Numbers and logic together in CP

Cost Function Networks learning and solving the WCSP

ACP'2020 GdR IA, RO and ANITI school

S. de Givry & T. Schiex (G. Katsirelos, N. Rousse, C. Brouard)

Université Fédérale de Toulouse, ANITI, INRAE MIAT, UR 875, Toulouse, France

November 2020

ANITI
ARTIFICIAL & NATURAL INTELLIGENCE TOULOUSE INSTITUTE

INRAE
science for people, life & earth

Format, target

- A mix of basic and advanced theory and algorithms
- Trying to show connections between fields
- With practicals on a VM with toulbar2 (and more)
- Make you curious and dig into it
- Whether you like theory, algorithms or code!

Format, target

1. Introducing Cost Function Networks (CFN) and coding a Visual Sudoku solver
2. Algorithms and theory: how does it work (not exhaustive)
3. Modeling and solving with toulbar2
4. Learning CFN from data (back to the Visual Sudoku)

What you should have done

- Install VirtualBox: https://www.virtualbox.org/wiki/Downloads
- Download our (4GB) Virtual Machine disk: http://shorturl.at/hL157
- Uncompress it to `ACPAIOR.vdi` (7z, more than 12GB)
- Launch VirtualBox and create a new "Debian 64 bits" Machine (set # CPU/RAM/video RAM)
- Use the above "`.vdi`" disk for storage

If you didn't

It's Ok, we will also show how it's done.

A Constraint Network

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

Joint truth function

$$\Phi_{\mathcal{M}} = \bigwedge_{\varphi_S \in \Phi} \varphi_S$$

The Constraint Satisfaction Problem (NP-complete)

- Is it possible to satisfy all constraints simultaneously?
- Is it possible to make $\Phi_{\mathcal{M}} = t$?
- What is the minimum of $\Phi_{\mathcal{M}}$? ($t < f$)

A Constraint Network

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

Joint truth function

$$\Phi_{\mathcal{M}} = \bigwedge_{\varphi_S \in \Phi} \varphi_S$$

The Constraint Satisfaction Problem (NP-complete)

- Is it possible to satisfy all constraints simultaneously?
- Is it possible to make $\Phi_{\mathcal{M}} = t$?
- What is the minimum of $\Phi_{\mathcal{M}}$? $(t < f)$

A Constraint Network

- 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 satisfy all constraints simultaneously?
- Is it possible to make $\Phi_{\mathcal{M}} = t$?
- What is the minimum of $\Phi_{\mathcal{M}}$? ($t < f$)

A Constraint Network

- 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 satisfy all constraints simultaneously?
- Is it possible to make $\Phi_{\mathcal{M}} = t$?
- What is the minimum of $\Phi_{\mathcal{M}}$? $(t < f)$

A Constraint Network

- 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 satisfy all constraints simultaneously?
- Is it possible to make $\Phi_{\mathcal{M}} = t$?
- What is the minimum of $\Phi_{\mathcal{M}}$? $(t < f)$

A Constraint Network

- 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 satisfy all constraints simultaneously?
- Is it possible to make $\Phi_{\mathcal{M}} = t$?
- What is the minimum of $\Phi_{\mathcal{M}}$? $(t < f)$

Languages for domains and constraints

- Conjunctive Propositional Logic: Boolean domains, constraints as clauses
- Good Old Constraint Networks: Boolean tables (tensors) for domains and constraints
- Constraint Programming: interval variables, specialized constraints, control

Clauses

- A disjunction of litterals
- Each litteral is a variable or its negation
- Forbids one assignment
- Sensitive to negation

Example

- $(\neg X \vee Y \vee Z)$
- $(X = t, Y = f, Z = f)$

Languages for domains and constraints

- Conjunctive Propositional Logic: Boolean domains, constraints as clauses
- Good Old Constraint Networks: Boolean tables (tensors) for domains and constraints
- Constraint Programming: interval variables, specialized constraints, control

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

- A multidimensional table with a Boolean for every $\boldsymbol{v} \in D^{\boldsymbol{S}}$
- Says if $\boldsymbol{v}$ is authorized (t) or not (f)
- Insensitive to negation

Pairwise equality (3 values)

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

Languages for domains and constraints

- Conjunctive Propositional Logic: Boolean domains, constraints as clauses
- Good Old 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

ALLDIFFERENTs

Languages for domains and constraints

- Conjunctive Propositional Logic: Boolean domains, constraints as clauses
- Good Old Constraint Networks: Boolean tables (tensors) for domains and constraints
- Constraint Programming: interval variables, specialized constraints, control

Application domains: NP and beyond

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

Excel at the analysis of complex perfectly known systems

Cost Function Network CFN(k)

- 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 (NP-hard)

- What is the minimum of $\Phi_{\mathcal{M}}$?
- decision NP-complete: can $\Phi_{\mathcal{M}}$ be less than a given threshold?

Cost Function Network CFN(k)

- 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 (NP-hard)

- What is the minimum of $\Phi_{\mathcal{M}}$?
- decision NP-complete: can $\Phi_{\mathcal{M}}$ be less than a given threshold?

Cost Function Network CFN(k)

- 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 (NP-hard)

- What is the minimum of $\Phi_{\mathcal{M}}$?
- decision NP-complete: can $\Phi_{\mathcal{M}}$ be less than a given threshold?

Cost Function Network CFN(k)

- 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 (NP-hard)

- What is the minimum of $\Phi_{\mathcal{M}}$?
- decision NP-complete: can $\Phi_{\mathcal{M}}$ be less than a given threshold?

Cost Function Network CFN(k)

- 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 (NP-hard)

- What is the minimum of $\Phi_{\mathcal{M}}$?
- decision NP-complete: can $\Phi_{\mathcal{M}}$ be less than a given threshold?

Tables (or tensors) for φ_S

- A multidimensional table with a number for every $\boldsymbol{v} \in D^S$

Global functions

- Names for specific (useful) functions

Soft equality (3 values)

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

A useful one

$\textsc{WeightedRegular}_S(\mathcal{A})$

Tables (or tensors) for φ_S

- A multidimensional table with a number for every $v \in D^S$

Global functions

- Names for specific (useful) functions

Soft equality (3 values)

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

A useful one

$\textsc{WeightedRegular}_S(\mathcal{A})$

Costs and constraints

- We assume non negative integer costs (algorithms)
- A constraint is a cost function that maps to $\{0, k\}$
- A pure Constraint Network is just a CFN(1)

Optimum preserving operations

- scaling: fixed decimal point numbers ok ($2^{63} \approx 19$ digits)
- shifting: negative numbers ok
- so minimization and maximization ok

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

- CFNs have a constant function $\varphi_\varnothing$
- CFNs have all unary functions $\varphi_i, X_i \in \boldsymbol{V}$ — $\varphi_i(u) = k$ means u deleted
- All functions have different scopes

Crucial property

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

Quite general so many formats

- wcsp, (w)cnf, qpbo, opb, .uai, .LG
- CFN: a JSON-like format (more tolerant)
- An evolving Python API (binds to the C++ library)

We will only use

The CFN format and the Python API

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

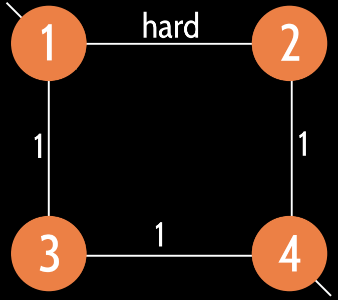
- vertices $\{1, 2, 3, 4\}$
- cut weight 1 or 1.5 $(1, 3)$
- edge $(1, 2)$ hard

