2.CFN(100000, 10, True)
  for i in range(r):
    model.AddVariable("R"+str(i), ["out", "in"])
  for j in range(c):
    model.AddVariable("C"+str(j), ["out", "in"])
  for i in range(r):
    for j in range(c):
      model.AddFunction(["R"+str(i), "C"+str(j)], [0.0, 0.0, 0.0, -m[j][i]])
  return model

(solution,, cost, _) = generate_model(sys.argv[1]).Solve()
```

ANITI INRAE

The Global Constraint author

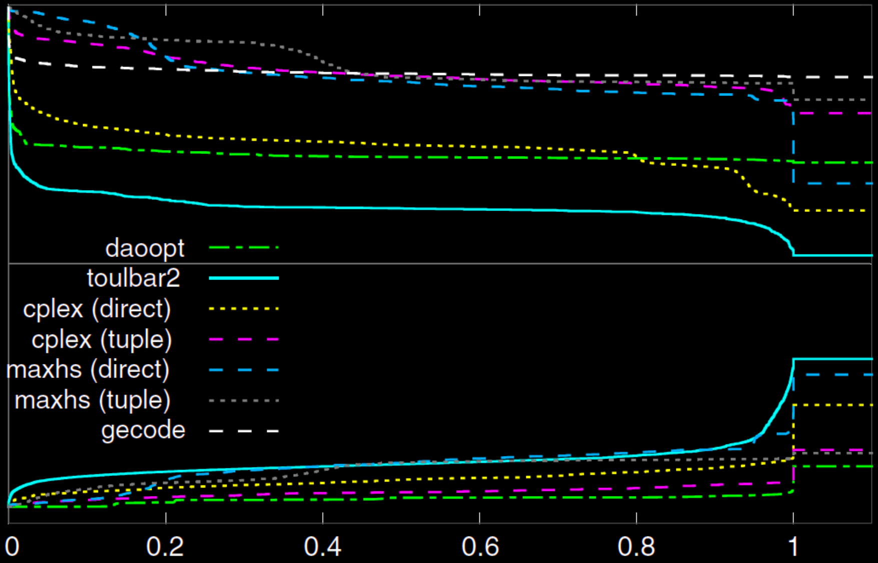
Je n'ai pas vraiment trouvé de cas [...] défavorable pour toulbar2.

3026 instances of various origins genoweb.toulouse.inra.fr/~degivry/evalgm

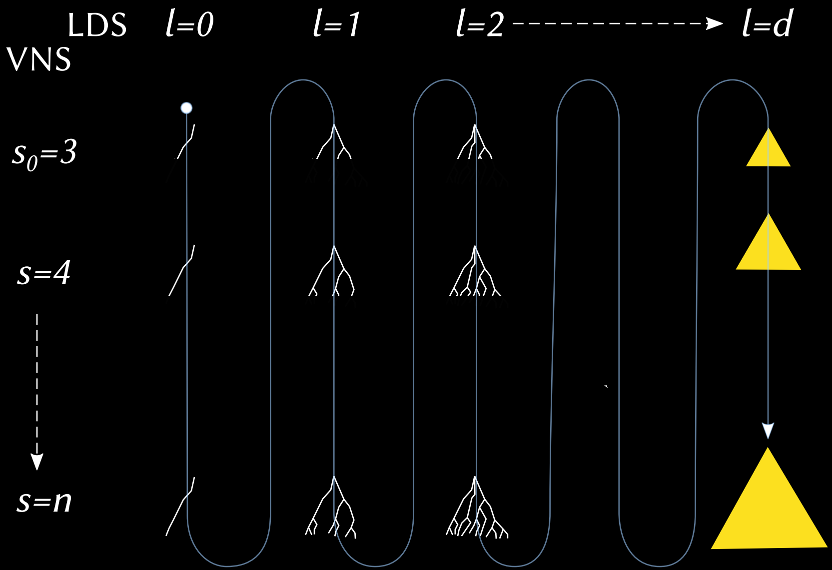
- MRF: Probabilistic Inference Challenge 2011
- CVPR: Computer Vision & Pattern Recognition OpenGM2
- CFN: Cost Function Library (CELAR, SPOT5, bioinformatics)
- MaxCSP: MaxCSP 2008 competition
