Numbers and logic together in CP: A Practical View of Cost Function Networks

CP'2020 tutorial

S. de Givry[1] & T. Schiex[1]

[1] Université Fédérale de Toulouse, ANITI, INRAE MIAT, UR 875, Toulouse, France

This is not a virtual tutorial

September 2020

ANITI
ARTIFICIAL & NATURAL INTELLIGENCE
TOULOUSE INSTITUTE

INRAE
science for people, life & earth

What is a (discrete) graphical model?

Informally (see also [CGS20])

A description of a multivariate function as the combination of a set of simple functions

WHAT FOR?

Concisely describing and analyzing complex systems

- Concise: we use a set of *small* functions
- Complex: the joint function results from the interaction of several small functions

Example

- A digital circuit — value of the output
- A Sudoku grid — solution or not
- A schedule or a time-table — feasibility, acceptability
- A pedigree with genotypes [SGS08] — Mendel consistency, probability
- A frequency assignment [Cab+99] — interference amount
- A 3D molecule [All+14] — energy, stability

Concisely describing and analyzing complex systems

- Concise: we use a set of *small* functions
- Complex: the joint function results from the interaction of several small functions

Example

- A digital circuit — value of the output
- A Sudoku grid — solution or not
- A schedule or a time-table — feasibility, acceptability
- A pedigree with genotypes [SGS08] — Mendel consistency, probability
- A frequency assignment [Cab+99] — interference amount
- A 3D molecule [All+14] — energy, stability

ANITI INRAE

Concisely describing and analyzing complex systems

- Concise: we use a set of *small* functions
- Complex: the joint function results from the interaction of several small functions

Example

- A digital circuit — value of the output
- A Sudoku grid — solution or not
- A schedule or a time-table — feasibility, acceptability
- A pedigree with genotypes [SGS08] — Mendel consistency, probability
- A frequency assignment [Cab+99] — interference amount
- A 3D molecule [All+14] — energy, stability

Concisely describing and analyzing complex systems

- Concise: we use a set of *small* functions
- Complex: the joint function results from the interaction of several small functions

Example

- A digital circuit — value of the output
- A Sudoku grid — solution or not
- A schedule or a time-table — feasibility, acceptability
- A pedigree with genotypes [SGS08] — Mendel consistency, probability
- A frequency assignment [Cab+99] — interference amount
- A 3D molecule [All+14] — energy, stability

Concisely describing and analyzing complex systems

- Concise: we use a set of *small* functions
- Complex: the joint function results from the interaction of several small functions

Example

- A digital circuit — value of the output
- A Sudoku grid — solution or not
- A schedule or a time-table — feasibility, acceptability
- A pedigree with genotypes [SGS08] — Mendel consistency, probability
- A frequency assignment [Cab+99] — interference amount
- A 3D molecule [All+14] — energy, stability

Concisely describing and analyzing complex systems

- Concise: we use a set of *small* functions
- Complex: the joint function results from the interaction of several small functions

Example

A digital circuit	value of the output
A Sudoku grid	solution or not
A schedule or a time-table	feasibility, acceptability
A pedigree with genotypes [SGS08]	Mendel consistency, probability
A frequency assignment [Cab+99]	interference amount
A 3D molecule [All+14]	energy, stability

Concisely describing and analyzing complex systems

- Concise: we use a set of *small* functions
- Complex: the joint function results from the interaction of several small functions

Example

- A digital circuit — value of the output
- A Sudoku grid — solution or not
- A schedule or a time-table — feasibility, acceptability
- A pedigree with genotypes [SGS08] — Mendel consistency, probability
- A frequency assignment [Cab+99] — interference amount
- A 3D molecule [All+14] — energy, stability

Notations

- Variables: $X, Y, Z, \dots$, possibly indexed as X_i
- Domains: D^X for variable X, or D^i for variable X_i
- Unknown values: $u, v, w, x, y, z \dots$
- Sequence of variables: $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \dots$
- Sequence of possible values: $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \dots$
- Domain of a sequence of variables $\boldsymbol{X} : D^{\boldsymbol{X}}$ (Cartesian product of the domains)
- $\boldsymbol{u} \in D^{\boldsymbol{X}}$ is an assignment of $\boldsymbol{X}$ (a value for each variable in $\boldsymbol{X}$)
- $\boldsymbol{u}[\boldsymbol{Y}]$: projection of $\boldsymbol{u}$ on $\boldsymbol{Y}$ (the sequence of values of $\boldsymbol{Y}$ in $\boldsymbol{u}$)

Notations

- Variables: X, Y, Z, . . ., possibly indexed as X_i
- Domains: D^X for variable X, or D^i for variable X_i
- Unknown values: $u, v, w, x, y, z \ldots$
- Sequence of variables: $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \ldots$
- Sequence of possible values: $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \ldots$
- Domain of a sequence of variables $\boldsymbol{X} : D^{\boldsymbol{X}}$ (Cartesian product of the domains)
- $\boldsymbol{u} \in D^{\boldsymbol{X}}$ is an assignment of $\boldsymbol{X}$ (a value for each variable in $\boldsymbol{X}$)
- $\boldsymbol{u}[\boldsymbol{Y}]$: projection of $\boldsymbol{u}$ on $\boldsymbol{Y}$ (the sequence of values of $\boldsymbol{Y}$ in $\boldsymbol{u}$)

Notations

- Variables: X, Y, Z, ..., possibly indexed as X_i
- Domains: D^X for variable X, or D^i for variable X_i
- Unknown values: u, v, w, x, y, $z \dots$
- Sequence of variables: $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}, \dots$
- Sequence of possible values: $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$, $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z} \dots$
- Domain of a sequence of variables $\boldsymbol{X} : D^{\boldsymbol{X}}$ (Cartesian product of the domains)
- $\boldsymbol{u} \in D^{\boldsymbol{X}}$ is an assignment of $\boldsymbol{X}$ (a value for each variable in $\boldsymbol{X}$)
- $\boldsymbol{u}[\boldsymbol{Y}]$: projection of $\boldsymbol{u}$ on $\boldsymbol{Y}$ (the sequence of values of $\boldsymbol{Y}$ in $\boldsymbol{u}$)

Notations

- Variables: $X, Y, Z, \ldots$, possibly indexed as X_i
- Domains: D^X for variable X, or D^i for variable X_i
- Unknown values: $u, v, w, x, y, z \ldots$
- Sequence of variables: $\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}, \ldots$
- Sequence of possible values: $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \ldots$
- Domain of a sequence of variables $\boldsymbol{X}$: $D^{\boldsymbol{X}}$ (Cartesian product of the domains)
- $\boldsymbol{u} \in D^{\boldsymbol{X}}$ is an assignment of $\boldsymbol{X}$ (a value for each variable in $\boldsymbol{X}$)
- $\boldsymbol{u}[\boldsymbol{Y}]$: projection of $\boldsymbol{u}$ on $\boldsymbol{Y}$ (the sequence of values of $\boldsymbol{Y}$ in $\boldsymbol{u}$)

Notations

- Variables: X, Y, Z, …, possibly indexed as X_i
- Domains: D^X for variable X, or D^i for variable X_i
- Unknown values: u, v, w, x, y, $z \dots$
- Sequence of variables: $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}$, …
- Sequence of possible values: $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$, $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z} \dots$
- Domain of a sequence of variables $\boldsymbol{X} : D^{\boldsymbol{X}}$ (Cartesian product of the domains)
- $\boldsymbol{u} \in D^{\boldsymbol{X}}$ is an assignment of $\boldsymbol{X}$ (a value for each variable in $\boldsymbol{X}$)
- $\boldsymbol{u}[\boldsymbol{Y}]$: projection of $\boldsymbol{u}$ on $\boldsymbol{Y}$ (the sequence of values of $\boldsymbol{Y}$ in $\boldsymbol{u}$)