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

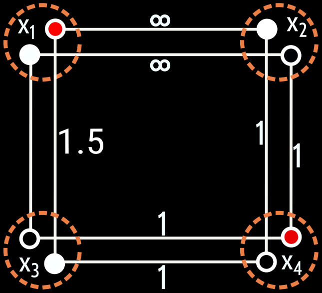


Graph $G = (\boldsymbol{V}, \boldsymbol{E})$ with edge weight function w

- 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 CFN
myCFN = CFN.CFN(100,1)   # ub, resolution (optional)
print("Starting Upper bound:",myCFN.GetUB())
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, _)
```

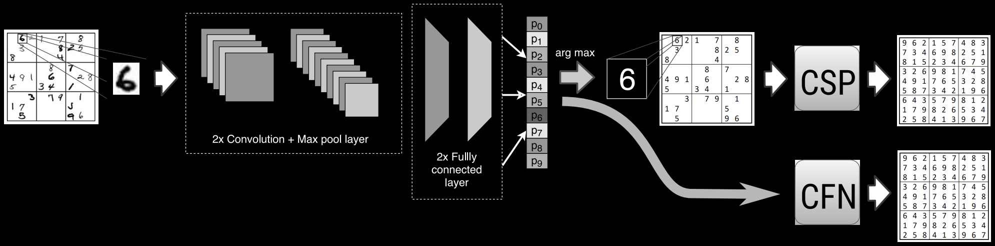
Definition

- Variables X_{ij} for cell (i, j) has domain $\{1, \cdots, 9\}$
- Set R_i contains all variables of row i, similarly for C_j & columns
- 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 try to write this as a toulbar2 Python API program: open a terminal in the VM and `cd sudoku`.

Numbers: interfacing with DL



Two models

Thanks to Tias Gun for the picture above

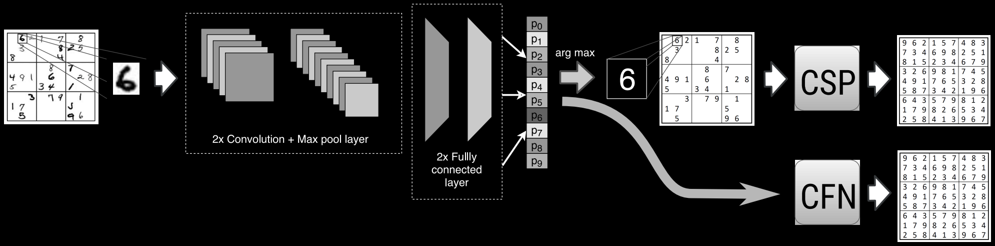
1. Booleans: Assign the cell variable with the prediction
2. Numbers: Add LeNet output tensor (negated) as a cost function
3. $(\min \sum -\log) \equiv (\max \prod)$ probabilities
4. Calibration: none here (simplest, listen to Tias tomorrow)

Example

Let's try to see the code for this (Boolean case).

We `cd ../sudoku-DL-CP`, then `cd ../sudoku-DL-CFN`

Numbers: interfacing with DL



Two models

Thanks to Tias Gun for the picture above

1. Booleans: Assign the cell variable with the prediction
2. Numbers: Add LeNet output tensor (negated) as a cost function
3. $(\min \sum -\log) \equiv (\max \prod)$ probabilities
4. Calibration: none here (simplest, listen to Tias tomorrow)

Example

Let's try to see the code for this (Boolean case).
We `cd ../sudoku-DL-CP`, then `cd ../sudoku-DL-CFN`

Compared to [Mul+20] (CPAIOR'20, using COP)

- The CFN variant corresponds to the "Hybrid1" approach of [Mul+20]
- On SAT-Net problems, with global All-Different, COP takes 0.79"
- CFN/toulbar2 with pairwise differences: 0.05" (one core)
- On 1000 problems, 996 are solved bactrack-free
- CFN bounds clearly tighter than COP bounds [LL12]

Extensive comparison of CFN/toulbar2 [Hur+16]

Winner of successive "Approximate Probabilistic Inference" challenges

Compared to [Mul+20] (CPAIOR’20, using COP)

- The CFN variant corresponds to the “Hybrid1” approach of [Mul+20]
- On SAT-Net problems, with global All-Different, COP takes 0.79"
- CFN/toulbar2 with pairwise differences: 0.05" (one core)
- On 1000 problems, 996 are solved bactrack-free
- CFN bounds clearly tighter than COP bounds [LL12]

Extensive comparison of CFN/toulbar2 [Hur+16]

Winner of successive “Approximate Probabilistic Inference” challenges

ANITI INRAE

As a discrete Markov Random Field (ML/Computer Vision)

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e non negative real-valued cost functions
- usually described as tables/tensors

$\mathcal{M}$: induces a probability distribution

$$\Phi_{\mathcal{M}} = \prod_{\varphi_S \in \Phi} \varphi_S \qquad P_{\mathcal{M}} \propto \Phi_{\mathcal{M}}$$

Maximum a Posteriori (MAP)

Maximizing $P_{\mathcal{M}}$ or $\Phi_{\mathcal{M}}$ is the same.

ANITI INRAE

As a discrete Markov Random Field (ML/Computer Vision)

- a sequence of discrete domain variables $\boldsymbol{V}$
- a set Φ of e non negative real-valued cost functions
- usually described as tables/tensors

$\mathcal{M}$: induces a probability distribution

$$\Phi_{\mathcal{M}} = \prod_{\varphi_{\boldsymbol{S}} \in \Phi} \varphi_{\boldsymbol{S}} \qquad\qquad P_{\mathcal{M}} \propto \Phi_{\mathcal{M}}$$

Maximum a Posteriori (MAP)

Maximizing $P_{\mathcal{M}}$ or $\Phi_{\mathcal{M}}$ is the same.

MRFs tightly connected to CFNs ($k = \infty$) (additive energy)

$$\text{MRF } \mathcal{M} \xrightarrow[-\log(x)]{} \text{CFN } \mathcal{M}^{\ell} \xrightarrow[\exp(-x)]{} \text{MRF } \mathcal{M}$$

In the end

WCSP on CFN(∞) $\Leftrightarrow$ MAP on MRF

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

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

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

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

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

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

ANITI INRAE

Only nd variables

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

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

With Boolean variables

Pure Quadratic Pseudo-Boolean Optimization[BH02] (posiform)

Definition (Function equivalence)

Two functions (or CFNs) are equivalent iff they are always equal

Definition (Relaxation of a function)

A function (or CFN) φ is a relaxation of φ' iff $\varphi \leq \varphi'$

For CFN(1) SAT/CSP

$(\varphi \text{ relaxation of } \varphi') \Leftrightarrow (\varphi' \models \varphi)$

Definition (Function equivalence)

Two functions (or CFNs) are equivalent iff they are always equal

Definition (Relaxation of a function)

A function (or CFN) φ is a relaxation of φ' iff $\varphi \leq \varphi'$

For CFN(1) SAT/CSP

$(\varphi \text{ relaxation of } \varphi') \Leftrightarrow (\varphi' \models \varphi)$

Definition (Function equivalence)

Two functions (or CFNs) are equivalent iff they are always equal

Definition (Relaxation of a function)

A function (or CFN) φ is a relaxation of φ' iff $\varphi \leq \varphi'$

For CFN(1) SAT/CSP

$(\varphi \text{ relaxation of } \varphi') \Leftrightarrow (\varphi' \models \varphi)$

1. Algorithms
   - Conditioning based: systematic and local search
   - Elimination based: local consistency and variable elimination
2. All Toulbar2 bells and whistles
3. Learning CFN from data

ANITI INRAE

Conditioning: $\varphi_{\boldsymbol{S}|X=a} \quad (X \in \boldsymbol{S})$	Assignment
$\varphi_{\boldsymbol{S}|X=a}(\boldsymbol{v}) = (\varphi_{\boldsymbol{S}}(\boldsymbol{v} \cup \{X = a\})$	Scope $\boldsymbol{S} - \{X\}$, negligible complexity

$$\begin{array}{cc|c|c|c} & & & X_1 & \\ & a & 1 & 2 & 3 \\ \hline X_2 & b & 3 & 1 & 2 \\ \hline & c & 2 & 3 & 1 \end{array}$$

Conditioning by $X_2 = b$

$$\begin{array}{c|c|c} & X_1 & \\ 3 & 1 & 2 \end{array}$$

Conditioning: $\varphi_{\boldsymbol{S}\mid X=a} \quad (X \in \boldsymbol{S})$	Assignment
$\varphi_{\boldsymbol{S}\mid X=a}(\boldsymbol{v}) = (\varphi_{\boldsymbol{S}}(\boldsymbol{v} \cup \{X=a\})$	Scope $\boldsymbol{S} - \{X\}$, negligible complexity

$$
\begin{array}{cc|c|c|c}
 & & & X_1 & \\
 & a & 1 & 2 & 3 \\
\hline
X_2 & b & 3 & 1 & 2 \\
\hline
 & c & 2 & 3 & 1
\end{array}
$$

Conditioning by $X_2 = b$

$$
\begin{array}{c|c|c}
 & X_1 & \\
3 & 1 & 2
\end{array}
$$

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 condition by $X_i = u$
  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!

Partial search

Relaxed pruning ($(1+\alpha)\varphi_\varnothing \geq k$) [Poh70], bounded number of backtracks or discrepencies (LDS [HG95])

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 condition by $X_i = u$
  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!

Partial search

Relaxed pruning ($(1+\alpha)\varphi_\varnothing \geq k$) [Poh70], bounded number of backtracks or discrepencies (LDS [HG95])

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 condition by $X_i = u$
  2. one where u is removed from D^X
- Return the minimum

Optimization — Branch and Bound [LW66]

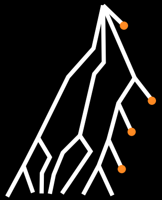
If the $\underbrace{\text{local lower bound}}_{\varphi_\varnothing}$ reaches the $\underbrace{\text{global upper bound}}_{k}$ — Prune!

Partial search

Relaxed pruning ($(1+\alpha)\varphi_\varnothing \geq k$) [Poh70], bounded number of backtracks or discrepencies (LDS [HG95])

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])



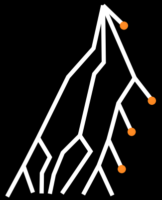
Nice properties

- 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])

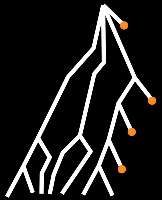


Nice properties

- 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])



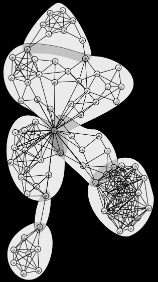
Nice properties

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

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

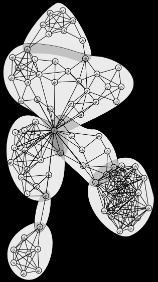
Hybrid Best First Search [All+15] — Anyspace

- 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



Hybrid Best First Search [All+15] — Anyspace

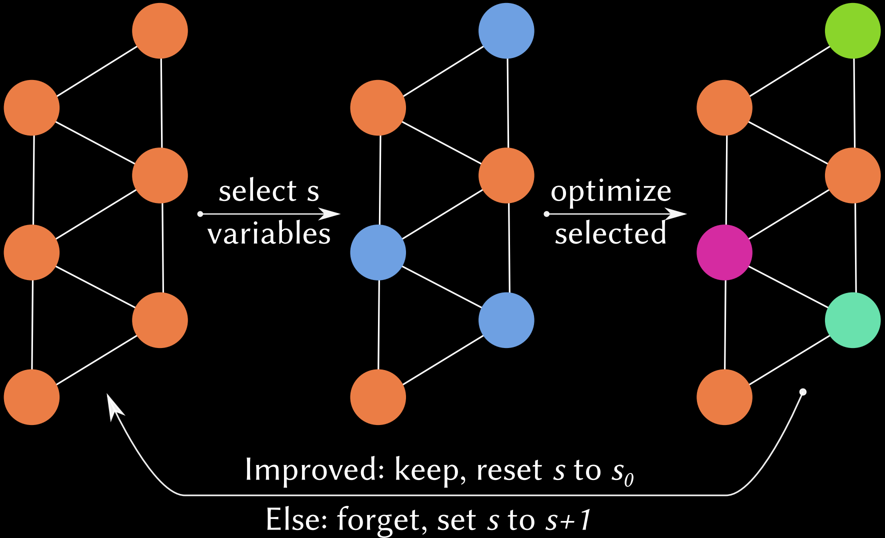
- Uses Depth-First Search for a bounded amount of backtracks
- Pending nodes are pushed onto a list of Open nodes
- The next DFS starts from the best Open node
- Tree-decomposition friendly (BTD [GSV06]/AND-OR search [MD09])

Nice properties

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

Also local search of course (VNS here)

ANITI INRAE



Combination of $\varphi_{\boldsymbol{S}}$ and $\varphi_{\boldsymbol{S}'}$ — Space/time $O(d^{|\boldsymbol{S}\cup\boldsymbol{S}'|})$ for tensors

$$(\varphi_{\boldsymbol{S}} \overset{k}{+} \varphi_{\boldsymbol{S}'})(\boldsymbol{v}) = \varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}]) \overset{k}{+} \varphi_{\boldsymbol{S}'}(\boldsymbol{v}[\boldsymbol{S}'])$$

X_2		X_1		
a	4	1	2	3
b	6	3	1	2
c	4	2	3	1

$\Longrightarrow$

X_2	X_1		
a	5	6	7
b	9	7	8
c	6	7	5

Elimination of $X \in \boldsymbol{S}$ from $\varphi_{\boldsymbol{S}}$ — Time $O(d^{|\boldsymbol{S}|})$, space $O(d^{|\boldsymbol{S}|-1})$ for tensors

$$\varphi_{\boldsymbol{S}}[-X](\boldsymbol{u}) = \min_{\boldsymbol{v}\in D^X} \varphi_{\boldsymbol{S}}(\boldsymbol{u}\cup\boldsymbol{v})$$

Produces relaxations

X_2	X_1		
a	5	6	7
b	9	7	8
c	6	7	5

Eliminate X_2

X_1		
5	6	5

Eliminate X_1

$\varnothing$

5

Combination of φ_S and $\varphi_{S'}$ — Space/time $O(d^{|S \cup S'|})$ for tensors

$$(\varphi_S \overset{k}{+} \varphi_{S'})(v) = \varphi_S(v[S]) \overset{k}{+} \varphi_{S'}(v[S'])$$

		X_2		$X_1 = 1$	$X_1 = 2$	$X_1 = 3$
X_2	a	4		1	2	3
	b	6		3	1	2
	c	4		2	3	1

$\Longrightarrow$

		$X_1 = 1$	$X_1 = 2$	$X_1 = 3$
X_2	a	5	6	7
	b	9	7	8
	c	6	7	5

Elimination of $X \in S$ from φ_S — Time $O(d^{|S|})$, space $O(d^{|S|-1})$ for tensors

$$\varphi_S[-X](u) = \min_{v \in D^X} \varphi_S(u \cup v)$$

Produces relaxations

		$X_1 = 1$	$X_1 = 2$	$X_1 = 3$
X_2	a	5	6	7
	b	9	7	8
	c	6	7	5

Eliminate X_2

X_1		
5	6	5

Eliminate X_1

$\varnothing$: 5

ANITI INRAE

Combination of $\varphi_{\boldsymbol{S}}$ and $\varphi_{\boldsymbol{S'}}$ — Space/time $O(d^{|\boldsymbol{S}\cup\boldsymbol{S'}|})$ for tensors

$$(\varphi_{\boldsymbol{S}} \overset{k}{+} \varphi_{\boldsymbol{S'}})(\boldsymbol{v}) = \varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}]) \overset{k}{+} \varphi_{\boldsymbol{S'}}(\boldsymbol{v}[\boldsymbol{S'}])$$