- WPMS: Weighted Partial MaxSAT evaluation 2013
- CP: MiniZinc challenge 2012/13 (decomposable)

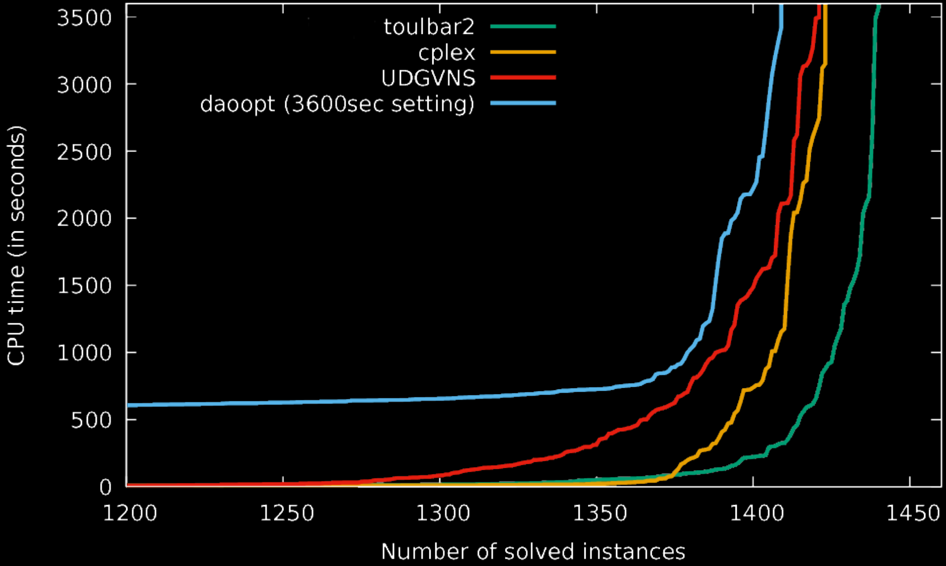
Benchmark	Nb.	UAI	WCSP	LP(direct)	LP(tuple)	WCNF(direct)	WCNF(tuple)	MINIZINC
MRF	319	187MB	475MB	2.4G	2.0GB	518MB	2.9GB	473MB
CVPR	1461	430MB	557MB	9.8GB	11GB	3.0GB	15GB	N/A
CFN	281	43MB	122MB	300MB	3.5GB	389MB	5.7GB	69MB
MaxCSP	503	13MB	24MB	311MB	660MB	73MB	999MB	29MB
WPMS	427	N/A	387MB	433MB	N/A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

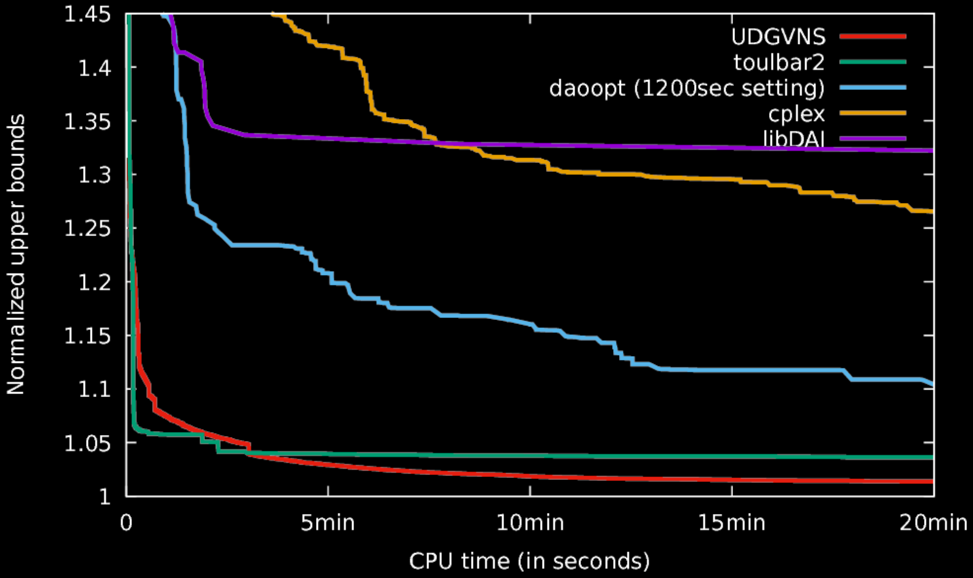
HBFS - Normalized LB and UB profiles (hard problems) [Hur+16]



Unified Decomposition Guided VNS [Oua+20; Oua+17]

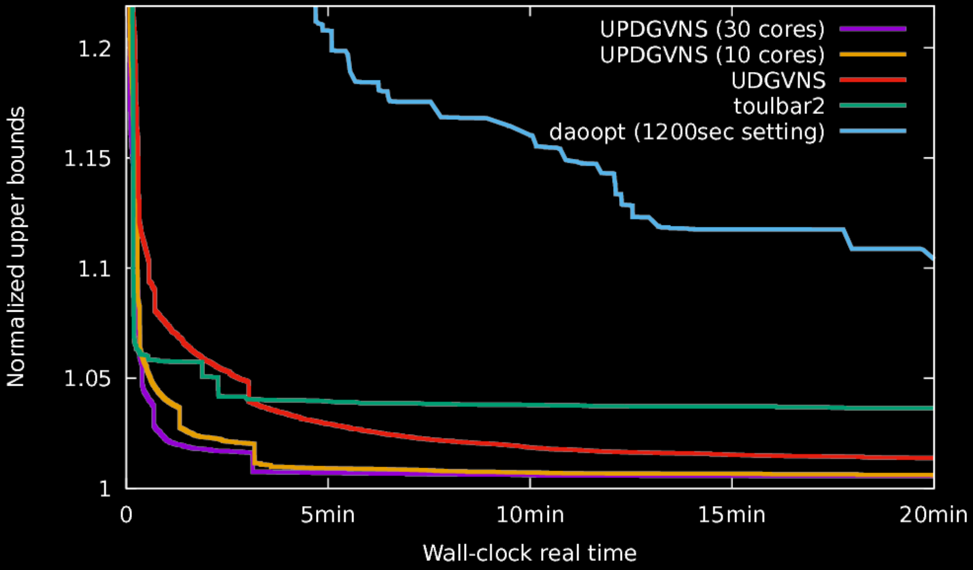



toulbar2
cplex
UDGVNS
daoopt (3600sec setting)
CPU time (in seconds)
0
500
1000
1500
2000
2500
3000
3500
1200
1250
1300
1350
1400
1450
Number of solved instances



UDGVNS
toulbar2
daoopt (1200sec setting)
cplex
libDAI
Normalized upper bounds
1.45
1.4
1.35
1.3
1.25
1.2
1.15
1.1
1.05
1
0
5min
10min
15min
20min
CPU time (in seconds)

UPDGVNS - Upper bound profiles[Oua+20]



Definition (Learning a pairwise CFN from high quality solutions)

Given:

- a set of variables V,
- a set of assignments E i.i.d. from an unknown distribution of high-quality solutions

Find a pairwise CFN $\mathcal{M}$ that can be solved to produce high-quality solutions

What does learning a CFN means exactly?

We use the language of pairwise tensors/tables

- There are at most $\frac{n(n-1)}{2}$ pairwise functions $\frac{81 \times 80}{2} = 3240$
- Each with $|D^i| \times |D^j|$ costs in $\mathbb{R}$ (differentiability) 81
- For the Sudoku, $262,440$ parameters to learn.