Notations

- Variables: X, Y, Z, ..., possibly indexed as X_i
- Domains: D^X for variable X, or D^i for variable X_i
- Unknown values: u, v, w, x, y, $z \dots$
- Sequence of variables: $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}, \dots$
- Sequence of possible values: $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$, $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z} \dots$
- Domain of a sequence of variables $\boldsymbol{X} : D^{\boldsymbol{X}}$ (Cartesian product of the domains)
- $\boldsymbol{u} \in D^{\boldsymbol{X}}$ is an assignment of $\boldsymbol{X}$ (a value for each variable in $\boldsymbol{X}$)
- $\boldsymbol{u}[\boldsymbol{Y}]$: projection of $\boldsymbol{u}$ on $\boldsymbol{Y}$ (the sequence of values of $\boldsymbol{Y}$ in $\boldsymbol{u}$)

Notations

- Variables: X, Y, Z, . . ., possibly indexed as X_i
- Domains: D^X for variable X, or D^i for variable X_i
- Unknown values: u, v, w, x, y, z . . .
- Sequence of variables: $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}$, . . .
- Sequence of possible values: $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$, $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$ . . .
- Domain of a sequence of variables $\boldsymbol{X} : D^{\boldsymbol{X}}$ (Cartesian product of the domains)
- $\boldsymbol{u} \in D^{\boldsymbol{X}}$ is an assignment of $\boldsymbol{X}$ (a value for each variable in $\boldsymbol{X}$)
- $\boldsymbol{u}[\boldsymbol{Y}]$: projection of $\boldsymbol{u}$ on $\boldsymbol{Y}$ (the sequence of values of $\boldsymbol{Y}$ in $\boldsymbol{u}$)

Notations

- Variables: X, Y, Z, ..., possibly indexed as X_i
- Domains: D^X for variable X, or D^i for variable X_i
- Unknown values: u, v, w, x, y, $z \dots$
- Sequence of variables: $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}$, $\dots$
- Sequence of possible values: $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$, $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z} \dots$
- Domain of a sequence of variables $\boldsymbol{X}$: $D^{\boldsymbol{X}}$ (Cartesian product of the domains)
- $\boldsymbol{u} \in D^{\boldsymbol{X}}$ is an assignment of $\boldsymbol{X}$ (a value for each variable in $\boldsymbol{X}$)
- $\boldsymbol{u}[\boldsymbol{Y}]$: projection of $\boldsymbol{u}$ on $\boldsymbol{Y}$ (the sequence of values of $\boldsymbol{Y}$ in $\boldsymbol{u}$)

ANITI INRAE

Definition (Graphical Model (GM [Bis06; KF09]))

A GM $\mathcal{M} = \langle \boldsymbol{V}, \Phi \rangle$ is defined by:

- a sequence of variables $\boldsymbol{V}$ — n
- each $X \in \boldsymbol{V}$ has finite domain D^X — max size d
- a set Φ of functions (or factors) — e
- Each function $\varphi_{\boldsymbol{S}} \in \Phi$ is a function from $D^{\boldsymbol{S}} \to B$ — scope $\boldsymbol{S}$, arity $|\boldsymbol{S}|$

Definition ($\mathcal{M}$ joint function)

$$\Phi_{\mathcal{M}}(\boldsymbol{v}) = \bigoplus_{\varphi_{\boldsymbol{S}} \in \Phi} \varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}])$$

Definition (Graphical Model (GM [Bis06; KF09]))

A GM $\mathcal{M} = \langle \boldsymbol{V}, \Phi \rangle$ is defined by:

- a sequence of variables $\boldsymbol{V}$ — n
- each $X \in \boldsymbol{V}$ has finite domain D^X — max size d
- a set Φ of functions (or factors) — e
- Each function $\varphi_{\boldsymbol{S}} \in \Phi$ is a function from $D^{\boldsymbol{S}} \rightarrow B$ — scope $\boldsymbol{S}$, arity $|\boldsymbol{S}|$

Definition ($\mathcal{M}$ joint function)

$$\Phi_{\mathcal{M}}(\boldsymbol{v}) = \bigoplus_{\varphi_{\boldsymbol{S}} \in \Phi} \varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}])$$

ANITI INRAE

Definition (Graphical Model (GM [Bis06; KF09]))

A GM $\mathcal{M} = \langle \boldsymbol{V}, \Phi \rangle$ is defined by:

- a sequence of variables $\boldsymbol{V}$ — n
- each $X \in \boldsymbol{V}$ has finite domain D^X — max size d
- a set Φ of functions (or factors) — e
- Each function $\varphi_{\boldsymbol{S}} \in \Phi$ is a function from $D^{\boldsymbol{S}} \to B$ — scope $\boldsymbol{S}$, arity $|\boldsymbol{S}|$

Definition ($\mathcal{M}$ joint function)

$$\Phi_{\mathcal{M}}(\boldsymbol{v}) = \bigoplus_{\varphi_{\boldsymbol{S}} \in \Phi} \varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}])$$

ANITI INRAE

Definition (Graphical Model (GM [Bis06; KF09]))

A GM $\mathcal{M} = \langle \boldsymbol{V}, \Phi \rangle$ is defined by:

- a sequence of variables $\boldsymbol{V}$ — n
- each $X \in \boldsymbol{V}$ has finite domain D^X — max size d
- a set Φ of functions (or factors) — e
- Each function $\varphi_{\boldsymbol{S}} \in \Phi$ is a function from $D^{\boldsymbol{S}} \to B$ — scope $\boldsymbol{S}$, arity $|\boldsymbol{S}|$

Definition ($\mathcal{M}$ joint function)

$$\Phi_{\mathcal{M}}(\boldsymbol{v}) = \bigoplus_{\varphi_{\boldsymbol{S}} \in \Phi} \varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}])$$

ANITI INRAE

Definition (Graphical Model (GM [Bis06; KF09]))

A GM $\mathcal{M} = \langle \boldsymbol{V}, \Phi \rangle$ is defined by:

- a sequence of variables $\boldsymbol{V}$ — n
- each $X \in \boldsymbol{V}$ has finite domain D^X — max size d
- a set Φ of functions (or factors) — e
- Each function $\varphi_{\boldsymbol{S}} \in \Phi$ is a function from $D^{\boldsymbol{S}} \to B$ — scope $\boldsymbol{S}$, arity $|\boldsymbol{S}|$

Definition ($\mathcal{M}$ joint function)

$$\Phi_{\mathcal{M}}(\boldsymbol{v}) = \bigoplus_{\varphi_{\boldsymbol{S}} \in \Phi} \varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}])$$

How are functions $\varphi_S \in \Phi$ represented?

- Default: as tensors over B (multidimensional tables)
- Boolean vars: (B-weighted) clauses (disjunction of variables or their negation)
- Arithmetic, polynomes [BH02]
- Predicates (ALL-DIFFERENT [Rég94; LL12],...)

ANITI INRAE

How are functions $\varphi_S \in \Phi$ represented?

- Default: as tensors over B (multidimensional tables)
- Boolean vars: (B-weighted) clauses (disjunction of variables or their negation)
- Arithmetic, polynomes [BH02]
- Predicates (ALL-DIFFERENT [Rég94; LL12],...)

How are functions $\varphi_S \in \Phi$ represented?

- Default: as tensors over B (multidimensional tables)
- Boolean vars: (B-weighted) clauses (disjunction of variables or their negation)
- Arithmetic, polynomes [BH02]
- Predicates (ALL-DIFFERENT [Rég94; LL12],...)

ANITI INRAE

How are functions $\varphi_S \in \Phi$ represented?

- Default: as tensors over B (multidimensional tables)
- Boolean vars: (B-weighted) clauses (disjunction of variables or their negation)
- Arithmetic, polynomes [BH02]
- Predicates (All-Different [Rég94; LL12],...)