X_2 / X_1:

X_2	φ
a	4
b	6
c	4

X_2 \ X_1	1	2	3
a	1	2	3
b	3	1	2
c	2	3	1

$\Longrightarrow$

X_2 \ X_1			
a	5	6	7
b	9	7	8
c	6	7	5

Elimination of $X \in \boldsymbol{S}$ from $\varphi_{\boldsymbol{S}}$ — Time $O(d^{|\boldsymbol{S}|})$, space $O(d^{|\boldsymbol{S}|-1})$ for tensors

$$\varphi_{\boldsymbol{S}}[-X](\boldsymbol{u}) = \min_{\boldsymbol{v}\in D^X} \varphi_{\boldsymbol{S}}(\boldsymbol{u}\cup\boldsymbol{v})$$

Produces relaxations

X_2 \ X_1			
a	5	6	7
b	9	7	8
c	6	7	5

Eliminate X_2

X_1		
5	6	5

Eliminate X_1

$\varnothing$

5

ANITI INRAE

Combination of $\varphi_{\boldsymbol{S}}$ and $\varphi_{\boldsymbol{S'}}$ — Space/time $O(d^{|\boldsymbol{S}\cup\boldsymbol{S'}|})$ for tensors

$$(\varphi_{\boldsymbol{S}} \overset{k}{+} \varphi_{\boldsymbol{S'}})(\boldsymbol{v}) = \varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}]) \overset{k}{+} \varphi_{\boldsymbol{S'}}(\boldsymbol{v}[\boldsymbol{S'}])$$

X_2	
a	4
b	6
c	4

X_2 \ X_1			
a	1	2	3
b	3	1	2
c	2	3	1

$\Longrightarrow$

X_2 \ X_1			
a	5	6	7
b	9	7	8
c	6	7	5

Elimination of $X \in \boldsymbol{S}$ from $\varphi_{\boldsymbol{S}}$ — Time $O(d^{|\boldsymbol{S}|})$, space $O(d^{|\boldsymbol{S}|-1})$ for tensors

$$\varphi_{\boldsymbol{S}}[-X](\boldsymbol{u}) = \min_{\boldsymbol{v}\in D^X} \varphi_{\boldsymbol{S}}(\boldsymbol{u}\cup\boldsymbol{v})$$

Produces relaxations

X_2 \ X_1			
a	5	6	7
b	9	7	8
c	6	7	5

Eliminate X_2

X_1		
5	6	5

Eliminate X_1

$\varnothing$
5

Used together

- Combination accumulates all information in a single function
- Elimination forgets one variable without loosing *optimality* information

At the core of

- Local consistencies, Unit propagation: subproblem induced by one function
- Variable elimination, the Resolution Principle: subproblem around one variable

Used together

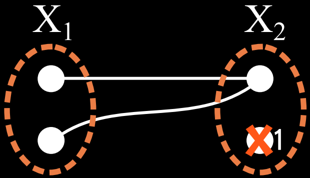
- Combination accumulates all information in a single function
- Elimination forgets one variable without loosing *optimality* information

At the core of

- Local consistencies, Unit propagation: subproblem induced by one function
- Variable elimination, the Resolution Principle: subproblem around one variable

FIltering by Arc Consistency (simplicity)

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



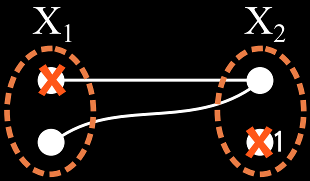
Arc consistency

Makes the domain X_i (φ_i) consistent with φ_{ij} and φ_j

ANITI INRAE

FIltering by Arc Consistency (simplicity)

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



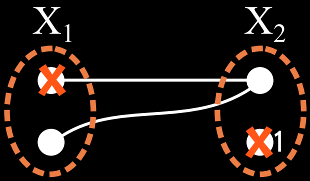
Arc consistency

Makes the domain X_i (φ_i) consistent with φ_{ij} and φ_j

ANITI INRAE

FIltering by Arc Consistency (simplicity)

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



Arc consistency

Makes the domain X_i (φ_i) consistent with φ_{ij} and φ_j

ANITI INRAE

Arc consistency of X_i w.r.t. φ_{ij} [RBW06]

- Combine φ_{ij} and the unary φ_j
- Eliminate X_j producing a function (message) on X_i

$$m_i^j = (\varphi_{ij} \overset{k}{+} \varphi_j)[-X_j]$$

$$X_2 \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \overset{k}{+} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

Eliminate X_2 $\begin{bmatrix} 1 & 1 \end{bmatrix}$

Properties

- The message can be added to φ_i (relaxation, value deletion)
- X_i is AC w.r.t. φ_{ij} if $m_j^i \leq \varphi_i$ (no new information)
- Unique fixpoint, reached in polynomial time (inconsistency detection)
- Support of $u \in D^i$ on D^j the argmin of the elimination

Arc consistency of X_i w.r.t. φ_{ij} [RBW06]

- Combine φ_{ij} and the unary φ_j
- Eliminate X_j producing a function (message) on X_i

$$m_i^j = (\varphi_{ij} \stackrel{k}{+} \varphi_j)[-X_j]$$

$$X_2 \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \stackrel{k}{+} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

Eliminate X_2 $\begin{bmatrix} 1 & 1 \end{bmatrix}$

Properties

- The message can be added to φ_i (relaxation, value deletion)
- X_i is AC w.r.t. φ_{ij} if $m_j^i \leq \varphi_i$ (no new information)
- Unique fixpoint, reached in polynomial time (inconsistency detection)
- Support of $u \in D^i$ on D^j the argmin of the elimination

Obvious issue

Messages can not be included in the CFN: loss of equivalence, meaningless result

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

- Add the message m_i^j to φ_j with $+^k$
- Subtract m_i^j from its source using $-^k$

Can be reversed, any relaxation of m_i^j can be used instead

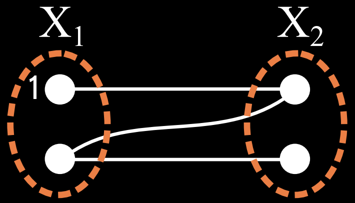
Obvious issue

Messages can not be included in the CFN: loss of equivalence, meaningless result

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

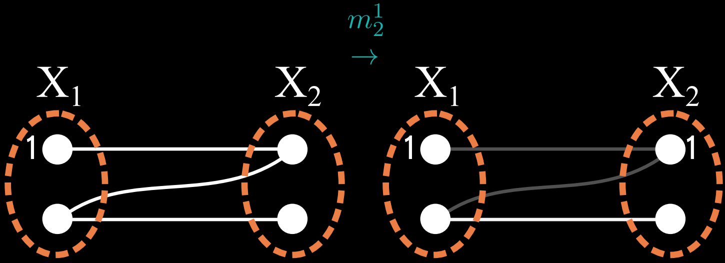
- Add the message m_i^j to φ_j with $+^k$
- Subtract m_i^j from its source using $-^k$

Can be reversed, any relaxation of m_i^j can be used instead



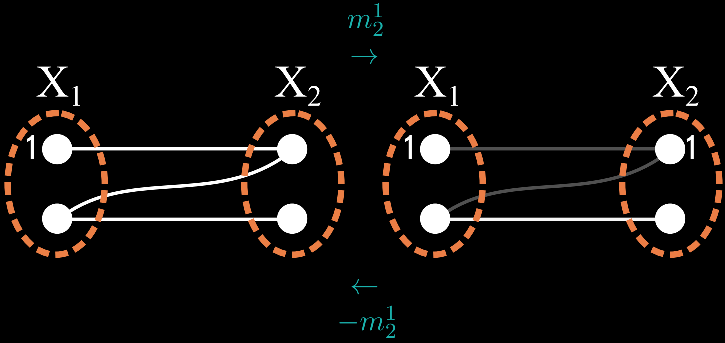
(Loss of) properties

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



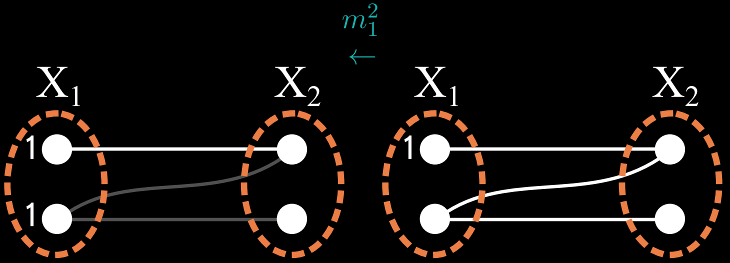
(Loss of) properties

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



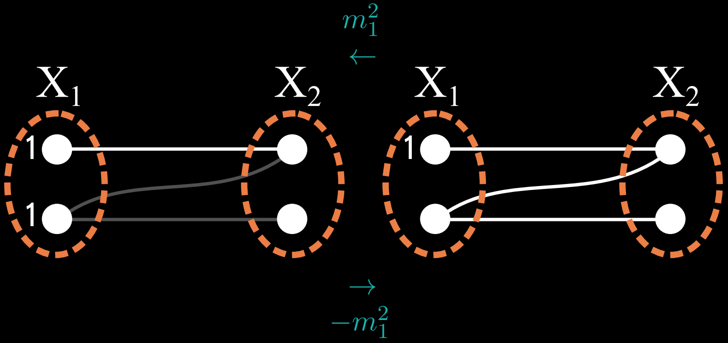
(Loss of) properties

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



(Loss of) properties

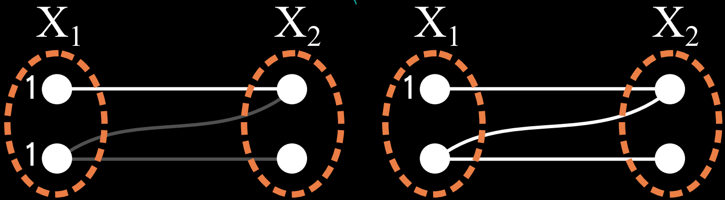
Preserves equivalence but non-monotonic and fixpoints may be non unique (or may not exist)



(Loss of) properties

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

$\xleftarrow{m_1^2}$

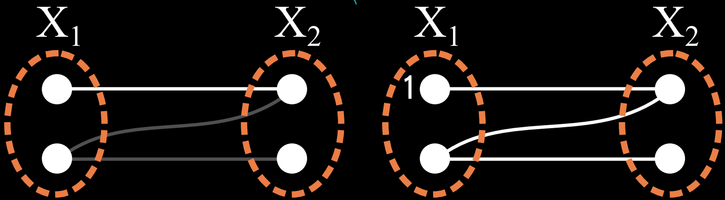


$\Downarrow \quad m_\varnothing^1$

(Loss of) properties

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

$$\overset{m_1^2}{\leftarrow}$$

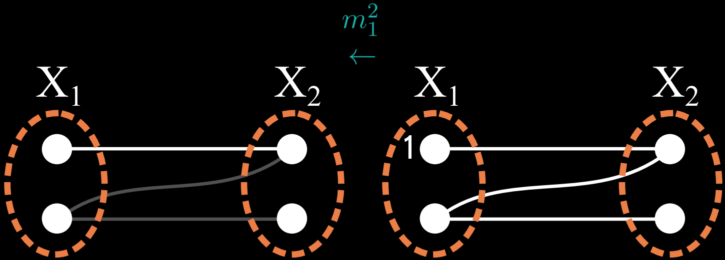


$$\Downarrow \quad m_\varnothing^1$$

$$\varphi_\varnothing = 1$$

(Loss of) properties

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



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

$\varphi_{\varnothing} = 1$

(Loss of) properties

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

ANITI INRAE

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

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

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

The many "soft ACs" [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 supports u for X_i: $\varphi_i(u) = 0$, can be extended for free on X_i's star
- VAC supports 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

The many "soft ACs" [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 supports u for X_i: $\varphi_i(u) = 0$, can be extended for free on X_i's star
- VAC supports 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 (CFN(1))
- Polynomial time and $O(ed)$ space (Generalized ACs)
- Incremental, strengthens $\varphi_\varnothing$ (NC $\leq$ AC $\leq$ FDAC $\leq$ EDAC$\leq$ VAC)
- Stronger bounds than AC in COP [LL12]

Sequence of integer EPTs

Computing a sequence of integer EPTs that maximizes $\varphi_\varnothing$ is decision NP-complete [CS04]

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

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

Properties

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

Sequence of integer EPTs

Computing a sequence of integer EPTs that maximizes $\varphi_\varnothing$ is decision NP-complete [CS04]

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

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

Properties

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

Sequence of integer EPTs

Computing a sequence of integer EPTs that maximizes $\varphi_\varnothing$ is decision NP-complete [CS04]

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

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

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

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

The "local polytope"

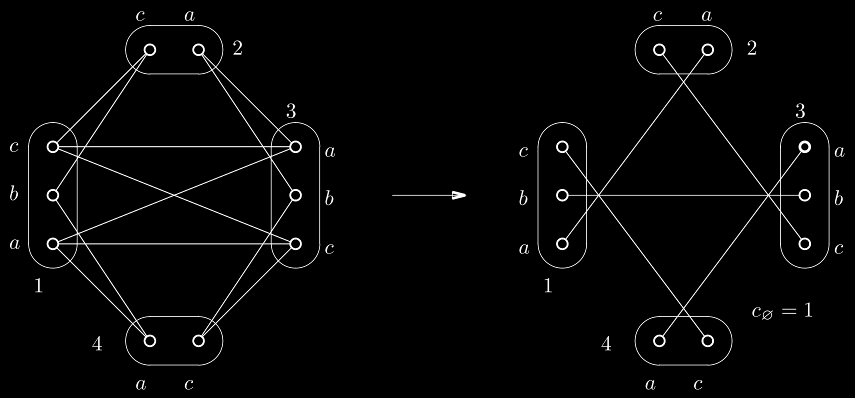
Minimize $\sum_{i,a} \varphi_i(a) \cdot x_{ia} + \sum_{\substack{\varphi_{ij} \in \Phi \\ a \in D^i, b \in D^j}} \varphi_{ij}(a,b) \cdot y_{iajb}$ such that

$$\sum_{a \in D^i} x_{ia} = 1 \qquad \forall i \in \{1, \ldots, n\} \tag{2}$$

$$\sum_{b \in D^j} y_{iajb} = x_{ia} \qquad \forall \varphi_{ij} \in \Phi, \forall a \in D^i \tag{3}$$

$$\sum_{a \in D^i} y_{iajb} = x_{jb} \qquad \forall \varphi_{ij} \in \Phi, \forall b \in D^j \tag{4}$$

u_i multiplier for (2), p_{ija}/p_{jib} for (3) and (4)



c
a
2
3
c
b
a
a
b
c
1
4
a
c
c
a
2
3
c
b
a
a
b
c
1
$c_\varnothing = 1$
4
a
c

The local polytope proved to be "Universal for LP" [PW15]

This means that

- Any (well-behaved) LP can be transformed in linear time in a CFN such that the OSAC bound is the optimum of the LP
- On this CFN, VAC will provide an approximation of the bound (faster)
- On VAC/LP, see also [DW20b; DW20a]

The local polytope proved to be "Universal for LP" [PW15]

This means that

- Any (well-behaved) LP can be transformed in linear time in a CFN such that the OSAC bound is the optimum of the LP
- On this CFN, VAC will provide an approximation of the bound (faster)
- On VAC/LP, see also [DW20b; DW20a]

Problem solved by OSAC/VAC [Coo+10; KZ17]

- Tree-structured problems
- Permutated submodular problems (eg. Min-Cut, Min/Max-closed relations)
- OSAC/VAC + $\forall X_i, \exists! u \in D^i$ s.t. $\varphi_i(u) = 0$ [Coo+10; HSS18; TGK20]

OSAC empirically too expensive

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

CFN Local Consistencies

Enhance CP with fast incremental approximate Linear Programming dual bounds

Problem solved by OSAC/VAC [Coo+10; KZ17]

- Tree-structured problems
- Permutated submodular problems (eg. Min-Cut, Min/Max-closed relations)
- OSAC/VAC + $\forall X_i, \exists! u \in D^i$ s.t. $\varphi_i(u) = 0$ [Coo+10; HSS18; TGK20]

OSAC empirically too expensive

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

CFN Local Consistencies

Enhance CP with fast incremental approximate Linear Programming dual bounds

Problem solved by OSAC/VAC [Coo+10; KZ17]

- Tree-structured problems
- Permutated submodular problems (eg. Min-Cut, Min/Max-closed relations)
- OSAC/VAC + $\forall X_i, \exists! u \in D^i$ s.t. $\varphi_i(u) = 0$ [Coo+10; HSS18; TGK20]

OSAC empirically too expensive

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

CFN Local Consistencies

Enhance CP with fast incremental approximate Linear Programming dual bounds

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.
```