ANITI INRAE

Maximum likelihood estimation

- E a set of i.i.d. assignments of $\boldsymbol{V}$
- Interpret costs as energies ($\propto -\log$(probabilities))
- Maximize the probability of observing the samples in E

Maximum loglikelihood $\mathcal{M}$ on $\mathcal{M}_\ell$

$$\begin{aligned}\mathcal{L}(\mathcal{M},\boldsymbol{E}) &= \log(\textstyle\prod_{\boldsymbol{v}\in\boldsymbol{E}} P_{\mathcal{M}}(\boldsymbol{v})) = \sum_{\boldsymbol{v}\in\boldsymbol{E}} \log(P_{\mathcal{M}}(\boldsymbol{v}))\\ &= \textstyle\sum_{\boldsymbol{v}\in\boldsymbol{E}} \log(\Phi_{\mathcal{M}}(\boldsymbol{v})) - \log(Z_{\mathcal{M}})\\ &= \underbrace{\sum_{\boldsymbol{v}\in\boldsymbol{E}}(-C_{\mathcal{M}^\ell}(\boldsymbol{v}))}_{\text{-costs of } \boldsymbol{E} \text{ samples}} \underbrace{- \log(\sum_{\boldsymbol{t}\in\prod_{X\in\boldsymbol{V}} D^X} \exp(-C_{\mathcal{M}^\ell}(\boldsymbol{t})))}_{\text{Soft-Min of all assignment costs}}\end{aligned}$$

Maximum likelihood estimation

- E a set of i.i.d. assignments of $\boldsymbol{V}$
- Interpret costs as energies ($\propto -\log(\text{probabilities})$)
- Maximize the probability of observing the samples in E

Maximum loglikelihood $\mathcal{M}$ on $\mathcal{M}_\ell$

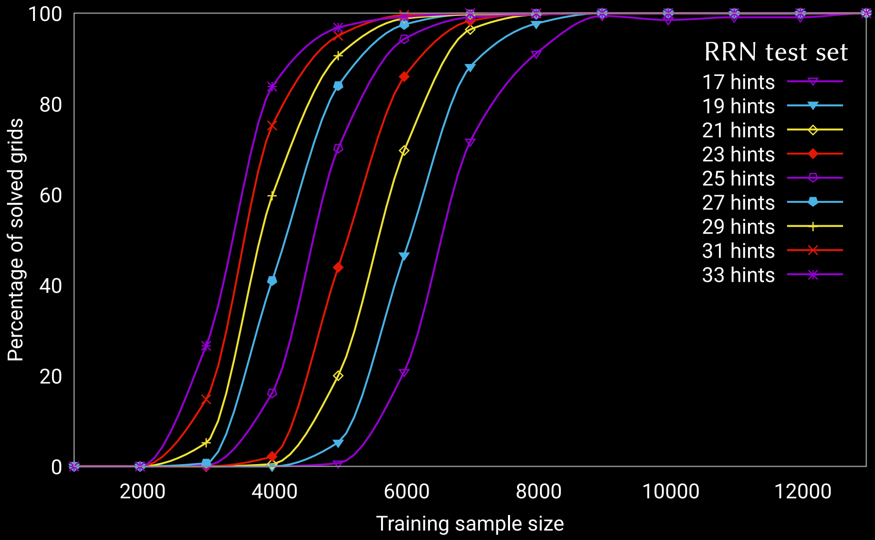
$$
\begin{aligned}
\mathcal{L}(\mathcal{M}, \boldsymbol{E}) &= \log(\textstyle\prod_{\boldsymbol{v}\in\boldsymbol{E}} P_{\mathcal{M}}(\boldsymbol{v})) = \sum_{\boldsymbol{v}\in\boldsymbol{E}} \log(P_{\mathcal{M}}(\boldsymbol{v})) \\
&= \textstyle\sum_{\boldsymbol{v}\in\boldsymbol{E}} \log(\Phi_{\mathcal{M}}(\boldsymbol{v})) - \log(Z_{\mathcal{M}}) \\