Constraint networks [RBW06]/SAT [BHM09] $B = \mathbb{B} = \{t, f\}, \oplus = \wedge$

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

$\mathcal{M}$ defines a joint Boolean feasibility/consistency function

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

ANITI INRAE

Constraint networks [RBW06]/SAT [BHM09] $B = \mathbb{B} = \{t, f\}, \oplus = \wedge$

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

$\mathcal{M}$ defines a joint Boolean feasibility/consistency function

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

Markov Random Fields: $B = \mathbb{R}^+, \oplus = \times$

- a set V of domain variables
- a set Φ of potential functions
- $\varphi_S \in \Phi : \prod_{X \in S} D^X \to \mathbb{R}^+$

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

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

ANITI INRAE

Markov Random Fields: $B = \mathbb{R}^+, \oplus = \times$

- a set V of domain variables
- a set Φ of potential functions
- $\varphi_S \in \Phi : \prod_{X \in S} D^X \to \mathbb{R}^+$

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

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

ANITI INRAE

Cost Function Networks $\mathcal{M}$ [FW92; SFV95; CS04]

$B = \overline{\mathbb{N}}^k, \oplus = \overset{k}{+}$

- a sequence of domain variables $\boldsymbol{V}$
- a set Φ of e numerical functions
- Each function $\varphi_{\boldsymbol{S}} \in \Phi$ is a function from $D^{\boldsymbol{S}} \to \overline{\mathbb{N}}^k$

- $\overline{\mathbb{N}}^k$: elements of $\mathbb{N} \cup \{\infty\}$ bounded by k — k finite or not
- $\overset{k}{+}$ is the k-bounded addition — $\alpha \overset{k}{+} \beta = \min(\alpha + \beta, k)$

$\mathcal{M}$ defines a joint (bounded) integer function

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

Cost Function Networks $\mathcal{M}$ [FW92; SFV95; CS04]

$B = \overline{\mathbb{N}}^k, \oplus = \overset{k}{+}$

- a sequence of domain variables $\boldsymbol{V}$
- a set Φ of e numerical functions
- Each function $\varphi_{\boldsymbol{S}} \in \Phi$ is a function from $D^{\boldsymbol{S}} \to \overline{\mathbb{N}}^k$

- $\overline{\mathbb{N}}^k$: elements of $\mathbb{N} \cup \{\infty\}$ bounded by k — k finite or not
- $\overset{k}{+}$ is the k-bounded addition — $\alpha \overset{k}{+} \beta = \min(\alpha + \beta, k)$

$\mathcal{M}$ defines a joint (bounded) integer function

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

ANITI INRAE

CFN "normal form" — Used inside the solver

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

Main properties

- $\varphi_\varnothing$ is a lower bound of the joint function $\Phi_{\mathcal{M}}$
- $k = 1$: Constraint networks and SAT, $\overset{k}{+}$ is $\wedge$

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 $e = (i,j) \in E : \varphi_{ij} = w(i,j) \times \mathbb{1}[x_i \neq x_j]$
- Hard edges: constraints with costs 0 or ∞ (when $x_i \neq x_j$)

A simple graph

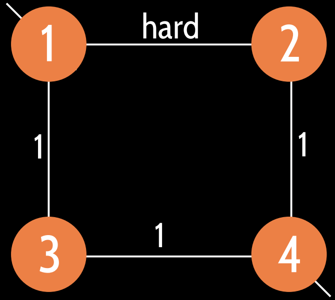
- vertices $\{1, 2, 3, 4\}$
- cut weight 1
- edge $(1, 2)$ hard

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 $e = (i, j) \in E$: $\varphi_{ij} = w(i, j) \times \mathbb{1}[x_i \neq x_j]$
- Hard edges: constraints with costs 0 or ∞ (when $x_i \neq x_j$)

A simple graph

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



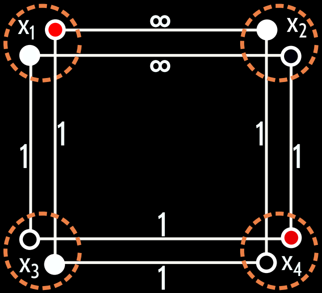
Example: Min-CUT with hard edges

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 $e = (i, j) \in E$: $\varphi_{ij} = w(i, j) \times \mathbb{1}[x_i \neq x_j]$
- Hard edges: constraints with costs 0 or ∞ (when $x_i \neq x_j$)

A simple graph

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



Min-CUT on 4 variables with hard edge

```
{
  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: [0.0,1.0,1.0,0.0]},
      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]}
}
```

Definition (Functions and graphical models equivalence)

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

Definition (Relaxation of a function or graphical model)

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

For $B = \mathbb{B}$, $t < f$

(φ relaxation of φ') $\Leftrightarrow$ ($\varphi' \models \varphi$)

Equivalence, relaxation

Definition (Functions and graphical models equivalence)

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

Definition (Relaxation of a function or graphical model)

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

For $B = \mathbb{B}, t < f$

$(\varphi$ relaxation of $\varphi') \Leftrightarrow (\varphi' \models \varphi)$

ANITI INRAE

Definition (Functions and graphical models equivalence)

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

Definition (Relaxation of a function or graphical model)

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

For $B = \mathbb{B}$, $t < f$

$(\varphi$ relaxation of $\varphi') \Leftrightarrow (\varphi' \models \varphi)$

Optimization

Minimization queries

- $B = \{t \equiv 0, f \equiv 1\}, \oplus = +^{1} = \wedge$, clauses — the SAT Problem
- $B = \{t \equiv 0, f \equiv 1\}, \oplus = +^{1} = \wedge$, tensors — the Constraint Satisfaction Problem
- $B = \overline{\mathbb{N}}^{k}, \oplus = +^{k}$, tensors — the Weighted Constraint Satisfaction Problem

We always use $+^{k}$

Minimization queries

- $B = \{t \equiv 0, f \equiv 1\}, \oplus = \stackrel{1}{+} = \wedge$, clauses — the SAT Problem
- $B = \{t \equiv 0, f \equiv 1\}, \oplus = \stackrel{1}{+} = \wedge$, tensors — the Constraint Satisfaction Problem
- $B = \overline{\mathbb{N}}^k, \oplus = \stackrel{k}{+}$, tensors — the Weighted Constraint Satisfaction Problem

We always use $\stackrel{k}{+}$

ANITI INRAE

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

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

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

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

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

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

$nd + ed^2$ variables, $n + 2ed$ constraints

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

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

Conditioning by $X_2 = b$

X_1		
3	1	2

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

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

Conditioning by $X_2 = b$

X_1		
3	1	2

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])

ANITI INRAE

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]

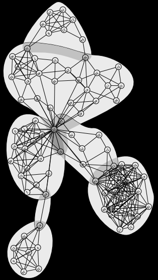
If the $\underbrace{\text{local lower bound}}_{\varphi_\varnothing}$ reaches the $\underbrace{\text{global upper bound}}_{k}$ — Prune!

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])



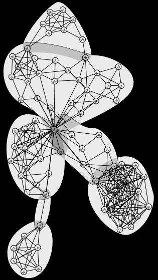
Nice properties

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

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

Hybrid Best First Search [All+15] Anyspace

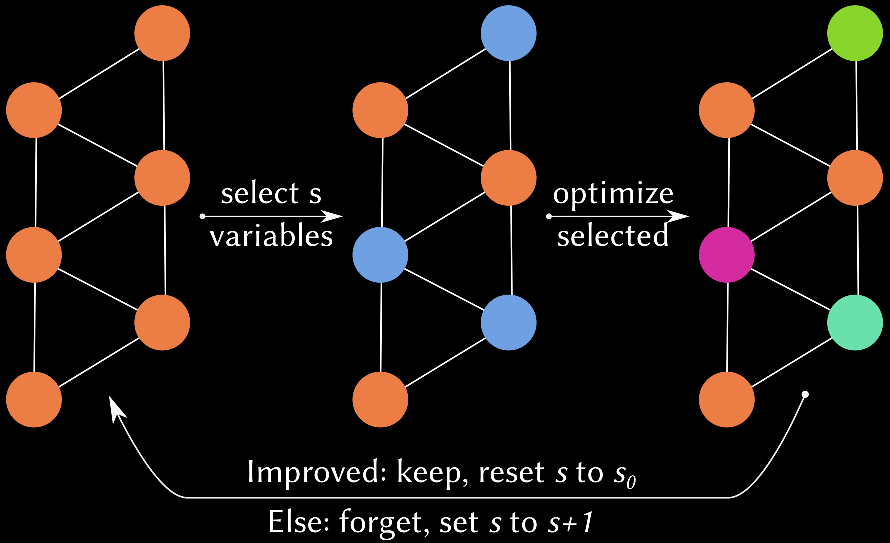
- Uses Depth-First Search for a bounded amount of backtracks
- Pending nodes are pushed onto a list of Open nodes
- The next DFS starts from the best Open node
- Tree-decomposition friendly (BTD [GSV06]/AND-OR search [MD09])