Kind words from OpenGM2 developpers

"ToulBar2 variants were superior to CPLEX variants in all our tests"[HSS18]

Kind words from a famous Protein Designer (Bruce Donald, [HD19])

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

"ToulBar2 variants were superior to CPLEX variants in all our tests"[HSS18]

Kind words from a famous Protein Designer (Bruce Donald, [HD19])

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

VAC: can we plan our cost shifts w/o LP?

CFN $(\boldsymbol{V}, \Phi)$

Bool(P) is the constraint network $(\boldsymbol{V}, \Phi - \{\varphi_\varnothing\})$ $(k = 1)$

Bool(P) forbids all positive cost assignments, ignoring $\varphi_\varnothing$

In P, a solution of Bool(P) is optimal and has cost $\varphi_\varnothing$

Virtual AC [Coo+08; Coo+10]

A CFN P is Virtual AC iff Bool(P) has a non empty AC closure

VAC: can we plan our cost shifts w/o LP?

CFN $(\boldsymbol{V}, \Phi)$

Bool(P) is the constraint network $(\boldsymbol{V}, \Phi - \{\varphi_\varnothing\})$ $(k = 1)$

Bool(P) forbids all positive cost assignments, ignoring $\varphi_\varnothing$

In P, a solution of Bool(P) is optimal and has cost $\varphi_\varnothing$

Virtual AC[Coo+08; Coo+10]

A CFN P is Virtual AC iff Bool(P) has a non empty AC closure

CFN $(\boldsymbol{V}, \Phi)$

Bool(P) is the constraint network $(\boldsymbol{V}, \Phi - \{\varphi_\varnothing\})$ $(k = 1)$

Bool(P) forbids all positive cost assignments, ignoring $\varphi_\varnothing$

In P, a solution of Bool(P) is optimal and has cost $\varphi_\varnothing$

Virtual AC [Coo+08; Coo+10]

A CFN P is Virtual AC iff Bool(P) has a non empty AC closure

Loop[Coo+10]

1. Enforce AC in Bool(P) until wipe-out (record propagation DAG)
2. Extract a minimal DAG D that wipes-out
3. Apply suitable cost shifting on D

Generalizes…

1. Ford-Fulkerson: an augmenting path is an augmenting DAG
2. The “roof-dual” lower bound of QPBO [BH02]
3. Solves submodular problems + all AC-decided Bool(P)

Related to convergent MP in MRFs

Same fixpoints as TRW-S [Kol06], MPLP1[Son+12], SRMP [Kol15], Max-+ diffusion [KK75; Coo+10]…

Loop[Coo+10]

1. Enforce AC in Bool(P) until wipe-out (record propagation DAG)
2. Extract a minimal DAG D that wipes-out
3. Apply suitable cost shifting on D

Generalizes…

1. Ford-Fulkerson: an augmenting path is an augmenting DAG
2. The “roof-dual” lower bound of QPBO [BH02]
3. Solves submodular problems + all AC-decided Bool(P)

Related to convergent MP in MRFs

Same fixpoints as TRW-S [Kol06], MPLP1[Son+12], SRMP [Kol15], Max-+ diffusion [KK75; Coo+10]…

Loop [Coo+10]

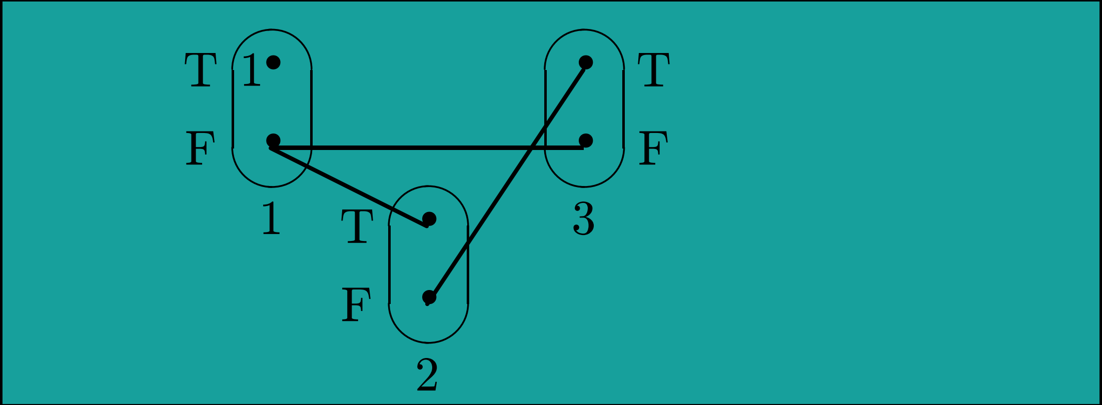
1. Enforce AC in Bool(P) until wipe-out (record propagation DAG)
2. Extract a minimal DAG D that wipes-out
3. Apply suitable cost shifting on D

Generalizes…

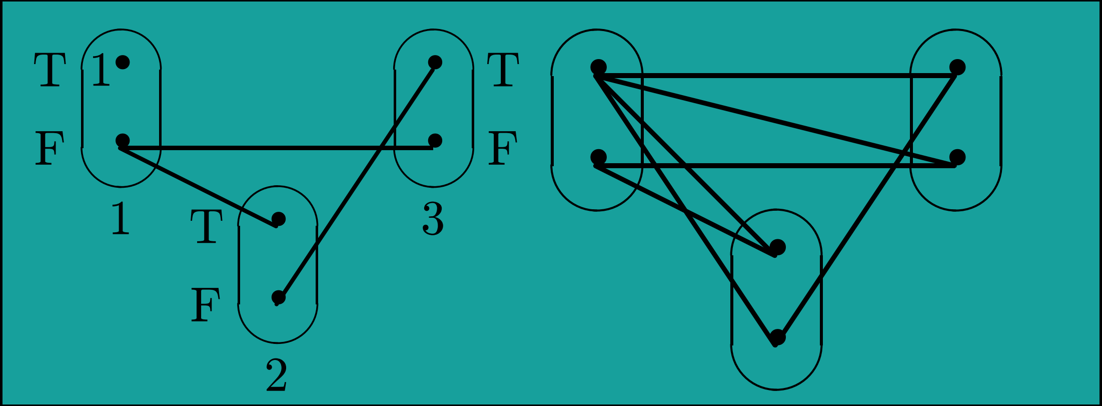
1. Ford-Fulkerson: an augmenting path is an augmenting DAG
2. The “roof-dual” lower bound of QPBO [BH02]
3. Solves submodular problems + all AC-decided Bool(P)

Related to convergent MP in MRFs

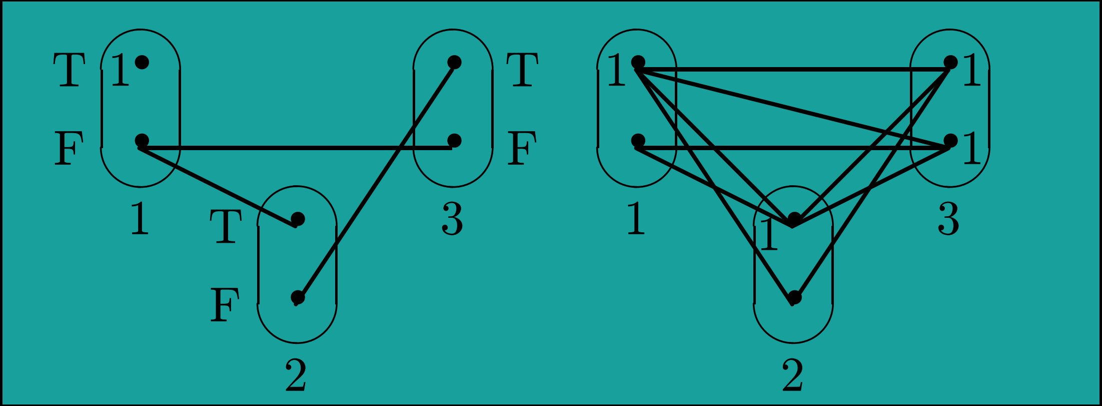
Same fixpoints as TRW-S [Kol06], MPLP1[Son+12], SRMP [Kol15], Max-+ diffusion [KK75; Coo+10]…



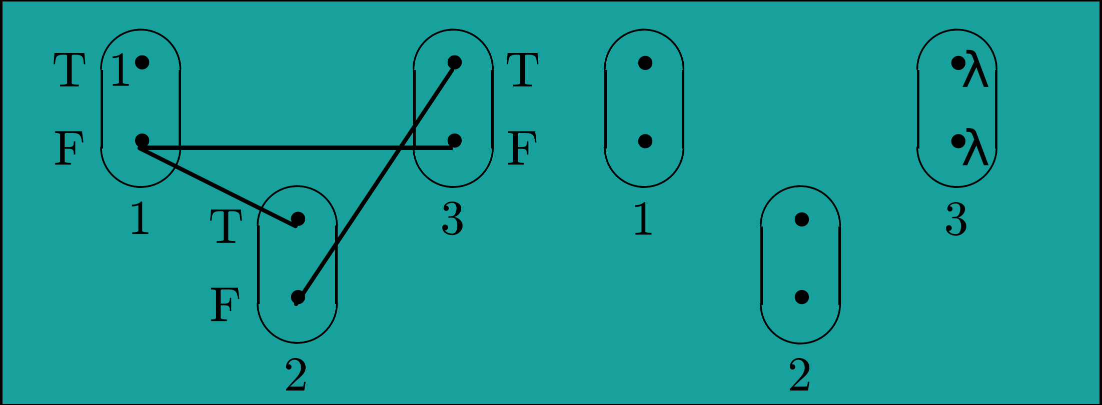
Original problem



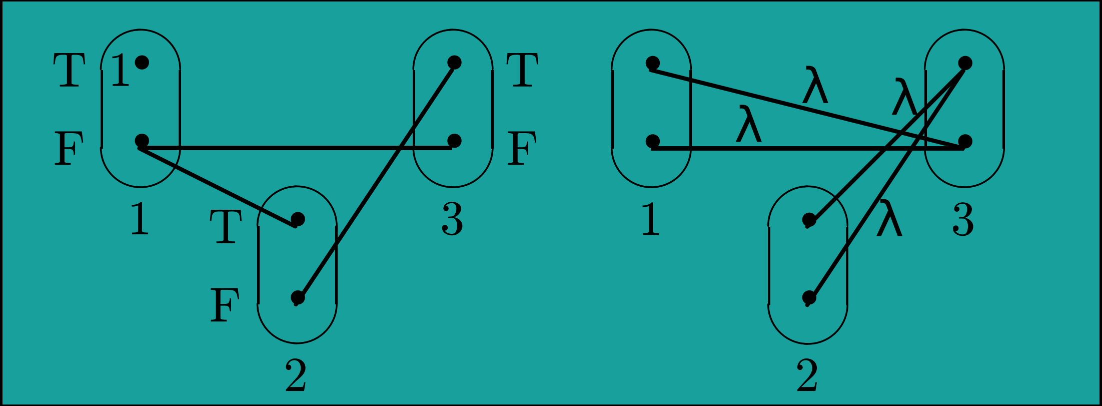
AC: deleting $(3, F)$ and $(2, T)$



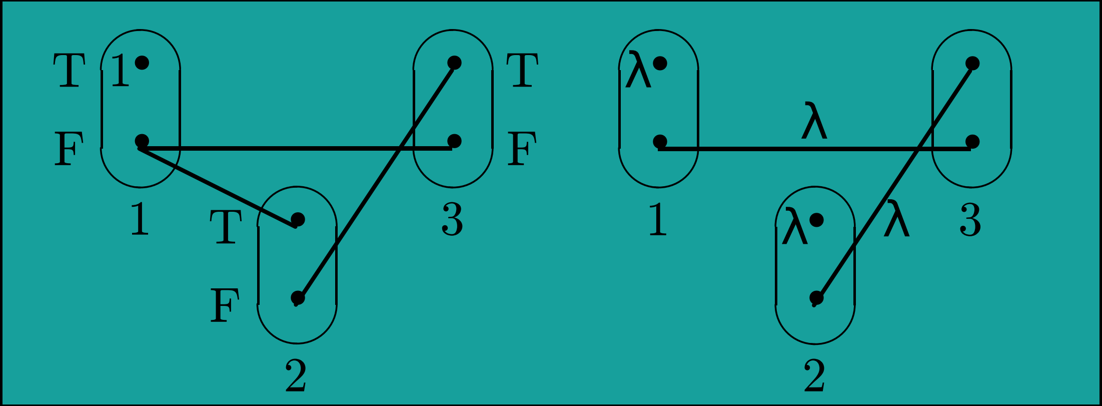
AC: deleting $(3, T)$: wipe out with 3 EPTs !