&= \underbrace{\sum_{\boldsymbol{v}\in\boldsymbol{E}} (-C_{\mathcal{M}^\ell}(\boldsymbol{v}))}_{\text{-costs of } \boldsymbol{E} \text{ samples}} \underbrace{- \log(\sum_{\boldsymbol{t}\in\prod_{X\in\boldsymbol{V}} D^X} \exp(-C_{\mathcal{M}^\ell}(\boldsymbol{t})))}_{\text{Soft-Min of all assignment costs}}
\end{aligned}
$$

Algorithms and data-sets

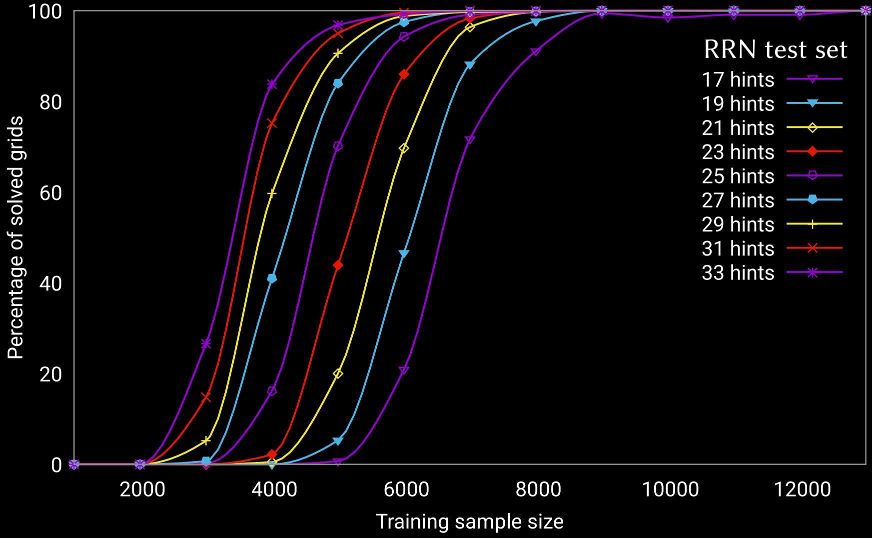
- PE-MRF [Par+17] with L1-norm Regularization
- Validation set from the SAT-Net paper[5] (36.2 hints)
- Validation set from the RRN paper[6] with 17-34 hints.

[5] Po-Wei Wang et al. "SATNet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver". In: *ICML'19 proceedings, arXiv preprint arXiv:1905.12149*. 2019.

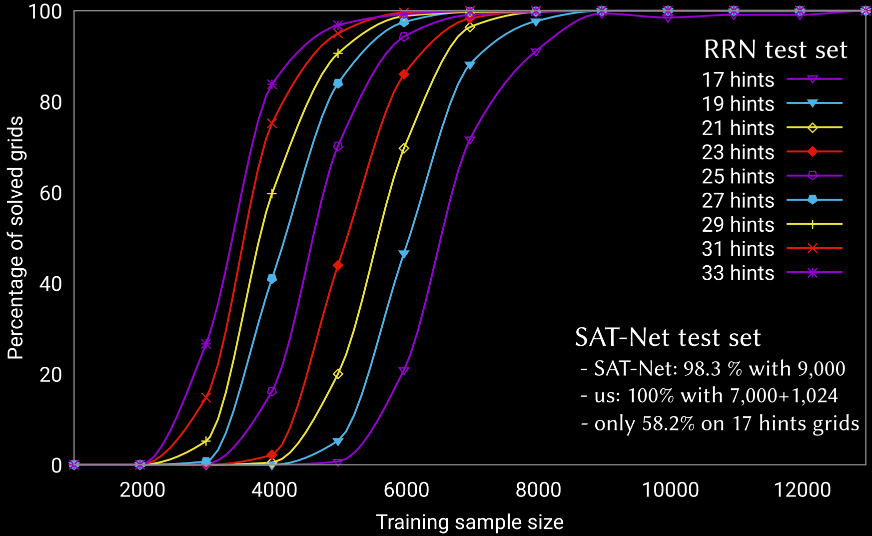
[6] Rasmus Palm, Ulrich Paquet, and Ole Winther. "Recurrent relational networks". In: *Advances in Neural Information Processing Systems*. 2018, pp. 3368–3378.



RRN: 96.6%
180,000 samples
(scopes given)



RRN: 96.6%
180,000 samples
(scopes given)



RRN: 96.6%
180,000 samples
(scopes given)

SAT-Net test set

- SAT-Net: 98.3 % with 9,000
- us: 100% with 7,000+1,024
- only 58.2% on 17 hints grids

Learning from uncertain DL output is possible

- LeNet has 99.2% accuracy on handwritten digits
- Argmax decoding: 74.7% of the learning data-set would be incorrect
- Important to accept probabilistic information as input (PE-MRF)

Comparing with SAT-Net

- SAT-Net (9,000 samples): 63.2%
- Toulbar2+PE-MRF (8,000+1,024 samples): 76.3%

Learning from images by connecting with PyTorch as before

Learning from uncertain DL output is possible

- LeNet has 99.2% accuracy on handwritten digits
- Argmax decoding: 74.7% of the learning data-set would be incorrect
- Important to accept probabilistic information as input (PE-MRF)

Comparing with SAT-Net

- SAT-Net (9,000 samples): 63.2%
- Toulbar2+PE-MRF (8,000+1,024 samples): 76.3%

Not only Sudokus of course...