Nice properties

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

Also local search of course (VNS here)



Two last tools: Combination and Elimination

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

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

φ_{X_2}:

X_2	
a	4
b	6
c	4

$\varphi_{X_1 X_2}$:

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

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	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 $X \in S$ 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_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 $\varphi_{\boldsymbol{S}}$ 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: 1	2	3
	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 $\varphi_{\boldsymbol{S}}$ 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

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

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

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

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

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

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

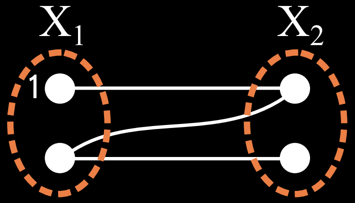
Obvious issue

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

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

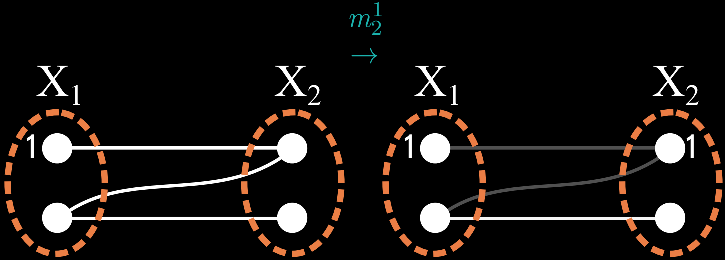
- Add the message m_i^j to φ_j with $+^k$
- Subtract m_i^j from its source using $-^k$

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



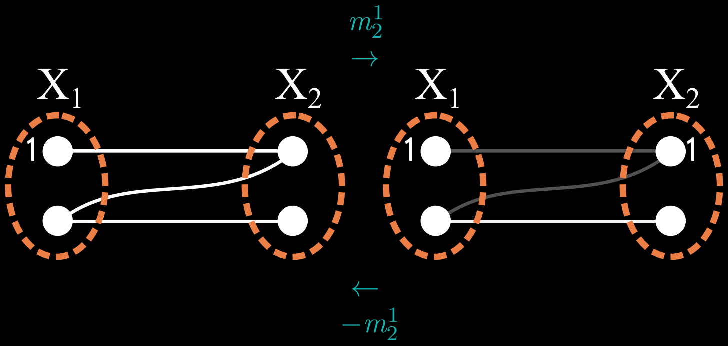
(Loss of) properties

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



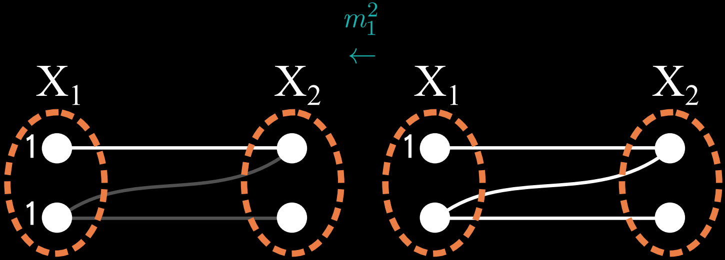
(Loss of) properties

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



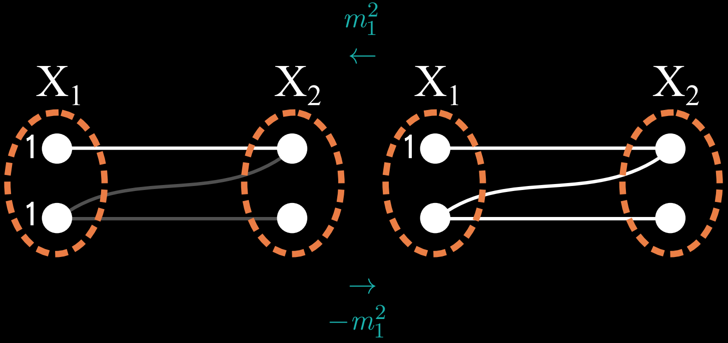
(Loss of) properties

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



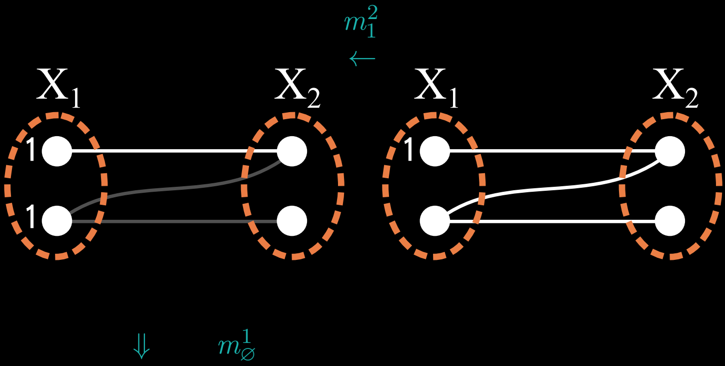
(Loss of) properties

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



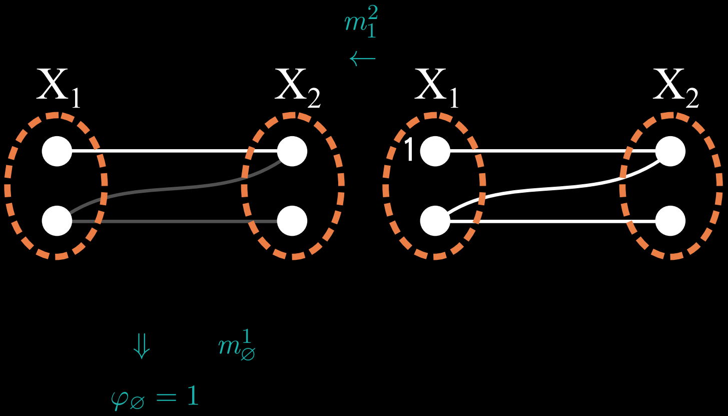
(Loss of) properties

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



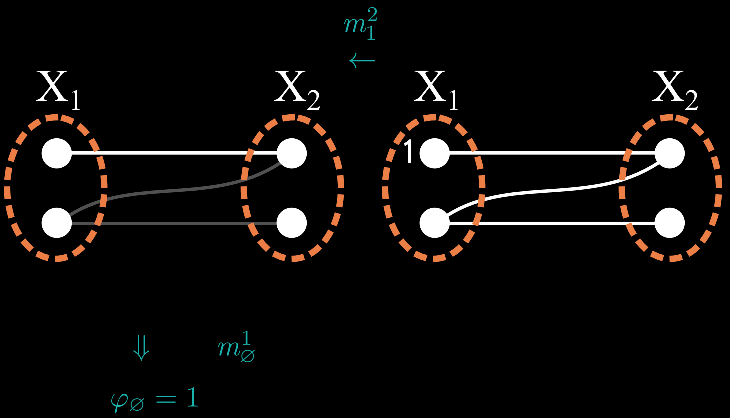
(Loss of) properties

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



(Loss of) properties

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



(Loss of) properties

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

Many way to avoid loops (enforce fixpoint existence)

ANITI INRAE

The many "soft ACs" [Coo+10]

- NC: one unary function [Lar02] — Unary supports ($\varphi_i(u) = 0$)
- +AC: one binary function [Sch00; Lar02] — Arc supports ($v \in D^j, \varphi_{ij}(u, v) = 0$)
- +DAC: FDAC, binary & unary function (+ direction) [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