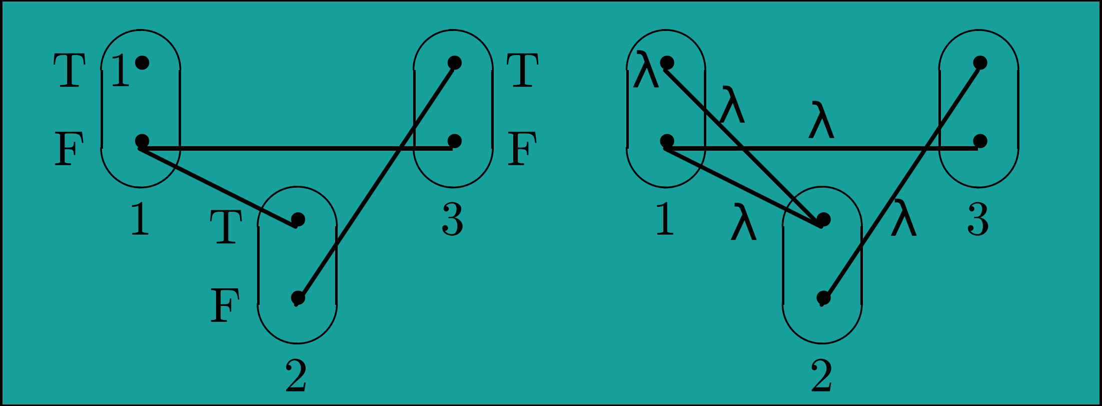
We want to bring λ cost unit to x_3, λ unknown.



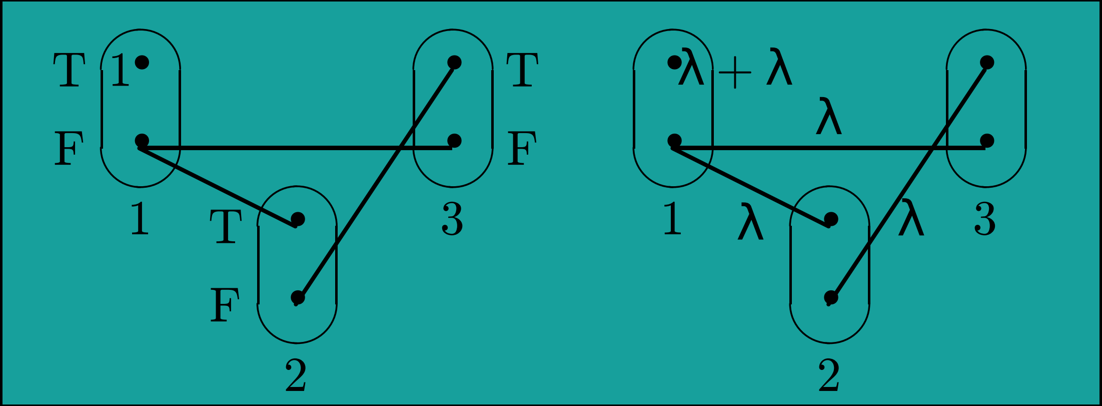
This requires λ virtual cost that needs to be paid by concrete costs...



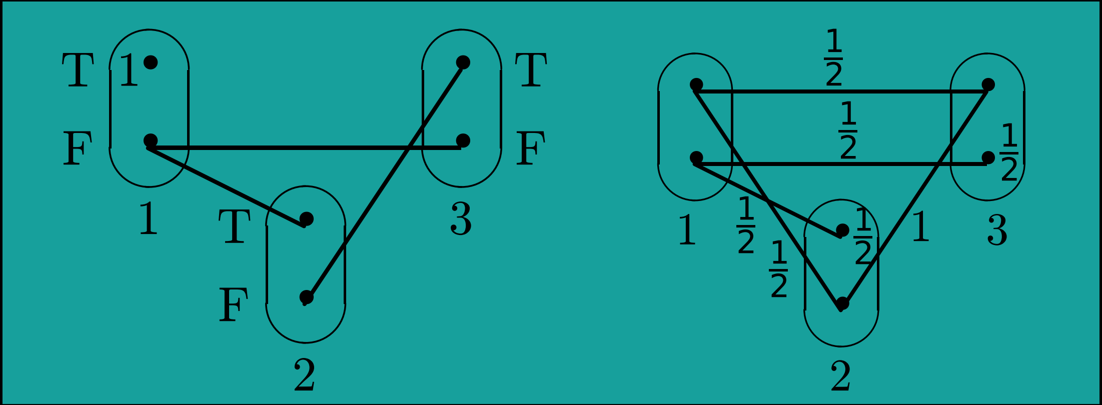
This requires λ virtual cost that needs to be paid by concrete costs… or propagated through EPTs



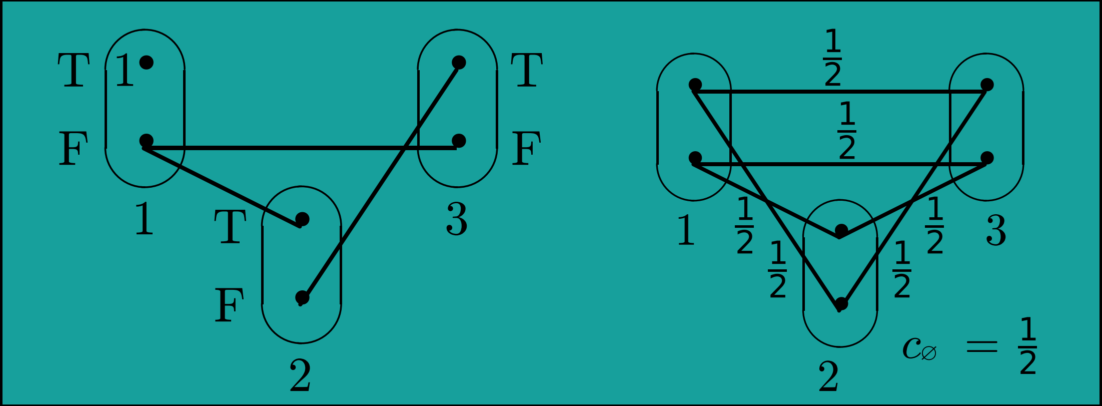
This requires λ virtual cost that needs to be paid by concrete costs… or propagated through EPTs back to concrete costs



we need 2λ on $(1, T)$ and have only 1 unit of cost: $\lambda = \frac{1}{2}$



We replay the EPTs using the values of λ



At the end we are able to project λ to $c_{\varnothing}$ (this means 1 (integrality))

Soft UP and Max resolution [LH05; BLM07] — **Some issues**

- combination and elimination are Ok
- but subtracting a clause from another clause does not yield a clause (CNF/DNF)
- generates additional “compensation” clauses [LH05; HLO07; BLM07; LHG08])

ANITI INRAE

Definition (Message from X to its neighbors)

Let $X \in \boldsymbol{V}$, and Φ^X be the set $\{\varphi_{\boldsymbol{S}} \in \Phi \text{ s.t. } X \in \boldsymbol{S}\}$, $\boldsymbol{T}$, the neighbors of X.

The message $m_{\boldsymbol{T}}^{\Phi^X}$ from Φ^X to $\boldsymbol{T}$ is:

$$m_{\boldsymbol{T}}^{\Phi^X} = \Big(\sum_{\varphi_{\boldsymbol{S}} \in \Phi^X}^{k} \varphi_{\boldsymbol{S}} \Big)[-X]$$

The message contains all the effect of X on the optimization problem Distributivity

$$\min_{\boldsymbol{v} \in D^{\boldsymbol{V}}} \left[\sum_{\varphi_{\boldsymbol{S}} \in \Phi}^{k} (\varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}])) \right] = \min_{\boldsymbol{v} \in D^{\boldsymbol{V}-\{X\}}} \left[\sum_{\varphi_{\boldsymbol{S}} \in \Phi - \Phi^X \cup \{m_{\boldsymbol{T}}^{\Phi^X}\}}^{k} (\varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}])) \right]$$

ANITI INRAE

Definition (Message from X to its neighbors)

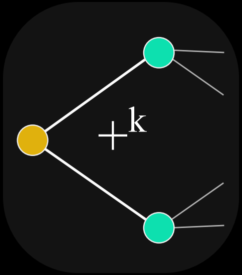
Let $X \in \boldsymbol{V}$, and Φ^X be the set $\{\varphi_{\boldsymbol{S}} \in \Phi \text{ s.t. } X \in \boldsymbol{S}\}$, $\boldsymbol{T}$, the neighbors of X.

The message $m^{\Phi_X}_{\boldsymbol{T}}$ from Φ^X to $\boldsymbol{T}$ is:

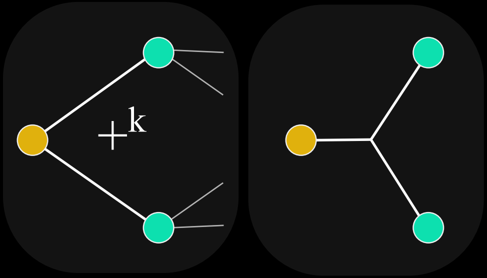
$$m^{\Phi_X}_{\boldsymbol{T}} = \Big(\sum_{\varphi_{\boldsymbol{S}} \in \Phi^X}^{k} \varphi_{\boldsymbol{S}} \Big)[-X]$$

The message contains all the effect of X on the optimization problem Distributivity

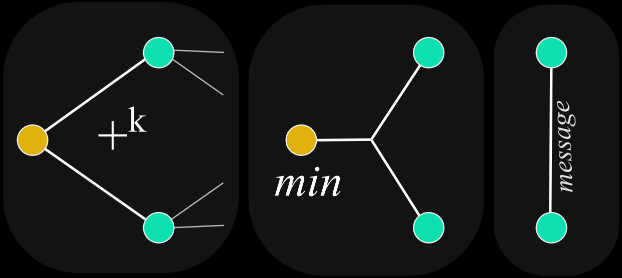
$$\min_{\boldsymbol{v} \in D^{\boldsymbol{V}}} \left[\sum_{\varphi_{\boldsymbol{S}} \in \Phi}^{k} (\varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}])) \right] = \min_{\boldsymbol{v} \in D^{\boldsymbol{V} - \{X\}}} \left[\sum_{\varphi_{<\boldsymbol{S}} \in \Phi - \Phi^X \cup \{m^{\Phi_X}_{\boldsymbol{T}}\}}^{k} (\varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}])) \right]$$



Daoopt & mini-buckets [DR03] split Φ^X in subsets of controlled size (lower bound)

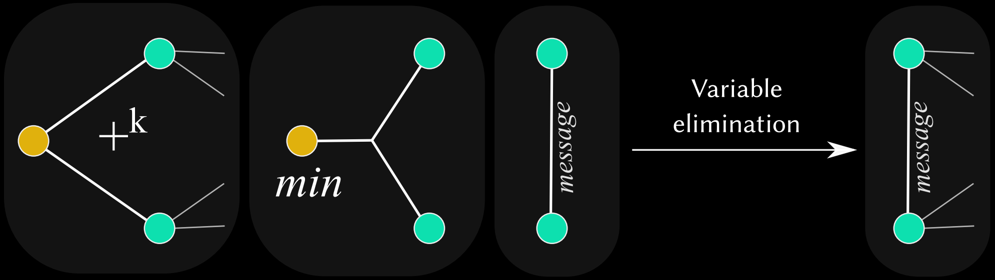


Daoopt & mini-buckets [DR03] split Φ^X in subsets of controlled size (lower bound)

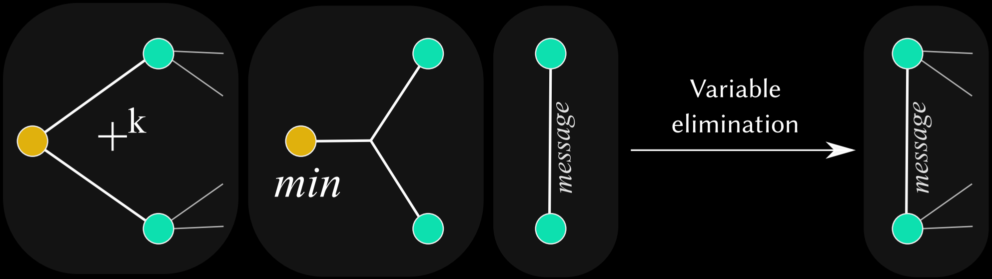


Daoopt & mini-buckets [DR03] split Φ^X in subsets of controlled size (lower bound)