See our CP2020 paper[7]

We show how it can learn user preferences and combine them with configuration constraints on Renault dataset (thanks to H. Fargier (IRIT)).

[7] Céline Brouard, Simon de Givry, and Thomas Schiex. "Pushing data into CP models using Graphical Model Learning and Solving". In: *Principles and Practice of Constraint Programming–CP 2020.* Springer, 2020.

CFN/WCSP solving has made important progress

- Fast approximate LP-bounds (tighter than COP) subsuming AC
- Free value ordering heuristics
- Reduced-cost-based filtering (cost backpropagation)
- Structure aware search with improving optimality gap

CFN can be learned from data and combined with constraints

- Shares with ILP the capacity of dealing with fine grained numerical information
- Tractable learning with probabilistic input (DL/ML connection)
- With the (adjustable) power of (exact) solvers

A lot remains to be done

Directions for improvement

- Global cost function and non monotonicity
- Interval variables and “arithmetic” filtering
- Unify CFN and COP: cost variables, multiple criteria
- Stronger incremental bounds
- Parallel search, conflict learning
- Try to minimize average tardiness in scheduling
- Improve CFN learning (sample size, (global) constraints)
- ...

Thank you all for your attention!

And to all CFN/toulbar2 contributors

S. de Givry (INRAE)
D. Allouche (INRAE)
M. Cooper (IRIT, Toulouse)
M. Sanchez (PostDoc)
G. Verfaillie (ONERA, ret.)
JP. Métivier (GREYC, Caen)
D. Simoncini (PostDoc, UT1)
P. Jégou (LSIS)
L. Loukil (GREYC)
M. Lemaître (CERT)
B. Neveu (INRIA, Sophia)

G. Katsirelos (INRAE)
M. Ruffini (PhD)
J. Larrosa (UPC, Spain)
E. Rollon (UPC, Spain)
JH. Lee (CU. Hong Kong)
S. Loudni (GREYC, Caen)
C. Viricel (PhD)
A. Ouali (GREYC)
P. Boizumault (GREYC)
L. Lobjois (CERT)
G. Trombettoni (INRIA)

M. Zytnicki (PhD, INRAE)
H. Nguyen (PhD)
F. Heras (UPC, Spain)
P. Meseguer (CSIC, Spain)
C. Bessiere (LIMM, Montpellier)
M. Fontaine (GREYC, Caen)
C. Terrioux (LSIS)
Y. Lebbah (GREYC)
Mario (CU. Hong-Kong)
B. Hurley (Insight)
...

Questions?

[All+14] David Allouche et al. “Computational protein design as an optimization problem”. In: *Artificial Intelligence* 212 (2014), pp. 59–79.

[All+15] David Allouche et al. “Anytime Hybrid Best-First Search with Tree Decomposition for Weighted CSP”. In: *Principles and Practice of Constraint Programming*. Springer. 2015, pp. 12–29.

[All+16] David Allouche et al. “Tractability-preserving transformations of global cost functions”. In: *Artificial Intelligence* 238 (2016), pp. 166–189.

[BdS20] Céline Brouard, Simon de Givry, and Thomas Schiex. “Pushing data into CP models using Graphical Model Learning and Solving”. In: *Principles and Practice of Constraint Programming–CP 2020*. Springer, 2020.

[BGS20] Céline Brouard, Simon de Givry, and Thomas Schiex. “Pushing data into CP models using Graphical Model Learning and Solving”. In: LNCS 4204 (2020).

[Bou+04] Frédéric Boussemart et al. “Boosting systematic search by weighting constraints”. In: *ECAI*. Vol. 16. 2004, p. 146.

[BSD17] Vincent Branders, Pierre Schaus, and Pierre Dupont. “Mining a sub-matrix of maximal sum”. In: *Proceedings of the 6th International Workshop on New Frontiers in Mining Complex Patterns in conjunction with ECML-PKDD 2017*. 2017.