Properties — Related works in Comp. Vision [Kol06; Son+12; Wer07; Kol15]

- Proper extension of classical NC/DAC or AC respectively — ($k = 1$)
- Polynomial time and $O(ed)$ space — (Generalized ACs)
- Incremental, strengthens $\varphi_\varnothing$ — (VAC $\geq$ EDAC $\geq$ FDAC $\geq$ AC $\geq$ NC)
- May have several fixpoints/$\varphi_\varnothing$

The many "soft ACs" [Coo+10]

- NC: one unary function [Lar02] — Unary supports ($\varphi_i(u) = 0$)
- +AC: one binary function [Sch00; Lar02] — Arc supports ($v \in D^j, \varphi_{ij}(u, v) = 0$)
- +DAC: FDAC, binary & unary function (+ direction) [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

Properties — Related works in Comp. Vision [Kol06; Son+12; Wer07; Kol15]

- Proper extension of classical NC/DAC or AC respectively ($k = 1$)
- Polynomial time and $O(ed)$ space (Generalized ACs)
- Incremental, strengthens $\varphi_\varnothing$ (VAC $\geq$ EDAC $\geq$ FDAC $\geq$ AC $\geq$ NC)
- May have several fixpoints/$\varphi_\varnothing$

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])

Computing a set of rational EPTs maximizing $\varphi_\varnothing$ is in P, solvable by Linear Prog. + AC

Solving the dual of the local polytope + AC enforcing (k)

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])

Computing a set of rational EPTs maximizing $\varphi_\varnothing$ is in P, solvable by Linear Prog. + AC

Solving the dual of the local polytope + AC enforcing (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"

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

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

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

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

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

Local polytope proved to be "Universal for LP" [PW15]

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]

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 cost-based pruning

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

Local consistencies vs. LP

- OSAC empirically very expensive to enforce
- 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

Local consistencies vs. LP

- OSAC empirically very expensive to enforce
- 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

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]

ANITI INRAE

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]

What if the language is CNF?

Soft UP and Max resolution [LH05; BLM07] — More 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:

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

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)

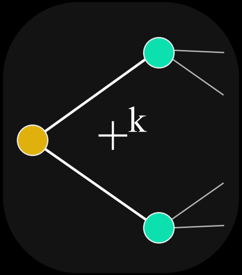
Let $X \in \boldsymbol{V}$, and Φ^X be the set $\{\varphi_{\boldsymbol{S}} \in \Phi \text{ s.t. } X \in \boldsymbol{S}\}$, $\boldsymbol{T}$, the neighbors of X.

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

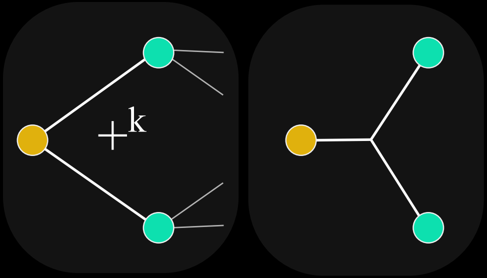
$$m_{\boldsymbol{T}}^{\Phi_X} = \Big(\sum_{\varphi_{\boldsymbol{S}} \in \Phi^X}^{k} \varphi_{\boldsymbol{S}} \Big)[-X]$$

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

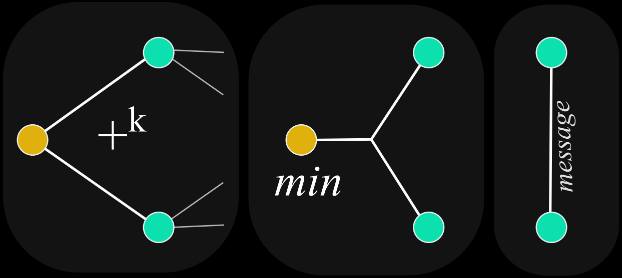
$$\min_{\boldsymbol{v} \in D^{\boldsymbol{V}}} \left[\sum_{\varphi_{\boldsymbol{S}} \in \Phi}^{k} (\varphi_{\boldsymbol{S}}(\boldsymbol{v}[\boldsymbol{S}])) \right] = \min_{\boldsymbol{v} \in D^{\boldsymbol{V}-\{X\}}} \left[\sum_{\varphi_{<\boldsymbol{S}} \in \Phi - \Phi^X \cup \{m_{\boldsymbol{T}}^{\Phi_X}\}}^{k}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



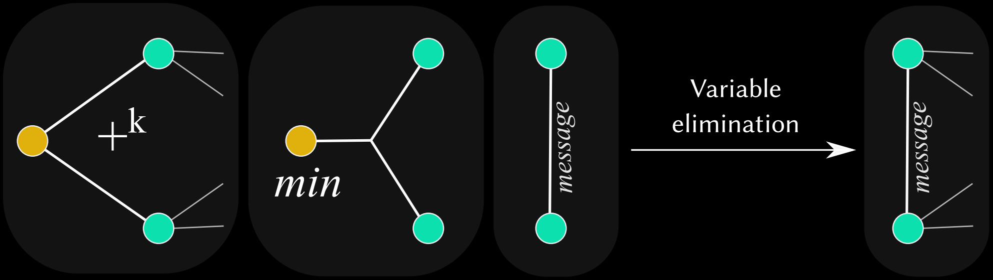
Daoopt & mini-buckets [DR03] split Φ^X in subsets of controlled size (lower bound)



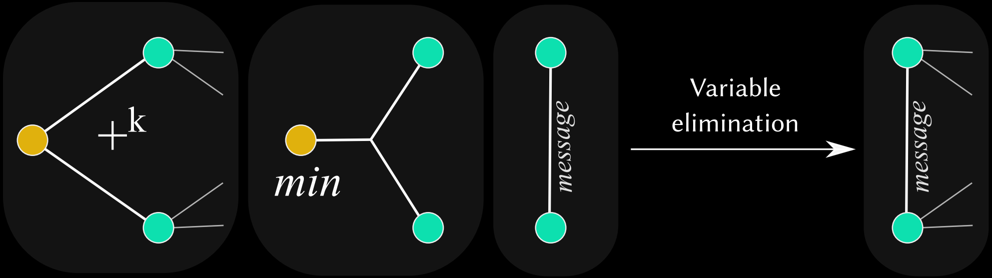
Daoopt & mini-buckets [DR03] split Φ^X in subsets of controlled size (lower bound)



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



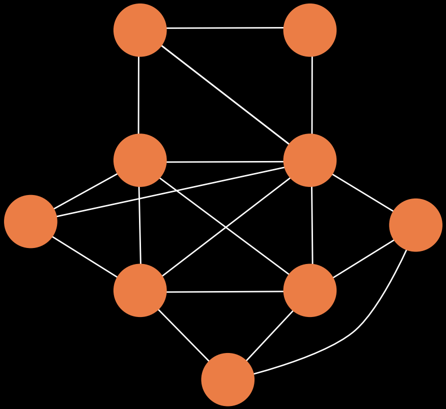
Daoopt & mini-buckets [DR03] split Φ^X in subsets of controlled size (lower bound)



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

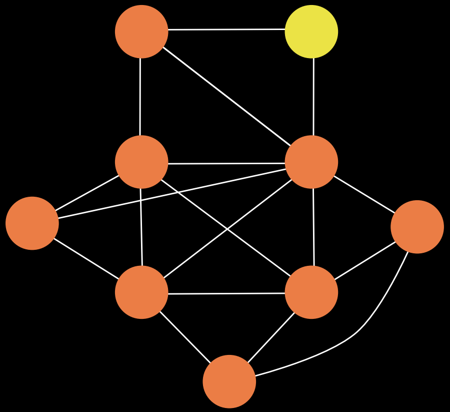
Boosting search with VE [Lar00]