Variable elimination



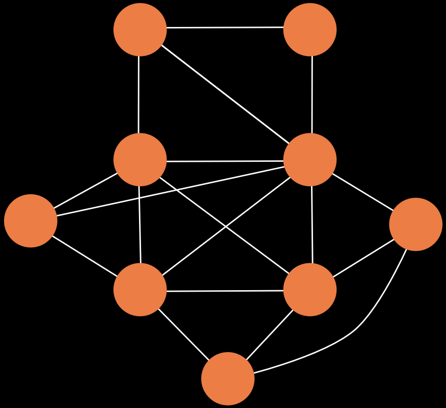
Daoopt & mini-buckets [DR03] split Φ^X in subsets of controlled size (lower bound)



Daoopt & mini-buckets [DR03] split Φ^X in subsets of controlled size (lower bound)

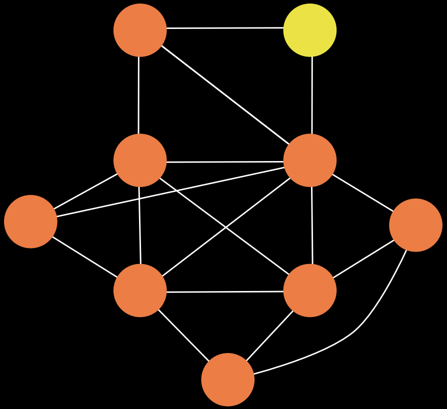
Boosting search with VE [Lar00]

If a variable has a small degree, eliminate it (backtrackable) else branch



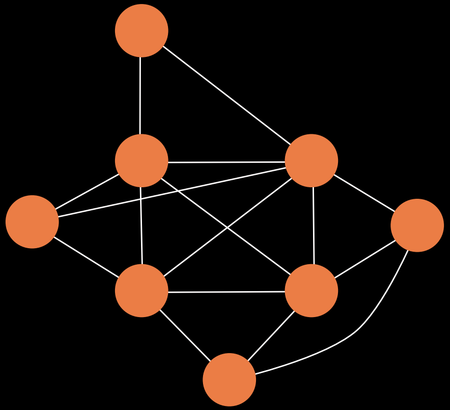
Boosting search with VE [Lar00]

If a variable has a small degree, eliminate it (backtrackable) else branch



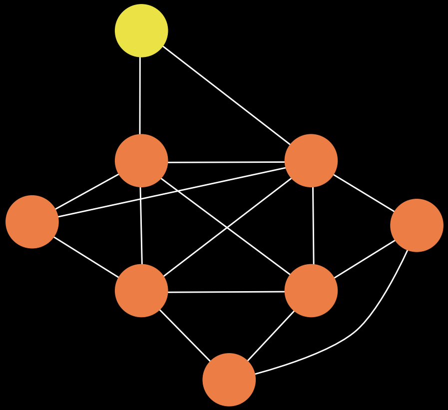
Boosting search with VE [Lar00]

If a variable has a small degree, eliminate it (backtrackable) else branch



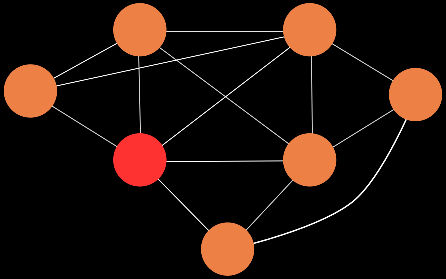
Boosting search with VE [Lar00]

If a variable has a small degree, eliminate it (backtrackable) else branch



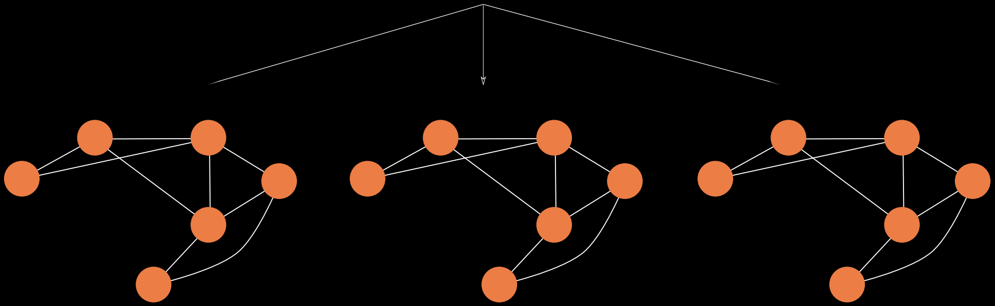
Boosting search with VE [Lar00]

If a variable has a small degree, eliminate it (backtrackable) else branch



Boosting search with VE [Lar00]

If a variable has a small degree, eliminate it (backtrackable) else branch



Presentation Outline

Additional algorithmic ingredients

- Value ordering (for free): existential or virtual supports
- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Dominance analysis (substitutability/DEE) [Fre91; Des+92; DPO13; All+14]
- Function decomposition [Fav+11]
- 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])

Additional algorithmic ingredients

- Value ordering (for free): existential or virtual supports
- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Dominance analysis (substitutability/DEE) [Fre91; Des+92; DPO13; All+14]
- Function decomposition [Fav+11]
- 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])

Additional algorithmic ingredients

- Value ordering (for free): existential or virtual supports
- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Dominance analysis (substitutability/DEE) [Fre91; Des+92; DPO13; All+14]
- Function decomposition [Fav+11]
- 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])

Additional algorithmic ingredients

- Value ordering (for free): existential or virtual supports
- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Dominance analysis (substitutability/DEE) [Fre91; Des+92; DPO13; All+14]
- Function decomposition [Fav+11]
- 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])

Additional algorithmic ingredients

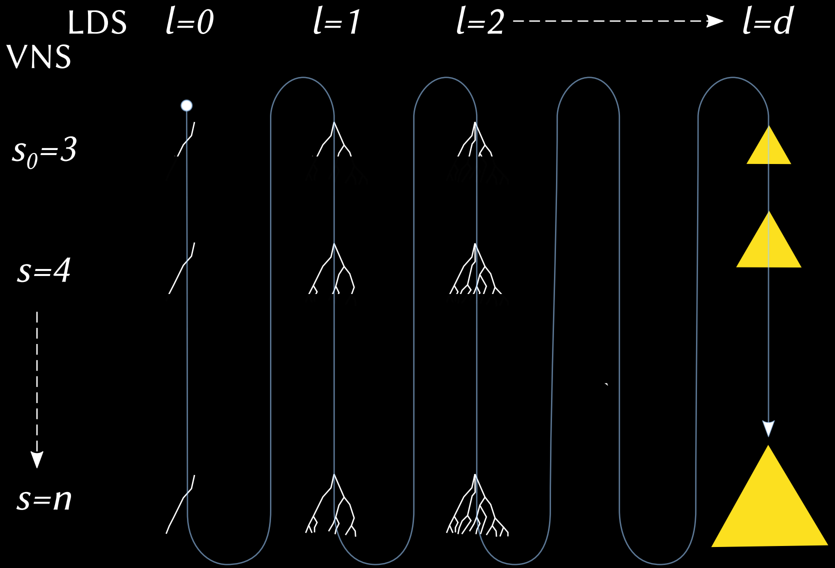
- Value ordering (for free): existential or virtual supports
- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Dominance analysis (substitutability/DEE) [Fre91; Des+92; DPO13; All+14]
- Function decomposition [Fav+11]
- Global cost functions (weighted Regular,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

Additional algorithmic ingredients

- Value ordering (for free): existential or virtual supports
- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Dominance analysis (substitutability/DEE) [Fre91; Des+92; DPO13; All+14]
- Function decomposition [Fav+11]
- 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])

Additional algorithmic ingredients

- Value ordering (for free): existential or virtual supports
- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Dominance analysis (substitutability/DEE) [Fre91; Des+92; DPO13; All+14]
- Function decomposition [Fav+11]
- 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])

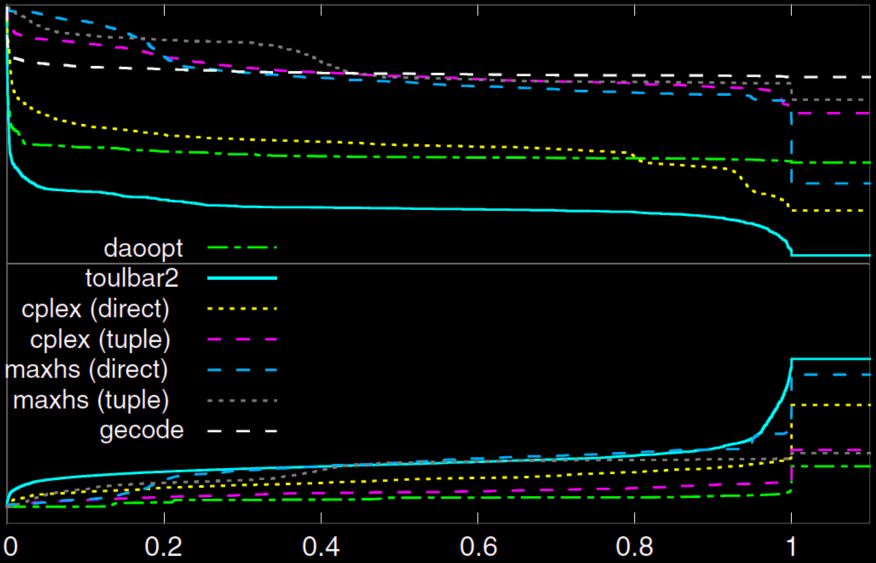

LDS
VNS
l=0
l=1
l=2
l=d
s0=3
s=4
s=n

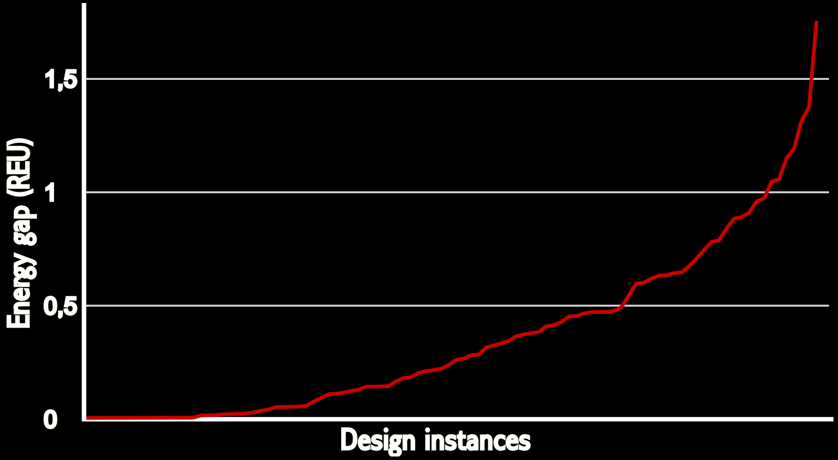
ANITI INRAE

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
- MaxCSP: MaxCSP 2008 competition
- WPMS: Weighted Partial MaxSAT evaluation 2013
- CP: MiniZinc challenge 2012/13

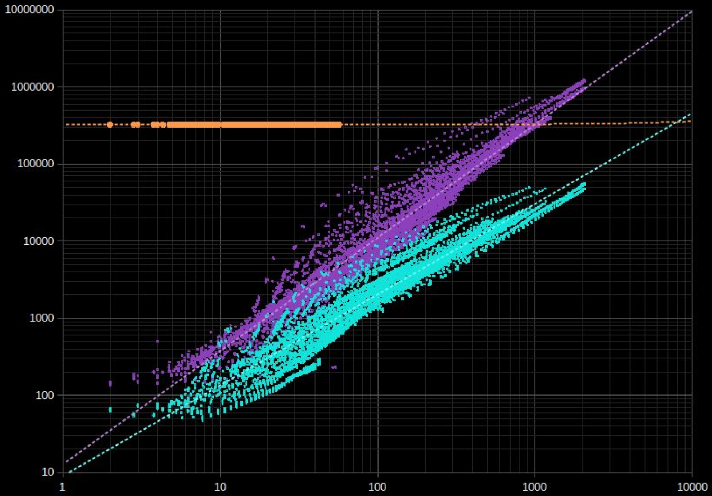
Benchmark	Nb.	UAI	WCSP	LP(direct)	LP(tuple)	WCNF(direct)	WCNF(tuple)	MINIZINC
MRF	319	187MB	475MB	2.4G	2.0GB	518MB	2.9GB	473MB
CVPR	1461	430MB	557MB	9.8GB	11GB	3.0GB	15GB	N/A
CFN	281	43MB	122MB	300MB	3.5GB	389MB	5.7GB	69MB
MaxCSP	503	13MB	24MB	311MB	660MB	73MB	999MB	29MB
WPMS	427	N/A	387MB	433MB	N/A	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


daoopt
toulbar2
cplex (direct)
cplex (tuple)
maxhs (direct)
maxhs (tuple)
gecode
0
0.2
0.4
0.6
0.8
1