[CGS07] M C. Cooper, S. de Givry, and T. Schiex. "Optimal soft arc consistency". In: *Proc. of IJCAI'2007*. Hyderabad, India, Jan. 2007, pp. 68–73.

[Coo+08] Martin C Cooper et al. "Virtual Arc Consistency for Weighted CSP". In: *AAAI*. Vol. 8. 2008, pp. 253–258.

[Coo+10] M. Cooper et al. "Soft arc consistency revisited". In: *Artificial Intelligence* 174 (2010), pp. 449–478.

[Coo03] M C. Cooper. "Reduction operations in fuzzy or valued constraint satisfaction". In: *Fuzzy Sets and Systems* 134.3 (2003), pp. 311–342.

[Coo07] M C. Cooper. "On the minimization of locally-defined submodular functions". In: *Constraints* (2007). To appear.

[CS04] M C. Cooper and T. Schiex. "Arc consistency for soft constraints". In: *Artificial Intelligence* 154.1-2 (2004), pp. 199–227.

[Der+19] Guillaume Derval et al. "The maximum weighted submatrix coverage problem: A CP approach". In: *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*. Springer. 2019, pp. 258–274.

[Des+92] Johan Desmet et al. "The dead-end elimination theorem and its use in protein side-chain positioning". In: *Nature* 356.6369 (1992), pp. 539–542.

[DPO13] Simon De Givry, Steven D Prestwich, and Barry O'Sullivan. “Dead-end elimination for weighted CSP”. In: *Principles and Practice of Constraint Programming*. Springer. 2013, pp. 263–272.

[Fav+11] A. Favier et al. “Pairwise decomposition for combinatorial optimization in graphical models”. In: *Proc. of IJCAI'11*. Barcelona, Spain, 2011.

[Fre91] Eugene C. Freuder. “Eliminating Interchangeable Values in Constraint Satisfaction Problems”. In: *Proc. of AAAI'91*. Anaheim, CA, 1991, pp. 227–233.

[GSV06] S. de Givry, T. Schiex, and G. Verfaillie. “Exploiting Tree Decomposition and Soft Local Consistency in Weighted CSP”. In: *Proc. of the National Conference on Artificial Intelligence, AAAI-2006*. 2006, pp. 22–27.

[HD19] Mark A Hallen and Bruce R Donald. “Protein design by provable algorithms”. In: *Communications of the ACM* 62.10 (2019), pp. 76–84.

[HSS18] Stefan Haller, Paul Swoboda, and Bogdan Savchynskyy. “Exact MAP-Inference by Confining Combinatorial Search with LP Relaxation”. In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[Hur+16] Barry Hurley et al. “Multi-language evaluation of exact solvers in graphical model discrete optimization”. In: *Constraints* (2016), pp. 1–22.

[Kol06] Vladimir Kolmogorov. “Convergent tree-reweighted message passing for energy minimization”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 28.10 (2006), pp. 1568–1583.

[Kos99] A M C A. Koster. “Frequency assignment: Models and Algorithms”. Available at www.zib.de/koster/thesis.html. PhD thesis. The Netherlands: University of Maastricht, Nov. 1999.

[KZ17] Andrei A. Krokhin and Stanislav Zivny. “The Complexity of Valued CSPs”. In: *The Constraint Satisfaction Problem: Complexity and Approximability*. Ed. by Andrei A. Krokhin and Stanislav Zivny. Vol. 7. Dagstuhl Follow-Ups. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017, pp. 233–266. ISBN: 978-3-95977-003-3. DOI: 10.4230/DFU.Vol7.15301.9. URL: https://doi.org/10.4230/DFU.Vol7.15301.9.

[Lar+05] J. Larrosa et al. “Existential arc consistency: getting closer to full arc consistency in weighted CSPs”. In: *Proc. of the 19th IJCAI*. Edinburgh, Scotland, Aug. 2005, pp. 84–89.

[Lar00] J. Larrosa. “Boosting search with variable elimination”. In: *Principles and Practice of Constraint Programming - CP 2000*. Vol. 1894. LNCS. Singapore, Sept. 2000, pp. 291–305.

[Lar02] J. Larrosa. “On Arc and Node Consistency in weighted CSP”. In: *Proc. AAAI’02*. Edmondton, (CA), 2002, pp. 48–53.