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



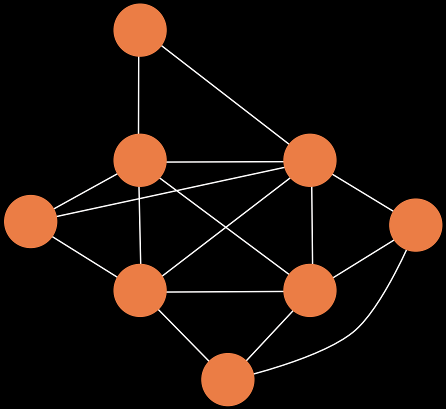
Boosting search with VE [Lar00]

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



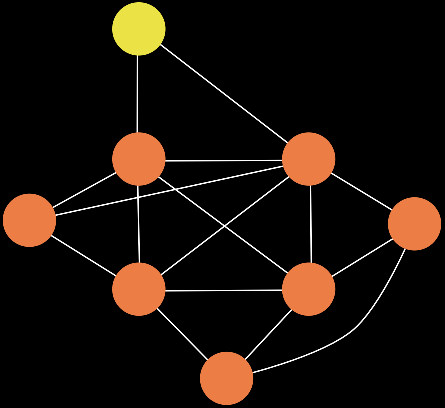
Boosting search with VE [Lar00]

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



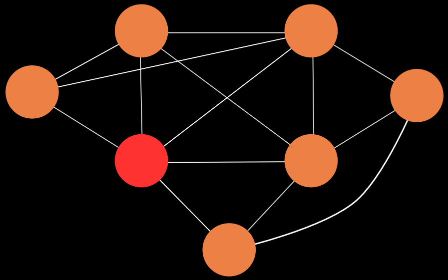
Boosting search with VE [Lar00]

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



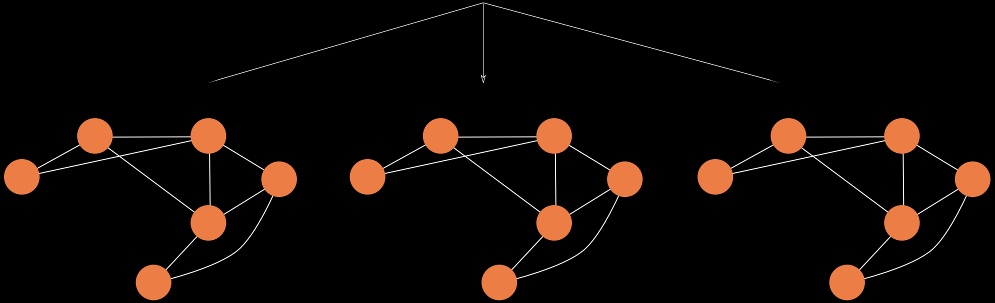
Boosting search with VE [Lar00]

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

- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Value ordering (for free): existential or virtual supports
- Dominance analysis (substitutability/DEE) [Fre91; 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]
- Parallel decomposed Variable Neighborhood Search/LDS (UPDGVNS [Oua+20])

Additional algorithmic ingredients

- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Value ordering (for free): existential or virtual supports
- Dominance analysis (substitutability/DEE) [Fre91; 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]
- Parallel decomposed Variable Neighborhood Search/LDS (UPDGVNS [Oua+20])

Additional algorithmic ingredients

- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Value ordering (for free): existential or virtual supports
- Dominance analysis (substitutability/DEE) [Fre91; 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]
- Parallel decomposed Variable Neighborhood Search/LDS (UPDGVNS [Oua+20])

Additional algorithmic ingredients

- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Value ordering (for free): existential or virtual supports
- Dominance analysis (substitutability/DEE) [Fre91; 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]
- Parallel decomposed Variable Neighborhood Search/LDS (UPDGVNS [Oua+20])

ANITI INRAE

Additional algorithmic ingredients

- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Value ordering (for free): existential or virtual supports
- Dominance analysis (substitutability/DEE) [Fre91; 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]
- Parallel decomposed Variable Neighborhood Search/LDS (UPDGVNS [Oua+20])

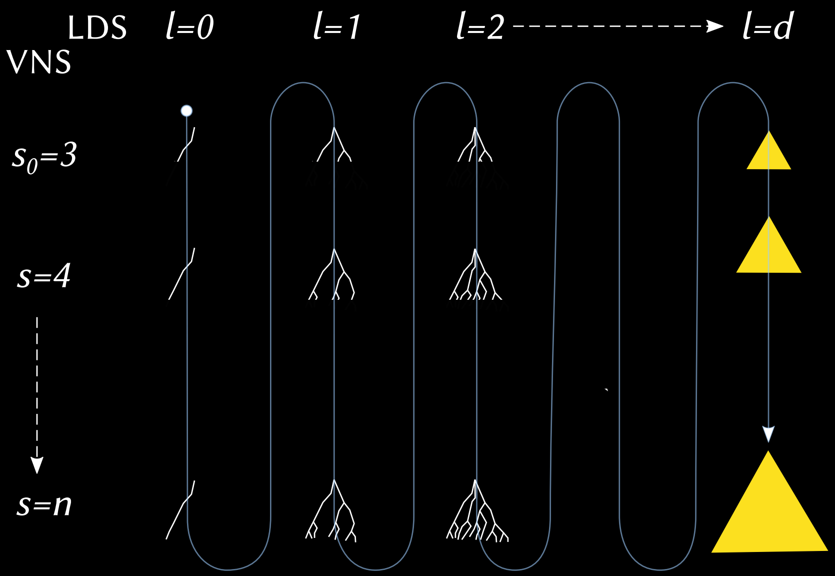
Additional algorithmic ingredients

- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Value ordering (for free): existential or virtual supports
- Dominance analysis (substitutability/DEE) [Fre91; 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]
- Parallel decomposed Variable Neighborhood Search/LDS (UPDGVNS [Oua+20])

Additional algorithmic ingredients

- Variable ordering: weighted degree [Bou+04], last conflict [Lec+09], VAC-based [TGK20]
- Value ordering (for free): existential or virtual supports
- Dominance analysis (substitutability/DEE) [Fre91; 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]
- Parallel decomposed Variable Neighborhood Search/LDS (UPDGVNS [Oua+20])

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



Practical aspects

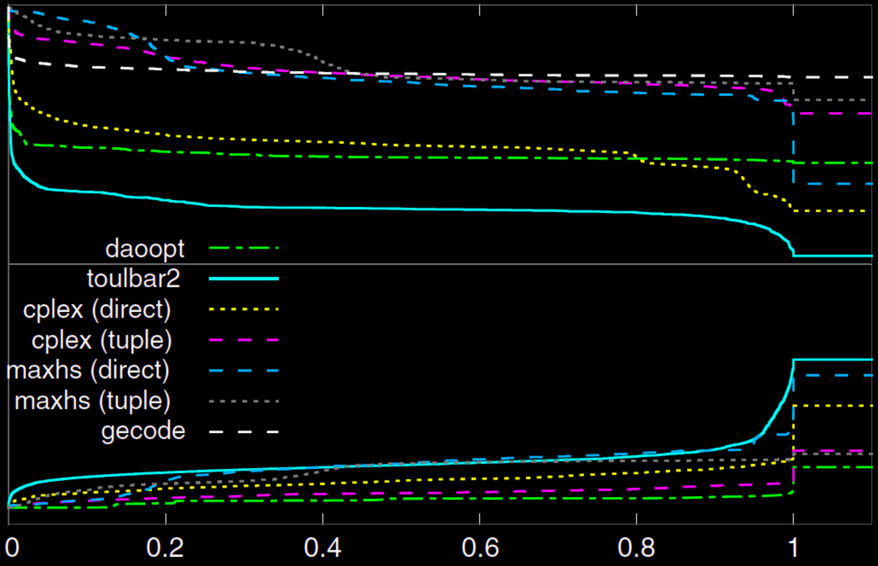
- C++ Open source, MIT licence on GitHub, available in Debian
- Uses 64 bits integer costs to represent adjustable precision decimal costs
- Tackles minimization, maximization with costs of arbitrary signs and constraints
- JSON compatible CFN input format
- Python API (PyToulbar2)