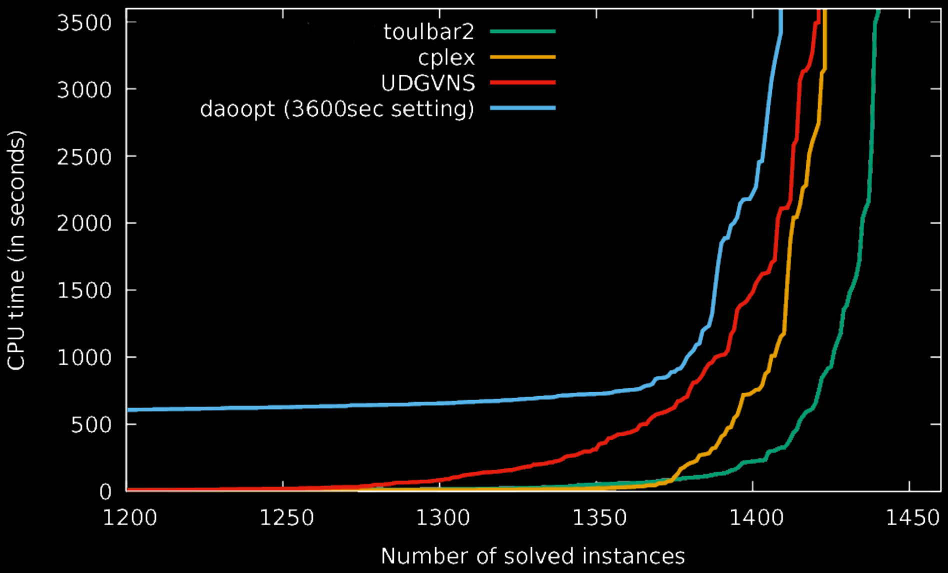
Optimality gap of the Simulated annealing solution as problems get harder

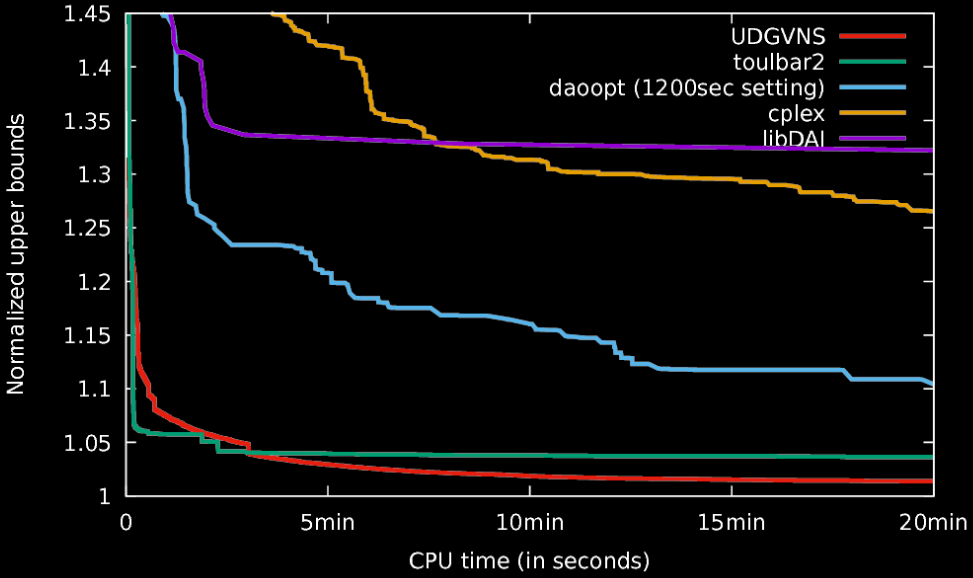
Quantum computing (DWave),Toulbar2 & SA [Mul+19]

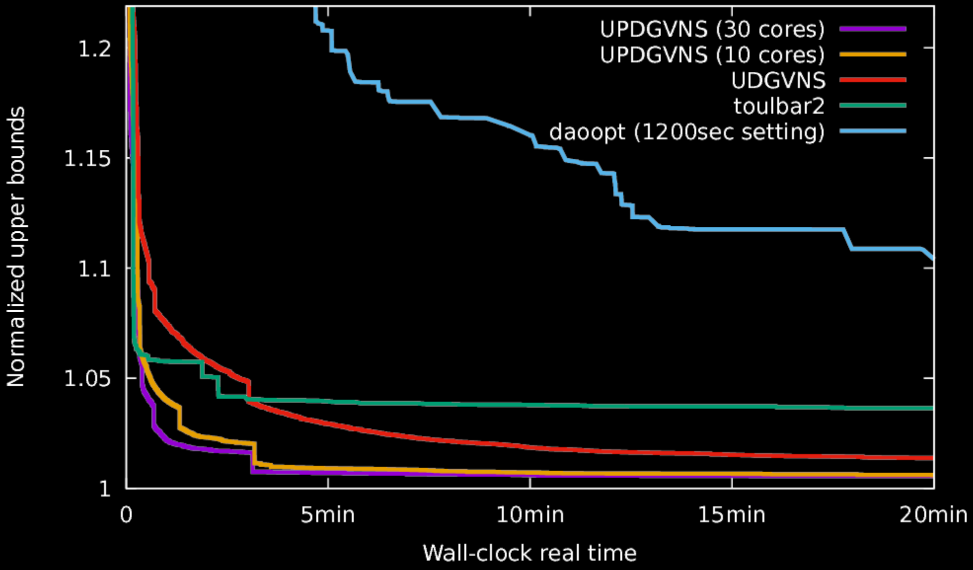


DWave approximations		kcal/mol
gap > 1.16 90% of the time	> 4.35, 50% of the time	> 8.45, 10% of the time

UDGVNS - Number of solved problems [Oua+17]




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 (30 cores)
UPDGVNS (10 cores)
UDGVNS
toulbar2
daoopt (1200sec setting)
Normalized upper bounds
1.2
1.15
1.1
1.05
1
0
5min
10min
15min
20min
Wall-clock real time

with Simon de Givry

Before going back to algorithms for learning CFN from data.

Definition (Learning a pairwise CFN from high quality solutions)

Given:

- a set of variables $\boldsymbol{V}$,
- a set of assignments $\boldsymbol{E}$ 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

MRFs tightly connected to CFNs ($k = \infty$) (additive energy)

$$\text{MRF } \mathcal{M} \xrightarrow[-\log(x)]{} \text{CFN } \mathcal{M}^{\ell} \xrightarrow[\exp(-x)]{} \text{MRF } \mathcal{M}$$

Definition (Learning a pairwise CFN from high quality solutions)

Given:

- a set of variables $\boldsymbol{V}$,
- a set of assignments $\boldsymbol{E}$ 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

MRFs tightly connected to CFNs ($k = \infty$) (additive energy)

$$\text{MRF } \mathcal{M} \xrightarrow[-\log(x)]{} \text{CFN } \mathcal{M}^{\ell} \xrightarrow[\exp(-x)]{} \text{MRF } \mathcal{M}$$

ANITI INRAE

The MRF connection opens the door to learning from data $\boldsymbol{E}$

- E a set of i.i.d. assignments of $\boldsymbol{V}$
- The log-likelihood of $\mathcal{M}$ given $\boldsymbol{E}$ is $\log(\prod_{\boldsymbol{v}\in\boldsymbol{E}} P_{\mathcal{M}}(\boldsymbol{v})) = \sum_{\boldsymbol{v}\in\boldsymbol{E}} \log(P_{\mathcal{M}}(\boldsymbol{v}))$
- Maximizing loglikelihood over all binary $\mathcal{M}$

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

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

ANITI INRAE

The MRF connection opens the door to learning from data $\boldsymbol{E}$

- E a set of i.i.d. assignments of $\boldsymbol{V}$
- The log-likelihood of $\mathcal{M}$ given $\boldsymbol{E}$ is $\log(\prod_{\boldsymbol{v}\in\boldsymbol{E}} P_{\mathcal{M}}(\boldsymbol{v})) = \sum_{\boldsymbol{v}\in\boldsymbol{E}} \log(P_{\mathcal{M}}(\boldsymbol{v}))$
- Maximizing loglikelihood over all binary $\mathcal{M}$

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

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

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.
- Ideally, plenty of them will be equal to zero and ignored (no function)

Regularized loglikelihood

- Penalizes loglikelihood by the norm of the costs learned (all the tables)
- Avoids over-fitting by pushing non-essential costs towards 0

Regularized loglikelihood estimation

$$\max \mathcal{L}(\mathcal{M}, \boldsymbol{E}) - \lambda ||\Phi||$$

Various approaches **Normalization #P-hard**

- Expensive Monte Carlo methods (MCMC)
- Pseudo-loglikelihood (no pseudo-counts)
- Approximate loglikelihood (expectations of counts as sufficient statistics)
- Convex optimization, differentiable or not (L1).

We use PE_MRF [Par+17]

- Approximate loglikelihood based (probabilistic input)
- ADMM based: easy to add norms, can use L1, L2, L1/L2,…
- Can learn hybrid continuous and discrete models (contextual models)
- λ needs to be adjusted (single dimension optimization)

Regularized Max Loglikelihood

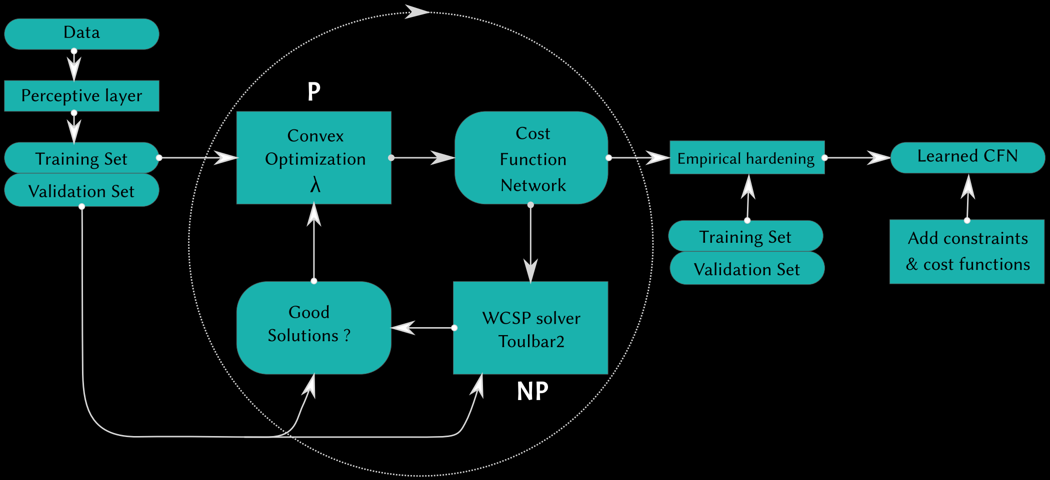
Various approaches — Normalization #P-hard

- Expensive Monte Carlo methods (MCMC)
- Pseudo-loglikelihood (no pseudo-counts)
- Approximate loglikelihood (expectations of counts as sufficient statistics)
- Convex optimization, differentiable or not (L1).

We use PE_MRF [Par+17]

- Approximate loglikelihood based (probabilistic input)
- ADMM based: easy to add norms, can use L1, L2, L1/L2,...
- Can learn hybrid continuous and discrete models (contextual models)
- λ needs to be adjusted (single dimension optimization)

The general picture



Using empirical risk minimization

- for each solution v in the validation set, assign a fraction of v (Sudoku)
- prefer values λ that give solutions close to the full v

Close to ?

- exact solutions: prefer a small Hamming distance between the solution found and the expected one.
- probabilistic ML output: prefer a large probability that the solution found is the expected one.

Or

Set λ using heuristic ML solutions (e.g. StARS [LRW10])

Influence of λ of learned CFNs (L1)

- Low λ: a lot of noisy functions, very hard to solve
- Best λ: less functions, also less noise (zero costs)
- High λ: less and less functions, ultimately an empty CFN.

Controlling PyToulbar2 optimization effort

- bounded optimization effort (backtracks, time, optimality gap)
- controllable faction of v assigned (more means exponentially easier)

Constraints: empirical hardening

Set positive costs that are never violated in the training/validation sets to ∞.

The cost of tuning λ

Influence of λ of learned CFNs (L1)

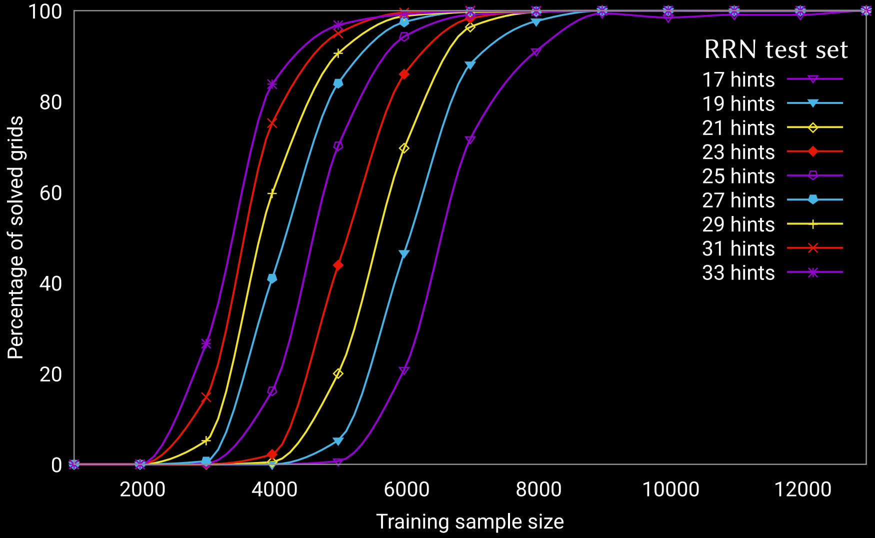
- Low λ: a lot of noisy functions, very hard to solve
- Best λ: less functions, also less noise (zero costs)
- High λ: less and less functions, ultimately an empty CFN.