[Lec+09] C. Lecoutre et al. “Reasoning from last conflict(s) in constraint programming”. In: *Artificial Intelligence* 173 (2009), pp. 1592, 1614.

[LL12] Jimmy Ho-Man Lee and Ka Lun Leung. “Consistency techniques for flow-based projection-safe global cost functions in weighted constraint satisfaction”. In: *Journal of Artificial Intelligence Research* 43.1 (2012), pp. 257–292.

[LS03] J. Larrosa and T. Schiex. “In the quest of the best form of local consistency for Weighted CSP”. In: *Proc. of the 18^{th} IJCAI*. Acapulco, Mexico, Aug. 2003, pp. 239–244.

[LS04] Javier Larrosa and Thomas Schiex. “Solving weighted CSP by maintaining arc consistency”. In: *Artif. Intell.* 159.1-2 (2004), pp. 1–26.

[LW66] Eugene L Lawler and David E Wood. “Branch-and-bound methods: A survey”. In: *Operations research* 14.4 (1966), pp. 699–719.

[MD09] Radu Marinescu and Rina Dechter. “AND/OR branch-and-bound search for combinatorial optimization in graphical models”. In: *Artificial Intelligence* 173.16-17 (2009), pp. 1457–1491.

[Mul+19] Vikram Khipple Mulligan et al. "Designing Peptides on a Quantum Computer". In: *bioRxiv* (2019), p. 752485.

[Mul+20] Maxime Mulamba et al. "Hybrid Classification and Reasoning for Image-based Constraint Solving". In: *Proc. of CPAIOR'20, also in arXiv preprint arXiv:2003.11001*. 2020, pp. 364–380.

[Oua+17] Abdelkader Ouali et al. "Iterative decomposition guided variable neighborhood search for graphical model energy minimization". In: *Conference on Uncertainty in Artificial Intelligence, UAI'17*. Sydney, Australia, 2017.

[Oua+20] Abdelkader Ouali et al. "Variable neighborhood search for graphical model energy minimization". In: *Artificial Intelligence* 278 (2020), p. 103194.

[Par+17] Youngsuk Park et al. "Learning the network structure of heterogeneous data via pairwise exponential Markov random fields". In: *Proceedings of machine learning research* 54 (2017), p. 1302.

[PPW18] Rasmus Palm, Ulrich Paquet, and Ole Winther. "Recurrent relational networks". In: *Advances in Neural Information Processing Systems*. 2018, pp. 3368–3378.

[Ruf+19] Manon Ruffini et al. "Guaranteed Diversity & Quality for the Weighted CSP". In: *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE. 2019, pp. 18–25.

[Sch00] T. Schiex. “Arc consistency for soft constraints”. In: *Principles and Practice of Constraint Programming - CP 2000*. Vol. 1894. LNCS. Singapore, Sept. 2000, pp. 411–424.

[Sch76] M.I. Schlesinger. “Sintaksicheskiy analiz dvumernykh zritelnikh signalov v usloviyakh pomekh (Syntactic analysis of two-dimensional visual signals in noisy conditions)”. In: *Kibernetika* 4 (1976), pp. 113–130.

[Sim+15] David Simoncini et al. “Guaranteed Discrete Energy Optimization on Large Protein Design Problems”. In: *Journal of Chemical Theory and Computation* 11.12 (2015), pp. 5980–5989. doi: 10.1021/acs.jctc.5b00594.

[TGK20] Fulya Trösser, Simon de Givry, and George Katsirelos. “VAC integrality based variable heuristics and initial upper-bounding (vacint and rasps): Relaxation-Aware Heuristics for Exact Optimization in Graphical Models”. In: *Proc. of CPAIOR-20*. 2020.

[Wan+19] Po-Wei Wang et al. “SATNet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver”. In: *ICML'19 proceedings, arXiv preprint arXiv:1905.12149*. 2019.

[Wer07] T. Werner. “A Linear Programming Approach to Max-sum Problem: A Review.”. In: *IEEE Trans. on Pattern Recognition and Machine Intelligence* 29.7 (July 2007), pp. 1165–1179. URL: http://dx.doi.org/10.1109/TPAMI.2007.1036.