ANITI INRAE

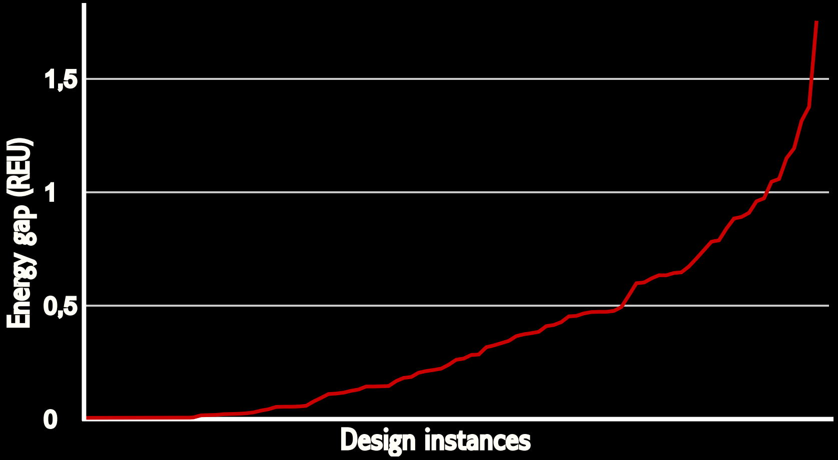
3026 instances of various origins genoweb.toulouse.inra.fr/~degivry/evalgm

- 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

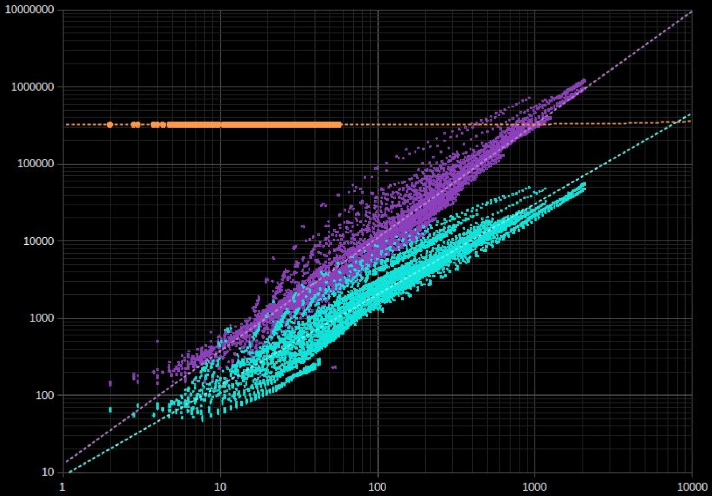
Benchmark	Nb.	UAI	WCSP	LP(direct)	LP(tuple)	WCNF(direct)	WCNF(tuple)	MINIZINC
MRF	319	187MB	475MB	2.4G	2.0GB	518MB	2.9GB	473MB
CVPR	1461	430MB	557MB	9.8GB	11GB	3.0GB	15GB	N/A
CFN	281	43MB	122MB	300MB	3.5GB	389MB	5.7GB	69MB
MaxCSP	503	13MB	24MB	311MB	660MB	73MB	999MB	29MB
WPMS	427	N/A	387MB	433MB	N/A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

Comparison with Rosetta's Simulated annealing [Sim+15]



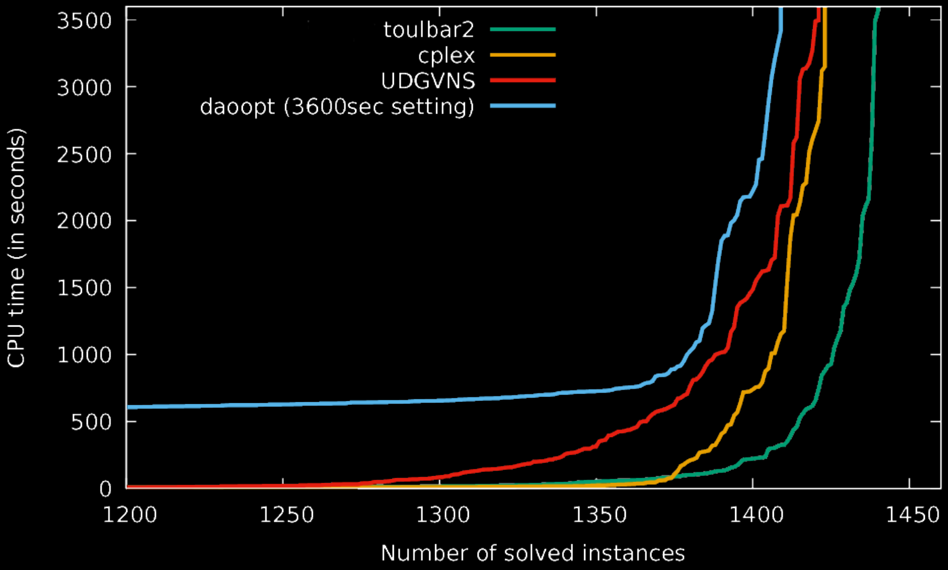
Optimality gap of the Simulated annealing solution as problems get harder

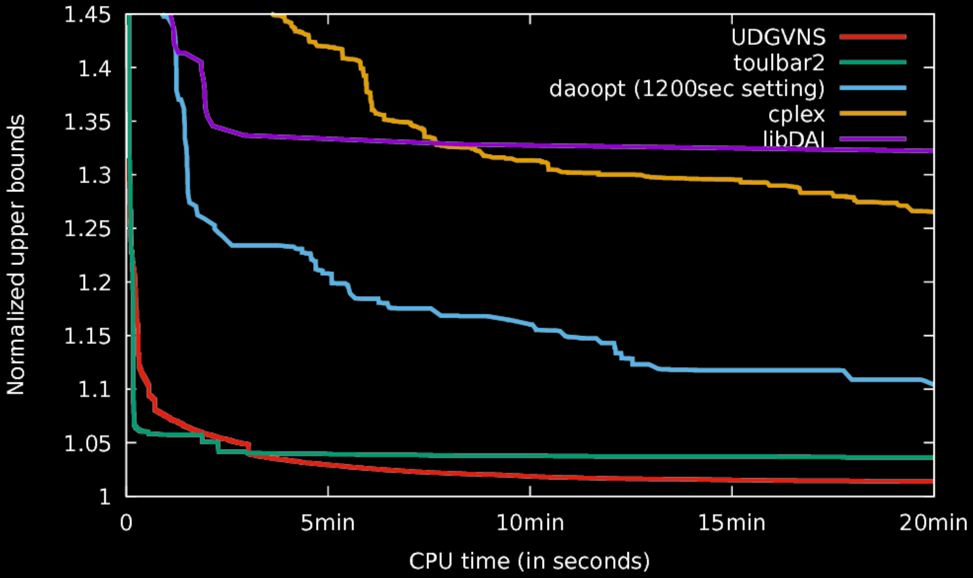


DWave approximations

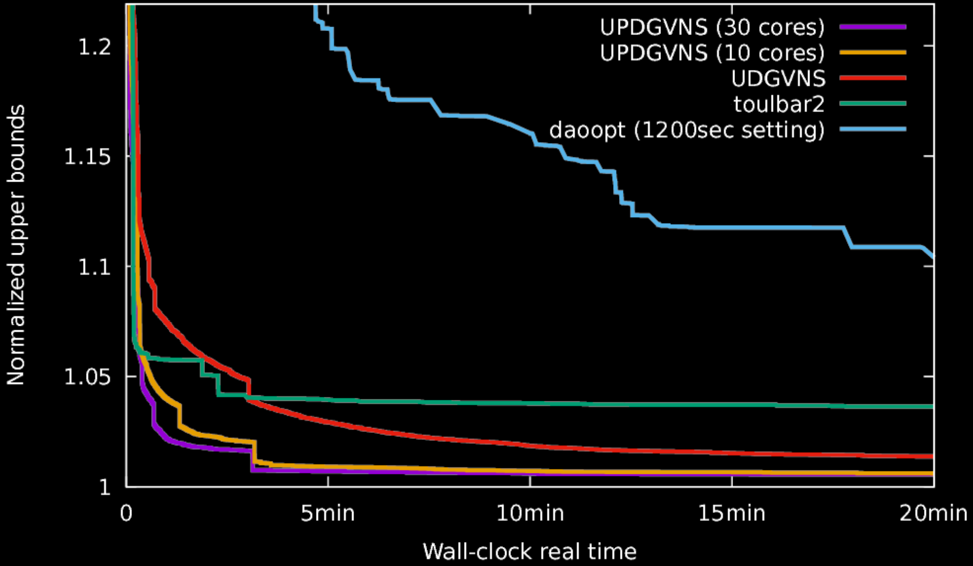
within 1.16 of optimum, 10% of the time | 4.35, 50% of the time | 8.45, 90% of the time