Controlling PyToulbar2 optimization effort

- bounded optimization effort (backtracks, time, optimality gap)
- controllable faction of v assigned (more means exponentially easier)

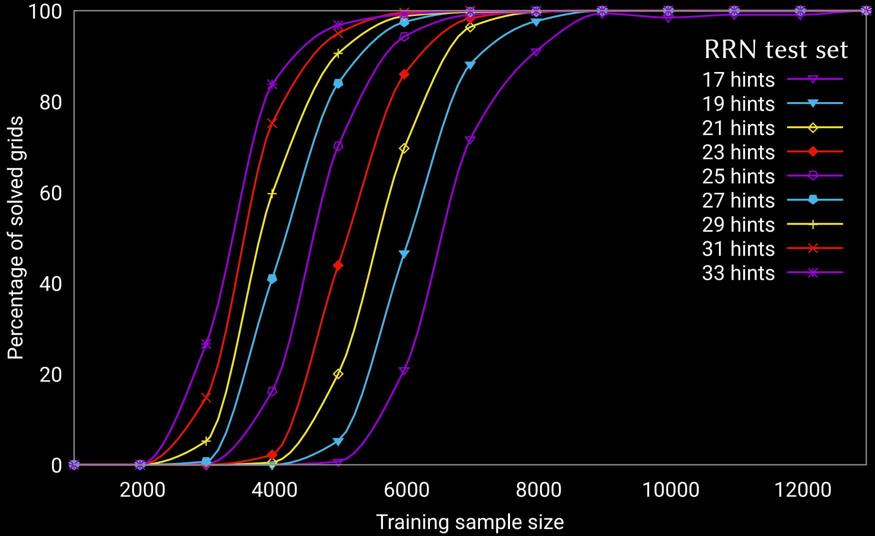
Constraints: empirical hardening

Set positive costs that are never violated in the training/validation sets to ∞.



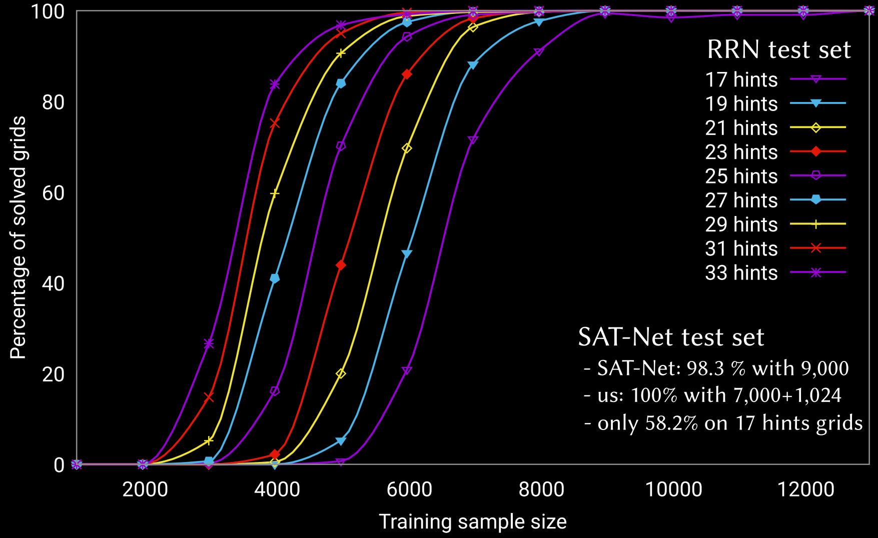
RRN: 96.6%
180,000 samples
(scopes given)

Better with less data and comparable biases



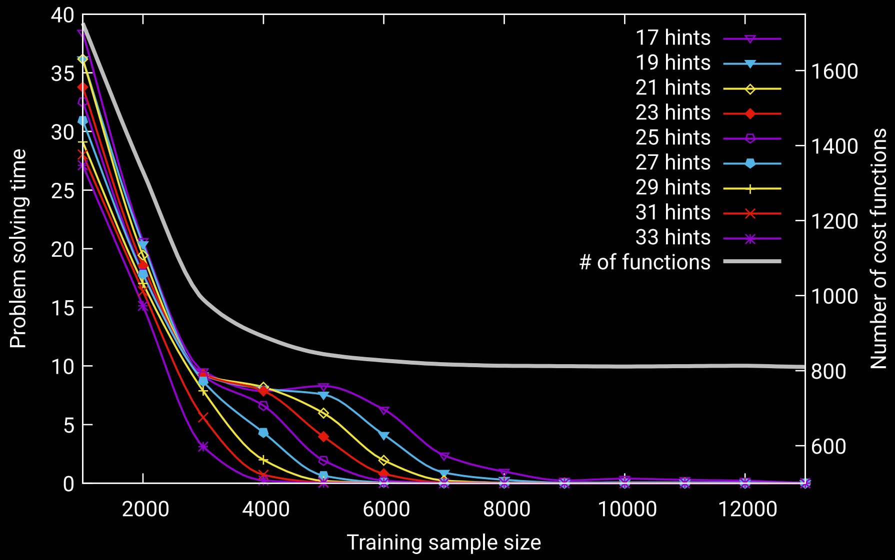
RRN: 96.6%
180,000 samples
(scopes given)

Better with less data and comparable biases



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

Efficient prediction, exact from 13,000 samples



Datasets

- Precomputed sufficient statistics" from RRN training set (8,000 samples)
- PE-MRF with L1-norm Regularization
- Validation set from the SAT-Net paper [Wan+19] (36.2 hints)
- Validation set from the RRN paper [PPW18] with 17 hints.

Learning from uncertain DL output is possible

- LeNet has 99.2% accuracy on handwritten digits
- SAT-Net test set, hints as images (36.2 avg): . . . . 74.7% max. accuracy

Let's try again

- Same $8,000$ sample, λ precomputed
- Hard RRN and SAT-Net test sets
- Toulbar2 is again able to correct LeNet errors

Learning from uncertain DL output is possible

- LeNet has 99.2% accuracy on handwritten digits
- SAT-Net test set, hints as images (36.2 avg): .... 74.7% max. accuracy

Let's try again

- Same $8,000$ sample, λ precomputed
- Hard RRN and SAT-Net test sets
- Toulbar2 is again able to correct LeNet errors

Learning from uncertain DL output is possible

- LeNet has 99.2% accuracy on handwritten digits
- SAT-Net test set, hints as images (36.2 avg): . . . . 74.7% max. accuracy

Let's try again

- Same $8,000$ sample, λ precomputed
- Hard RRN and SAT-Net test sets
- Toulbar2 is again able to correct LeNet errors

Not only Sudokus of course, protein design too and…

If Simon did not cover it, see our CP2020 paper where we show how it can learn user preferences and combine them with configuration constraints on Renault dataset (thanks to H. Fargier (IRIT)). It 's in the `Sudoku-CFN-learn` directory on the VM.

CFN/WCSP solving has made important progress

- Fast approximate LP-bounds (tighter than COP) subsuming AC
- Reduced cost based filtering (cost backpropagation)
- Structure aware search, guided by cost, 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 connection)
- With the (adjustable) power of (exact) solvers

To do

- Global cost function and non monotonicity
- Interval variables and “arithmetic” filtering
- How can we preserve CP efficiency on constraints
- Can we accelerate LP with Soft ACs (universality [PW13])
- Unify CFN with COP: cost variables, multiple criteria
- Improve parallel search
- Can we minimize average tardiness in scheduling with Soft ACs
- Can we improve CFN learning (sample size, (global) constraints)
- Can we integrate CFN and DL more tightly
- ...

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.

[AM00] Srinivas M Aji and Robert J McEliece. “The generalized distributive law”. In: *IEEE transactions on Information Theory* 46.2 (2000), pp. 325–343.

[BB69a] Umberto Bertele and Francesco Brioschi. “A new algorithm for the solution of the secondary optimization problem in non-serial dynamic programming”. In: *Journal of Mathematical Analysis and Applications* 27.3 (1969), pp. 565–574.

[BB69b] Umberto Bertele and Francesco Brioschi. “Contribution to nonserial dynamic programming”. In: *Journal of Mathematical Analysis and Applications* 28.2 (1969), pp. 313–325.

[BB72] Umberto Bertelé and Francesco Brioshi. *Nonserial Dynamic Programming*. Academic Press, 1972.

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

[BH02] E. Boros and P. Hammer. “Pseudo-Boolean Optimization”. In: *Discrete Appl. Math.* 123 (2002), pp. 155–225.

[BLM07] María Luisa Bonet, Jordi Levy, and Felip Manyà. “Resolution for max-sat”. In: *Artificial Intelligence* 171.8-9 (2007), pp. 606–618.

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

[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.

[Dec99] Rina Dechter. “Bucket Elimination: A Unifying Framework for Reasoning”. In: *Artificial Intelligence* 113.1–2 (1999), pp. 41–85.

[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.

[DR03] Rina Dechter and Irina Rish. “Mini-buckets: A general scheme for bounded inference”. In: *Journal of the ACM (JACM)* 50.2 (2003), pp. 107–153.

[DW20a] Tomás Dlask and Tomás Werner. “Bounding Linear Programs by Constraint Propagation: Application to Max-SAT”. In: *Principles and Practice of Constraint Programming - 26th International Conference, CP 2020, Louvain-la-Neuve, Belgium, September 7-11, 2020, Proceedings*. Ed. by Helmut Simonis. Vol. 12333. Lecture Notes in Computer Science. Springer, 2020, pp. 177–193. doi:

10.1007/978-3-030-58475-7_11. URL: https://doi.org/10.1007/978-3-030-58475-7%5C_11.

[DW20b] Tomás Dlask and Tomás Werner. "On Relation Between Constraint Propagation and Block-Coordinate Descent in Linear Programs". In: *Principles and Practice of Constraint Programming - 26th International Conference, CP 2020, Louvain-la-Neuve, Belgium, September 7-11, 2020, Proceedings*. Ed. by Helmut Simonis. Vol. 12333. Lecture Notes in Computer Science. Springer, 2020, pp. 194–210. DOI: 10.1007/978-3-030-58475-7_12. URL: https://doi.org/10.1007/978-3-030-58475-7%5C_12.

[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.

[HG95] W. D. Harvey and M. L. Ginsberg. "Limited Discrepency Search". In: *Proc. of the 14^{th} IJCAI*. Montréal, Canada, 1995.

[HLO07] Federico Heras, Javier Larrosa, and Albert Oliveras. "MiniMaxSat: A New Weighted Max-SAT Solver". In: *Proc. of SAT'2007*. LNCS 4501. Lisbon, Portugal, May 2007, pp. 41–55.

[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.

[KK75] VA Kovalevsky and VK Koval. "A diffusion algorithm for decreasing energy of max-sum labeling problem". In: *Glushkov Institute of Cybernetics, Kiev, USSR* (1975).

[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.

[Kol15] Vladimir Kolmogorov. "A new look at reweighted message passing". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 37.5 (2015), pp. 919–930.

[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* 19^{th} *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.

[LH05] J. Larrosa and F. Heras. "Resolution in Max-SAT and its relation to local consistency in weighted CSPs". In: *Proc. of the 19th IJCAI*. Edinburgh, Scotland, 2005, pp. 193–198.

[LHG08] Javier Larrosa, Federico Heras, and Simon de Givry. "A logical approach to efficient Max-SAT solving". In: *Artif. Intell.* 172.2-3 (2008), pp. 204–233. URL: `http://dx.doi.org/10.1016/j.artint.2007.05.006`.

[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.

[LRW10] Han Liu, Kathryn Roeder, and Larry Wasserman. "Stability approach to regularization selection (StARS) for high dimensional graphical models". In: *Proceedings of Advances in Neural Information Processing Systems (NIPS 2010)*. Vol. 24. 2010, pp. 1432–1440.

[LS03] J. Larrosa and T. Schiex. "In the quest of the best form of local consistency for Weighted CSP". In: *Proc. of the 18th 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.

[Poh70] Ira Pohl. "Heuristic search viewed as path finding in a graph". In: *Artificial intelligence* 1.3–4 (1970), pp. 193–204.

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

[PW13] Daniel Prusa and Tomas Werner. "Universality of the local marginal polytope". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 1738–1743.

[PW15] Daniel Prusa and Tomas Werner. "Universality of the local marginal polytope". In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 37.4 (2015), pp. 898–904.

[RBW06] F. Rossi, P. van Beek, and T. Walsh, eds. *Handbook of Constraint Programming*. Elsevier, 2006.

[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.

[Sha91] G. Shafer. *An Axiomatic Study of Computation in Hypertrees*. Working paper 232. Lawrence: University of Kansas, School of Business, 1991.

[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.

[Son+12] David Sontag et al. “Tightening LP relaxations for MAP using message passing”. In: *arXiv preprint arXiv:1206.3288* (2012).

[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.