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 - 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

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)

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

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 $\log(\prod_{\boldsymbol{v}\in\boldsymbol{E}} P_{\mathcal{M}}(\boldsymbol{v})) = \sum_{\boldsymbol{v}\in\boldsymbol{E}} \log(P_{\mathcal{M}}(\boldsymbol{v}))$
- Maximimizing loglikelihood over all binary $\mathcal{M}$ $\quad (O(\frac{n(n-1)}{2} d^2)$ costs)

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

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

ANITI INRAE

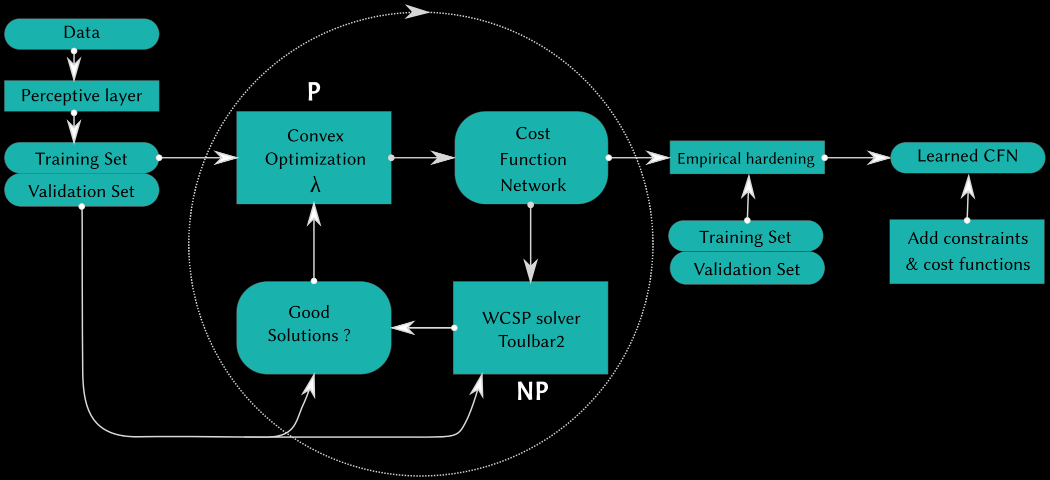
Opens the door to learning from data $\boldsymbol{E}$

- E a set of i.i.d. assignments of $\boldsymbol{V}$
- The log-likelihood of $\mathcal{M}$ given $\boldsymbol{E}$ is $\log(\prod_{\boldsymbol{v}\in\boldsymbol{E}} P_{\mathcal{M}}(\boldsymbol{v})) = \sum_{\boldsymbol{v}\in\boldsymbol{E}} \log(P_{\mathcal{M}}(\boldsymbol{v}))$
- Maximimizing loglikelihood over all binary $\mathcal{M}$ ($O(\frac{n(n-1)}{2} d^2)$ costs)

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

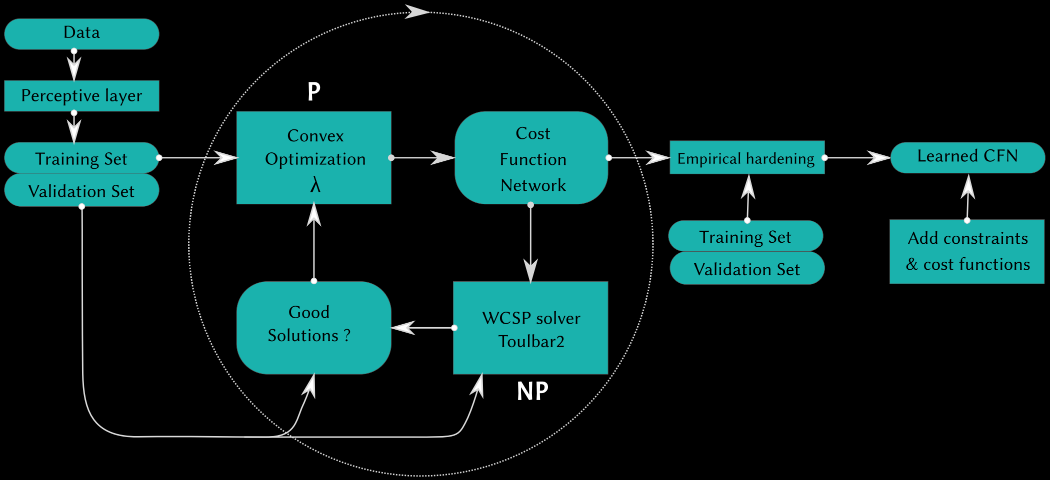
$$\begin{aligned}
\mathcal{L}(\mathcal{M}, \boldsymbol{E}) &= \log(\textstyle\prod_{\boldsymbol{v}\in\boldsymbol{E}} P_{\mathcal{M}}(\boldsymbol{v})) = \sum_{\boldsymbol{v}\in\boldsymbol{E}} \log(P_{\mathcal{M}}(\boldsymbol{v})) \\
&= \textstyle\sum_{\boldsymbol{v}\in\boldsymbol{E}} \log(\Phi_{\mathcal{M}}(\boldsymbol{v}))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

Learning a Cost Function Network from high-quality solutions



See how it learns how to play the Sudoku (and more) Friday 9/11, 1PM session

Learning a Cost Function Network from high-quality solutions



See how it learns how to play the Sudoku (and more) Friday 9/11, 1PM session

[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.

[BHM09] Armin Biere, Marijn Heule, and Hans van Maaren, eds. *Handbook of Satisfiability*. Vol. 185. IOS press, 2009.

[Bis06] Christopher M Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[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.

[Cab+99] B. Cabon et al. "Radio Link Frequency Assignment". In: *Constraints* 4 (1999), pp. 79–89.

[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.

[CGS20] Martin Cooper, Simon de Givry, and Thomas Schiex. “Graphical Models: Queries, Complexity, Algorithms”. In: *Leibniz International Proceedings in Informatics (STACS’2020)* 154 (2020), pp. 4–1.

[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.

[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.

[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.

[FW92] E.C. Freuder and R.J. Wallace. “Partial Constraint Satisfaction”. In: *Artificial Intelligence* 58 (Dec. 1992), pp. 21–70.

[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.

[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.

[KF09] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[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 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.

[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.

[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.

[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.

[POH70] Ira Pohl. "Heuristic search viewed as path finding in a graph". In: *Artificial intelligence* 1.3–4 (1970), pp. 193–204.

[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.

[RÉG94] J.C. Régin. "A filtering algorithm for constraints of difference in CSPs". In: *Proc. of AAAI'94*. Seattle, WA, 1994, pp. 362–367.

[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.

[SFV95] T. Schiex, H. Fargier, and G. Verfaillie. “Valued Constraint Satisfaction Problems: hard and easy problems”. In: *Proc. of the 14^{th} IJCAI*. Montréal, Canada, Aug. 1995, pp. 631–637.

[SGS08] Martí Sánchez, Simon de Givry, and Thomas Schiex. “Mendelian Error Detection in Complex Pedigrees Using Weighted Constraint Satisfaction Techniques”. In: *Constraints* 13.1-2 (2008), pp. 130–154.

[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.

[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.